PRIMECLUSTER

Reliant Monitor Services (RMS) with Wizard Tools
Configuration and Administration Guide 4.3
(Oracle Solaris/Linux)

Continued ▶

Appendix—Attributes

Appendix—Environment variables

Appendix—List of manual pages

Glossary

Abbreviations

Figures

Tables

Index

Contents

Contents

Contents

Contents

1   Preface

Reliant® Monitor Services (RMS) is a software monitor designed to guarantee the high availability of applications in a cluster of nodes. This manual describes how to configure RMS using the RMS Wizards and how to administer RMS using the Cluster Admin GUI.

The manual is aimed at system administrators who create and maintain RMS configurations. Familiarity with the following system functions and components is assumed:

● PRIMECLUSTER family of products

● Oracle Solaris or Linux operating system

● Non-PRIMECLUSTER products such as volume managers and storage area networks.

This document assumes that the PRIMECLUSTER software has been installed as described in the *Software Release Guide PRIMECLUSTER* and the *PRIME-CLUSTER Installation Guide* for your operating system.

1.1   About this manual

This manual is structured as follows:

● The chapter "Introduction" provides an introduction to RMS terminology and describes basic principles of operation.

● The chapter "Using the Wizard Tools interface (hvw)" describes how to configure RMS using the RMS Wizard Tools.

● The chapter "Configuration example" illustrates the Wizard Tools configuration process for two simple applications on a small cluster.

● The chapter "Using the Cluster Admin GUI" describes how to start and use the Cluster Admin graphical user interface (GUI), which is the primary tool for all RMS administrative functions.

● The chapter "Additional administrative tools" describes the RMS clusterwide table, RMS graphs, and the RMS log viewer.

● The chapter "Controlling RMS operation" describes common RMS administrative functions available through the Cluster Admin GUI, including the equivalent CLI procedures.

● The chapter "Appendix—Site preparation" describes network and file settings required for RMS operation.

- The chapter "Appendix—States" lists the object states that are supported by RMS.

- The chapter "Appendix—Object types" lists the object types that are supplied with RMS.

- The chapter "Appendix—Attributes" lists the attributes that are supported by RMS object types.

- The chapter "Appendix—Environment variables" describes the RMS environment variables.

- The chapter "Appendix—RMS command line interface" lists the RMS administrative CLI commands.

- The chapter "Appendix—List of manual pages" lists the manual pages for PRIMECLUSTER CLI commands related to RMS.

- The chapter "Appendix—Release information" lists changes in this manual.

1.2   PRIMECLUSTER manuals

The manuals listed below provide details about PRIMECLUSTER products. Please contact your sales representative for ordering information.

- "PRIMECLUSTE Concepts Guide"-Provides conceptual details on the PRIMECLUSTER family of products.

- "PRIMECLUSTER Installation and Administration Guide"-Provides instructions for installing and upgrading PRIMECLUSTER products.

- "PRIMECLUSTER Cluster Foundation (CF) Configuration and Administration Guide"-Provides instructions for configuring and administering the PRIMECLUSTER Cluster Foundation.

- "PRIMECLUSTER Reliant Monitor Services (RMS) Reference Guide"-Describes operational principles and diagnostic procedures for the RMS high availability manager, including how to view and interpret RMS log files. Provides a list of all RMS error messages with a probable cause and suggested action for each condition.

- "PRIMECLUSTER Global Disk Services Configuration and Administration Guide"-Provides information on configuring and administering Global Disk Services (GDS).

- "PRIMECLUSTER Global File Services Configuration and Administration Guide"-Provides information on configuring and administering Global File Services (GFS).

- "PRIMECLUSTER Global Link Services Configuration and Administration Guide (Redundant Line Control Function)"-Provides information on configuring and administering the redundant line control function for Global Link Services (GLS).

- "PRIMECLUSTER Global Link Services Configuration and Administration Guide (Multipath Function)"-Provides information on configuring and administering the multipath function for Global Link Services (GLS).

- "PRIMECLUSTER Web-Based Admin View Operation Guide"-Provides information on using the Web-Based Admin View management GUI.

- "PRIMECLUSTER DR/PCI Hot Plug User's Guide"-Provides the operation of Dynamic Reconfiguration function and PCI Hot Plug function by using PRIMECLUSTER.

- "PRIMECLUSTER Messages"-Provides how to set the environment for PRIMECLUSTER. Messages output during system operation are also provided.

- Release notices for all products - Release notices provide late-breaking information about installation, configuration, and operation.

1.3 Conventions

To standardize the presentation of material, this manual uses a number of notational, typographical, and syntactical conventions.

1.3.1 Notation

This manual uses the following notational conventions.

1.3.1.1 Prompts

Command line examples that require system administrator (or root) rights to execute are preceded by the system administrator prompt, the hash sign (#). Entries that do not require system administrator rights are preceded by a dollar sign ($).

In some examples, the notation *<nodename>*# indicates a root prompt on the specified node. For example, a command preceded by `shasta1#` would mean that the command was run as user `root` on the node named `shasta1`.

1.3.1.2   Manual page section numbers

References to operating system commands may sometimes be followed by their manual page section numbers in parentheses, *e.g.*, `cp`(1).

1.3.1.3   The keyboard

Keystrokes that represent nonprintable characters are displayed as key icons such as ⌷Enter⌷ or ⌷F1⌷. For example, ⌷Enter⌷ means press the key labeled **Enter**; ⌷Ctrl-b⌷ means hold down the key labeled **Ctrl** or **Control** and then press the ⌷B⌷ key.

1.3.1.4   Typefaces

The following typefaces highlight specific elements in this manual.

Typeface	Usage
`Constant Width`	Computer output and program listings; commands, file names, and other literal programming elements in the main body of text.
Italic	Variables in a command line that you must replace with an actual value. May be enclosed in angle brackets to emphasize the difference from adjacent text, *e.g.*, *<nodename>*`RMS`. Unless directed otherwise, you should not enter the angle brackets. The name of an item in a character-based or graphical user interface. This may refer to a menu item, a radio button, a checkbox, a text input box, a panel, or a window title.
`Bold`	Items in a command line that you must type exactly as shown.

Typeface conventions are shown in the following examples.

1.3.1.5   Example 1

Several entries from an /etc/passwd file are shown below:

```
root:x:0:1:0000-Admin(0000):/:/sbin/ksh
sysadm:x:0:0:System Admin.:/usr/admin:/usr/sbin/sysadm
setup:x:0:0:System Setup:/usr/admin:/usr/sbin/setup
daemon:x:1:1:0000-Admin(0000):/:
```

1.3.1.6   Example 2

To use the cat(1) command to display the contents of a file, enter the following command line:

$ **cat** *file*

1.3.2   Command line syntax

The command line syntax observes the following conventions.

Symbol	Name	Meaning
[ ]	Brackets	Enclose an optional item.
{ }	Braces	Enclose two or more items of which only one is used. The items are separated from each other by a vertical bar (|).
|	Vertical bar	When enclosed in braces, it separates items of which only one is used. When not enclosed in braces, it is a literal element indicating that the output of one program is piped to the input of another.
( )	Parentheses	Enclose items that must be grouped together when repeated.
...	Ellipsis	Signifies an item that may be repeated. If a group of items can be repeated, the group is enclosed in parentheses.

1.4   Important notes and cautions

| i | **Important**
Indicates important information.

⚠ **Caution**
Indicates a situation that can cause harm to data.

⚡ **Note**
Indicates information that needs special attention.

1.5   Abbreviations

Oracle Solaris might be described as Solaris, Solaris Operating System, or Solaris OS.

If "Solaris X" is indicated in the reference manual name of the Oracle Solaris manual, replace "Solaris X" with "Oracle Solaris 10 (Solaris 10)" or "Oracle Solaris 11 (Solaris 11)".

1.6   Revision history

Revision	Location	Manual code
Deleted the attribute "OnlineTimeout."	11.1	J2UZ-5295-04ENZ0(01)
Changed the explanation of configuration files (hvenv and hvenv.local) for global variables and revised whole description.	2.9 2.9.1	J2UZ-5295-04ENZ0(02)
Changed the explanation of the Application-Edit menu of the RMS Wizards.	3.3.2.1	
Added the description to "i" (Important) icon of the Configuration-Activate menu of the RMS Wizards.	3.3.2.1	
Added the explanation of the behavior when exiting maintenance mode.	7.3.3	
Deleted "7.3.4 Clearing faults in maintenance mode."	7.3.4	
Changed the work area from /tmp to /var/tmp.	8.3.1.1	
Changed the possible values of HV_COM_PORT.	12.2	
Changed the possible values of HV_MAXPROC.	12.3	

Revision	Location	Manual code
Changed the explanation of HV_OFFLINE_REASON.	12.4	
Deleted the explanations of the following commands: - hvdist - hvgdmake - hvrclev - hvthrottle	7.1.2 13.1 14.6	
Changed the explanation of the following global environment variables: - RELIANT_LOG_LIFE - RELIANT_LOG_PATH	12.2	J2UZ-5295-04ENZ0(03)
Added the note to HV_SYSLOG_USE.	12.3	

2 Introduction

This chapter contains general information on Reliant Monitor Services (RMS), introduces the PRIMECLUSTER family of products, details how RMS, RMS Wizard Tools, and the RMS Wizard Kit work together to produce high-availability configurations, and introduces Cluster Admin.

Chapter contents:

● "PRIMECLUSTER overview"

● "How RMS provides high availability"

● "How the Wizard Tools provide easy configuration"

● "RMS wizard products"

● "Cluster Admin administration tool"

● "RMS components"

● "RMS CLI"

● "Object types"

● "Object attributes"

● "Environment variables"

● "RMS Directory structure"

2.1 PRIMECLUSTER overview

The PRIMECLUSTER family of products is an integrated set of cluster services, including configuration and administration services, high availability, scalability, parallel application support, cluster file system, and cluster volume management. Figure 1 illustrates the relationship of PRIMECLUSTER services to each other and to the operating system environment.

User interface			
RMS configuration (**Wizard Tools**)	Cluster management and configuration (**Cluster Admin**)		
High availability (**RMS**)	Parallel applications (**PAS**)	Scalable Internet Services (**SIS**)	Custom services (**GDS, GFS, GLS**)
Cluster services (**CF**)			
Operating system kernel			
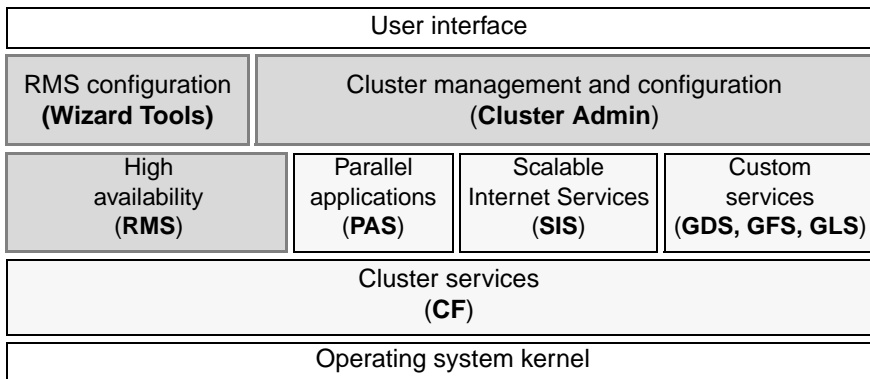
Figure 1: Overview of PRIMECLUSTER products

This manual focuses on PRIMECLUSTER products and services that relate to high availability operation (shown with a solid gray background in the figure above). They are as follows:

- RMS—This high availability manager is a software monitor that provides high availability (HA) for customer applications in a cluster of nodes. Its task is to monitor systems and application resources, to identify any failures, and to provide application availability virtually without interruption in the event of any such failures.

  RMS also provides integrated services for market-specific applications. See your sales representative for availability and details.

- RMS Wizard Tools—This configuration tool provides a character-based interface to create RMS configurations. It includes templates for generic applications and commonly used resources.

  The RMS Wizard Kit works with the Wizard Tools to configure popular enter-prise products for operation with RMS.

- Cluster Admin—The Cluster Admin GUI is the primary administrative tool for RMS.

- Cluster Foundation (CF)—This comprehensive suite provides cluster administration and communication services for user applications and other PRIMECLUSTER services.

PRIMECLUSTER products shown with a dotted background in the figure above are described in their respective manuals. See the section "PRIMECLUSTER manuals" in the Preface.

2.2   How RMS provides high availability

RMS provides high availability of a customer's application by controlling and monitoring the state of all resources in use by a given application. Resources include items such as network interfaces, local and remote file systems, and storage area networks.

2.2.1   Applications, resources, and objects

RMS relies on a virtual representation of the cluster called a **configuration**. The configuration represents each machine, application, and system resource as an object, and the objects are logically arranged in a tree structure according to their dependencies. For instance, suppose a user application depends on a network interface and a file system in order to operate properly. In the tree structure, the corresponding application object would appear as a parent and the network and file system objects would appear as its children. The tree structure is commonly known as a **graph**.

Each object in the graph contains the **state** of the corresponding item along with any other parameters that may be required. An object is typically in the **online** (enabled, available) state or the **offline** (disabled, unavailable) state, but other states are possible according to the type of object. For the complete list of states supported by RMS, see "Appendix—States".

At runtime, the configuration is managed by the **RMS base monitor**, which initiates actions when an object's state changes, or, in the case of a **timeout**, when an object has remained in the same state for some specified time interval even though a change was expected. This design is known as a **state machine**.

Nodes and heartbeats

Machines that are members of a cluster are called **nodes**. When RMS monitors the health of a node, its highest priority is to detect a complete failure of the node or its base monitor. Its second priority is to detect slow response times that may be caused by system overloads. RMS uses two mechanisms to detect these problems.

RMS transmits a UDP **heartbeat** signal at regular intervals. If the elapsed time since the last heartbeat from a node exceeds an adjustable connection timeout, RMS assumes the node has lost connectivity. RMS then begins a recovery period for the node. If the node heartbeat is detected during the recovery period, RMS assumes the node is functional and returns it to normal status. However,

if RMS receives no heartbeats from the node before the recovery period expires, it assumes the node is down, even if other communication with the node is possible.

Once RMS marks a node as down, it takes a series of steps to ensure application and cluster integrity. First, it is necessary to ensure that the node is truly shut down. Otherwise, the node and its applications could unexpectedly recover later, causing conflicts and data corruption. To avoid these problems, RMS directs the Shutdown Facility (described later) to eliminate the node. This is often done by rebooting the node or turning off its power, but the exact action depends on which shutdown agents have been configured for the node. Only after the node has been eliminated is it safe for RMS to restart the node's applications elsewhere in the cluster. The process of automatically switching applications from a failed node to a healthy node is called application **failover**.

Application switchover impacts cluster performance, so it is important to choose a recovery timeout that avoids false detection of node outages. The optimum UDP recovery time depends on the conditions in the cluster. A short recovery period is the best choice to deal with failures of nodes or base monitors. However, a long recovery period allows time for overloaded nodes to respond, which avoids unnecessary shutdowns. If the UDP method is used by itself, these opposing requirements make it difficult to tune the recovery time in large or busy clusters.

In addition, the UDP method can be unreliable, because it has three potential points of failure: first, an outgoing request for a response may not get through to the remote node, so it has no reason to respond; second, the remote node may be so busy that it cannot respond within the recovery period, especially if the recovery timeout is set to a low value; third, a response packet may be sent from the remote node, but it may not get through to the local node. In all three cases, the local node cannot take action until the recovery period expires.

To improve cluster response, RMS uses its Enhanced Lock Manager (ELM) as the primary method to determine machine states and connectivity. ELM is not a polling method. Instead, it relies on locks that are managed at the kernel level by the Cluster Foundation. When a node joins the cluster, it creates a lock and holds it as long as its base monitor is running. The lock is released when the node or its base monitor goes down. The state of the locks is available locally on each node, because the Cluster Foundation maintains them in the background.

ELM is designed to address the high priority issue of node or base monitor failures. The UDP heartbeat can therefore be optimized to detect slow node response, with the recovery time set to a relatively large value. This provides an

important complement to ELM. A node with an overloaded CPU or network interface may respond so slowly that the underlying Cluster Foundation cannot determine the state of the node's lock. If this condition persists, the UDP heartbeat recovery period eventually expires, and RMS proceeds to shut down the node. ELM's efficiency and reliability make this a very infrequent occurrence.

Experts can manually disable ELM for rolling upgrade or debugging operations. In this case, when RMS starts up, the expert must also manually adjust the heartbeat recovery timeout to a smaller value.

Detectors

RMS monitors each resource by using **detectors**, which are processes that deliver status reports to the RMS base monitor process. RMS interprets the status reports to determine the state of the corresponding virtual object. When an object's state changes, RMS takes action according to the parameters set in the object. Each object may be associated with a detector.

Detectors are persistent: when RMS starts on a cluster node it starts the detectors for its configuration, which normally continue to run on that node until RMS is shut down. RMS has the ability to restart a detector if it terminates prematurely.

A complete list of the states that can be reported by detectors or displayed in the user interface is presented later in this chapter.

Note that RMS does not use detector processes to monitor the state of machines in the cluster. Instead, each node transmits a **heartbeat** signal at regular intervals, and RMS uses this to determine machine states and connectivity

Scripts

Each object type has an associated set of **scripts**. A script is a command string (possibly including pipes, redirection, command interpolation, and variable substitution) that can be executed by the operating system shell—in other words, a valid shell script. Normally, each script is designed to interact with items in the operating system such as user applications or physical resources. Scripts provide the only means for RMS to directly influence items outside its virtual representation.

Some scripts are reactive: they define the actions that RMS should take in response to state changes. Other scripts are proactive: they define the actions that RMS should use to take control of individual objects. For instance, RMS

would process one script when a resource reports a transition from the online
state to the offline state; however, RMS would process a different script when it
must force the resource to the offline state.

Scripts are transient: after performing their programmed tasks, they exit and
return a status code to the base monitor.

A complete list of the scripts that may be specified for RMS objects is presented
later in this chapter.

Object types

Most high-availability applications rely on a set of physical resources such as
network interfaces, files systems, or virtual disks. RMS represents these as
`gResource` objects. Most `gResource` objects have scripts that allow them to be
brought online or taken offline.

Internally, RMS represents an actual application that runs in the operating
system environment as a `userApplication` object. The set of `gResource`
objects that represent the actual application's resource requirements are called
its **dependent resources**. Bringing a `userApplication` object to the online
state, along with all of its dependent resources, is called **online processing**.
Taking a `userApplication` object to the offline state, along with all of its
dependent resources, is called **offline processing**.

Machines that are members of a cluster are called nodes.Each node that may run
one or more applications in the high availability configuration is represented by
an RMS `SysNode` object. Like `gResource` objects and `userApplication`
objects, `SysNode` objects can be brought online or taken offline, and they have
an associated set of scripts. However, booting up or shutting down the corre-
sponding physical machine requires more than simple script processing.

For the complete list of the RMS object types supported by the Wizard Tools,
see "Appendix—Object types".

Shutdown Facility

While scripts and detectors provide a direct interface between RMS and the
operating system, the **Shutdown Facility** (SF) provides an indirect interface to
the machines in the cluster. When it necessary to take a `SysNode` object offline,
RMS works with the SF to guarantee that the corresponding node has been
physically shut down, or **killed**. RMS waits for successful completion of the
node kill before switching any `userApplication` from the offline `SysNode` to
another `SysNode`. This prevents any user application from running on two
machines at the same time, which could lead to data corruption.

For more information about the Shutdown Facility, see "PRIMECLUSTER Cluster Foundation (CF) Configuration and Administration Guide" for your operating system.

2.2.2 Relationship of RMS configurations to the real world

It is important to understand that RMS does not interact directly with "real-world" items such as machines, users' applications, or system resources—it interacts only with the objects in its virtual representation. Figure 2 illustrates the relationship between an actual user application in the operating system environment and the corresponding `userApplication` object in an RMS configuration.
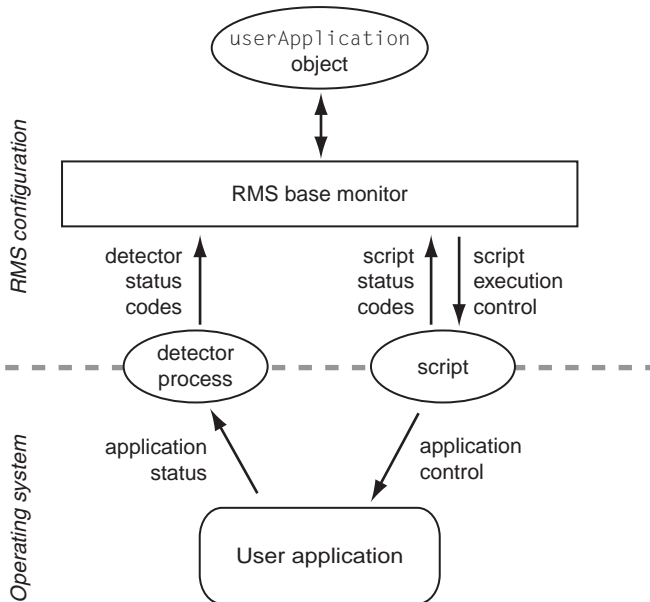

Figure 2: Interface between RMS and the operating system

Note that the interface between the RMS virtual representation and the actual operating system depends entirely on the scripts and detectors provided by the configuration tools. The script in the figure represents any of the standard scripts discussed later in this chapter: it reports whether or not it completed its

tasks successfully by returning a status code, and RMS combines this with the status code from the object's detector to determine the object's state. RMS has no other way to determine what actually happened to the user application in the operating system environment (the part of the figure below the dashed line).

For instance, if a `userApplication` object's `Online` script reports success, its detector reports that it is online, and all of its resources are online, then RMS considers that object to be online, regardless of the state of the actual user application. Similarly, if a resource object's detector reports an `Offline` state, it does not necessarily mean that the physical resource is unavailable.

> **i** For reliable high availability operation, RMS requires scripts that properly control the corresponding real world items, and detectors that accurately reflect the items' states.

Configuration terminology

This manual discusses configuration procedures within the RMS context (represented by the part of Figure 2 above the dashed line). Strictly speaking, our principle concern is with `SysNode` objects, `userApplication` objects, and other RMS entities, and not the real-world items they represent.

However, it is intuitive to use terms such as "node" instead of "`SysNode` object" and "application" instead of "`userApplication` object," because the relationships are so close, and because it is always understood we are working from the RMS perspective. This also helps to simplify many of the technical discussions. Therefore, unless there is a need to distinguish between an RMS object and the actual item it represents, this manual and the configuration tools it describes use the following terms interchangeably:

● "node" and "`SysNode` object" and "`SysNode`"

● "application" and "`userApplication` object" and "`userApplication`"

● "resource" and "`gResource` object" and "`gResource`"

The descriptions of object states and attributes are abbreviated similarly. For instance, instead of "the `gResource` object named *xyz* is in the `Offline` state," it is customary to say, "the *xyz* file system is offline." It is also common to refer to a script by its attribute name, so "the script specified by the `PreOnlineScript` attribute" becomes simply "the `PreOnlineScript`."

2.2.3   Node and application failover

During normal operation, one instance of RMS runs on each node in the cluster. Every instance communicates with the others to coordinate the actions configured for each userApplication. If a node crashes or loses contact with the rest of the cluster, then RMS can switch all userApplication objects from the failed node to a surviving node in the cluster. This operation is known as **failover**.

Failover can also operate with individual applications. Normally, a userApplication object is allowed to be online on only one node at a time. (Exceptions to this rule are shared objects like Oracle RAC vdisk.) If a fault occurs within a resource used by a userApplication object, then only that userApplication can be switched to another node in the cluster. userApplication failover involves offline processing for the object on the first node, followed by online processing for the object on a second node.

There are also situations in which RMS requires a node to be shut down, or **killed**. In any case, before switching applications to a new node, RMS works together with the PRIMECLUSTER Shutdown Facility to guarantee that the original node is completely shut down. This helps to protect data integrity.

RMS also has the ability to recover a resource locally; that is, a faulted resource can be brought back to the online state without switching the entire userApplication to another cluster node.

2.2.4   Controlled applications and controller objects

In some situations, it is desirable for one application to control another in a parent/child relationship. Consider a scenario in which a bank teller application depends on the local network (represented by a network resource object) and a database application. This can be represented by the graph in Figure 3.

Parent application

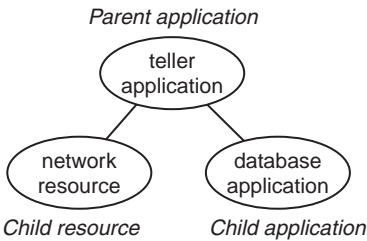
Child resource   Child application

Figure 3: Parent application with two dependencies

Assume that if the network fails in some way, or if the database fails in some way, then the parent teller application cannot complete any transactions. The lines joining the objects in the figure indicate these dependencies. From the RMS perspective, then, we would like both the network resource and the database application to be configured in similar ways: they should both act as dependent resources that must be online if the teller application is to function properly.

However, RMS does not allow any application to be directly configured as the child of another application. Instead, RMS accommodates parent/child relationships between applications by providing an intermediate `controller` object, which is often simply called a **controller**. Unlike other resource objects, a controller has no scripts or detectors. Instead, it propagates online and offline requests from the parent to the child application, and it determines its status from that of the child application.

Figure 4 demonstrates how RMS would represent the banking scenario with the teller application, the controller, and the database application all running on `node1`. For the purposes of this example and the discussions that follow, only the applications and the controller are included in the illustration; the resource object representing the network interface is not shown.
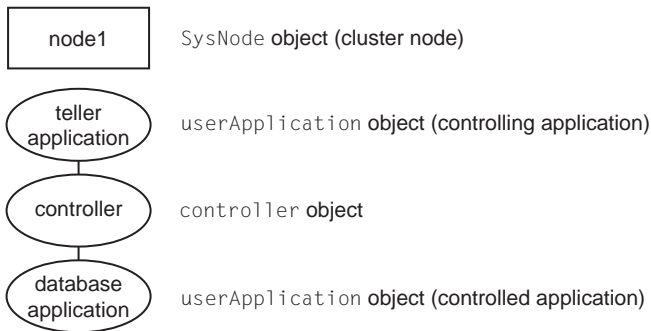
Figure 4: RMS representation of controlled application

> **i** In accordance with the manual of the PRIMECLUSTER Products supported the operation mode of the controller objects, use the control of a user application resource by the controller objects.

> **i** Each controlled application requires a separate controller as a child of the parent application. Also, controllers exist only for internal RMS management purposes—there is no equivalent in the context of the real-world operating system.

If a child changes to an offline or faulted state, RMS will attempt to recover locally or switch the parent, the child, and the dependent resources to another node.

2.2.4.1   Follow controllers

RMS controllers operate in follow mode: the child application must always run on the same node as the parent. If the parent is switched to another node, the application and all its dependent resources will be switched there too. Likewise, if the child application fails in a way that requires it to be switched to another node, then the parent must be switched there as well.

Assume that the parent teller application tree is originally online on `node1` as shown in Figure 4 above. If either the parent or child application needs to be switched to `node2` for any reason, the rest of the tree follows. Figure 5 illustrates the result.
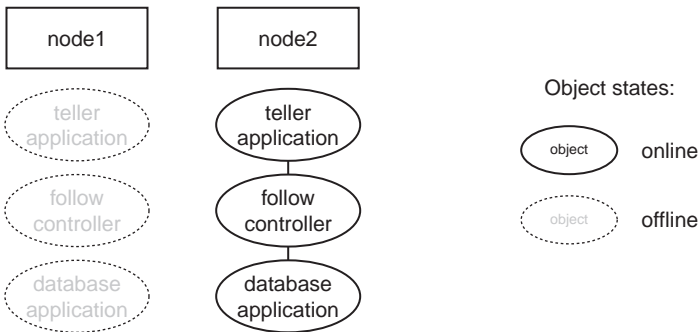
Figure 5: Result of follow-mode switchover

Note the state of the controller in Figure 5. Like the child application, it is brought online only on the same node as the parent. Follow controllers can guarantee that a group of applications and their resources remain closely coupled, so they always run together on the same machine.

i	When RMS switches an application from node1 to node2, no objects are moved within the corresponding graph. Instead, the objects in the part of the graph corresponding to node1 are first taken offline, and then the objects in the part of the graph corresponding to node2 are brought online. The sequence used by RMS in an actual configuration is crucial to high availability operation. For a more detailed discussion, see the "Advanced RMS concepts" chapter in the *RMS Reference Guide*.

2.2.4.2   Scalable controllers

Generally speaking, **scalability** is the ability of the overall system to adapt to changes in resources or workload. One of the main features of RMS scalable controllers is that they allow the parent and child applications to run on separate machines. This not only provides more flexibility but it may also prevent delays or outages when resources fail in certain combinations.

In our banking scenario, for example, the teller application depends on a network and a database application. Suppose the file system on node1 fails and the database goes offline. If the database controller is operating in follow mode, RMS will attempt to switch the teller and database to node2. However, if the network on node2 is offline or faulted, the teller can't be brought online there

either. This is an extreme example, but it illustrates how high loads or resource outages could severely impact a configuration that relies only on a follow controller.

Using a scalable controller can help the configuration adjust dynamically in such situations. If the network is online on node1, while the file system is online on node2, then the database can be switched independently as shown in Figure 6.
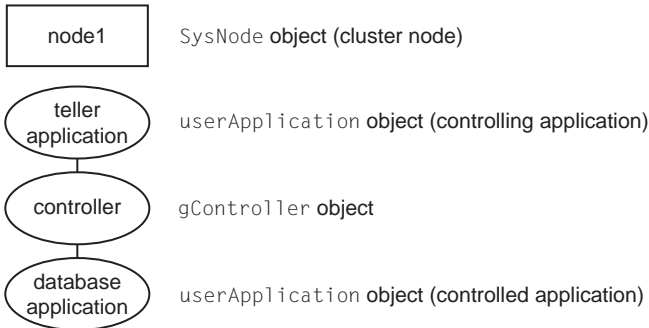
Figure 6: Scalable mode child (controlled) application switchover

Conversely, a network outage could cause RMS to switch the teller to node2 while leaving the database online on node1, as shown in Figure 7.

Figure 7: Scalable mode parent (controlling) application switchover

If an application can run on more than one node, RMS allows only one instance of that application to be online in the cluster. That is, an application can run on only one node at a time. However, controller objects do not have the same

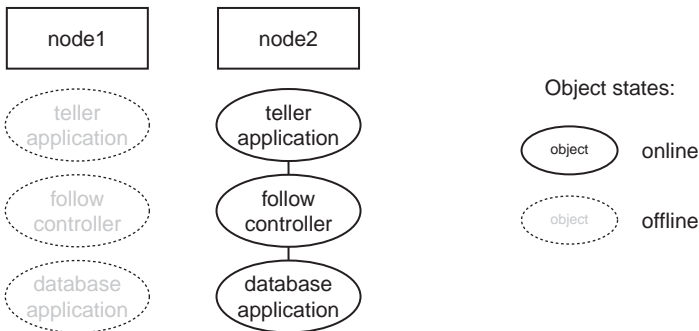restriction. Note the state of the controller objects in Figure 6 and Figure 7. For each scalable child application, an instance of its controller is online on every node where that application can run.

2.2.4.3   Further notes about controllers

The follow and scalable modes are mutually exclusive: a controller for a child application can operate in either follow mode or scalable mode, but not both. The Wizard Tools ensure that each controller's configuration is self-consistent.

A parent application can have more than one child application, but the Wizard Tools require the controller type to be the same for all children. If follow mode is used, then there will be one distinct follow controller for each child. If scalable mode is used, there will be one scalable controller with multiple children. For example, suppose the teller application in the banking scenario also has an automated teller machine (ATM) controlled application. The difference between the follow graph and the scalable graph is shown in Figure 8.
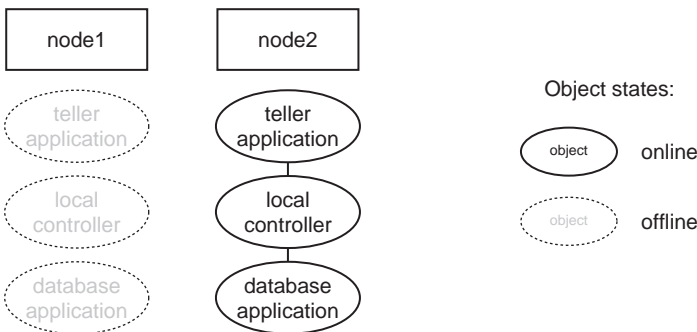
Figure 8: Controller structure with multiple children

For a more detailed discussion, see the *RMS Reference Guide*.

2.3   How the Wizard Tools provide easy configuration

RMS is a mature product with many features and options. Experts who develop, debug, and fine tune complete RMS configurations must know how RMS works and what RMS needs in order to function properly. For each application in the configuration, the expert must do the following:

- Define the set of resources used by the application, including:

  - Disks

  - Volume managers

  - File systems

  - processes to be monitored

  - IP addresses

- Define the relationship between each resource and its dependent resources, *e.g.*, which file system depends on which virtual or physical disk, which processes depend on which file systems, and so forth.

- Define the relationship between the applications being controlled; for example, which applications must be up and running before others are allowed to start.

- Provide scripts to bring each resource online and offline.

- Provide a detector to determine the state of each resource.

Configuring the above set of requirements by hand can be quite time consuming and prone to errors. This is why the RMS Wizard Tools were developed.

The PRIMECLUSTER RMS Wizard Tools allow the creation of flexible and quality-tested RMS configurations while minimizing your involvement. A simple user interface prompts you for details regarding your applications and resources. Using these details, the Wizard Tools automatically select the proper scripts and detectors and combine them in a pre-defined structure to produce a complete RMS configuration.

Specialists skilled in popular applications and in RMS worked together to create the RMS wizards. The wizards are designed to easily configure RMS for certain popular applications such as Oracle or SAP R/3, and they are flexible enough to create custom RMS configurations that can control any other type of application.

2.4   RMS wizard products

The RMS wizards are divided into the following separate products:

- RMS Wizard Tools—user interface, general-purpose application wizards, and basic set of subapplication wizards. Provided as a standard component of RMS

● RMS Wizard Kit—set of custom wizards designed to configure specific appli-
   cations. Available as additional product.

Figure 9 depicts the relationship between RMS, the Wizard Tools, and the RMS
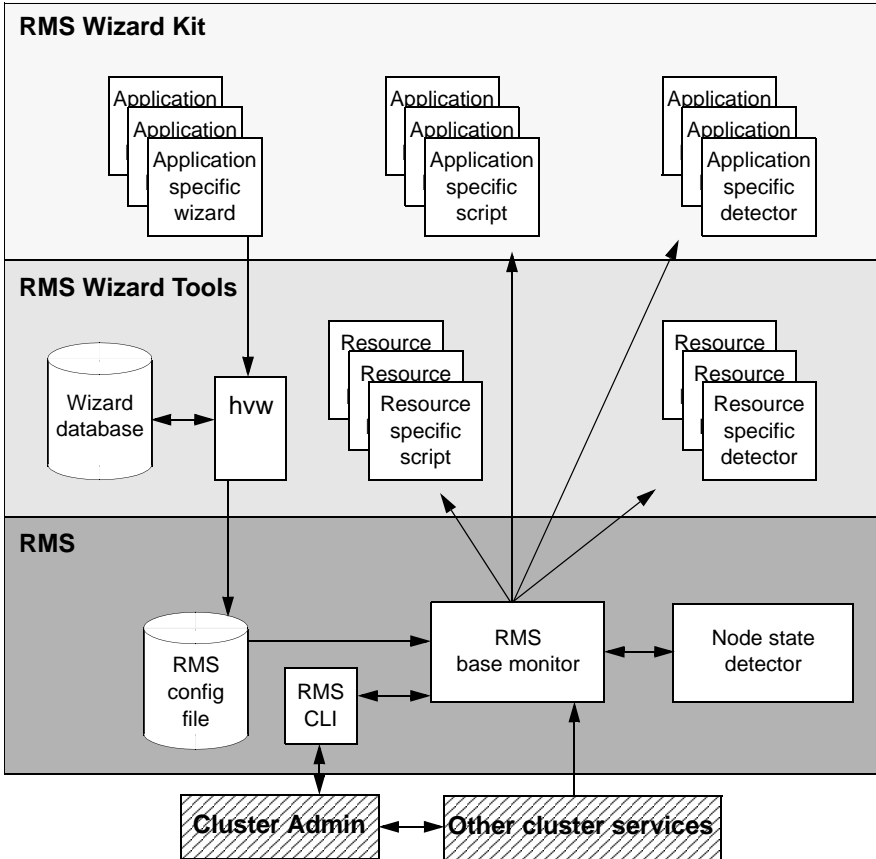Wizard Kit.

Figure 9: Relationship between RMS and RMS Wizards

2.4.1   RMS Wizard Tools

The RMS Wizard Tools provides the following for basic resource types (such as file systems and IP addresses):

● `Online` scripts

● `Offline` scripts

● Detectors

In addition to the basic resource support, the RMS Wizard Tools package contains the `hvw` command, which is the entry point to the user configuration interface. The `hvw` interface provides a simple menu-driven interface to allow a user to enter information specific to applications placed under the control of RMS. `hvw` also provides an interface through which application-specific knowledge can be dynamically added to provide turnkey solutions for those applications typically found in the data center. These application-specific modules are provided by the RMS Wizard Kit.

2.4.2   RMS Wizard Kit

The RMS Wizard Kit provides application knowledge modules which can be used by the `hvw` command. The knowledge modules provide `hvw` with information specific to popular applications, which greatly eases the configuration task. The following are also provided for specific applications:

● `Online` scripts

● `Offline` scripts

● Detectors

> **i**   For information on the availability of the RMS Wizard Kit, contact your local customer support service.

2.5   Cluster Admin administration tool

The Cluster Admin GUI is the primary administrative tool for RMS. It allows users full access to the application control functions of RMS, including the following:

● RMS startup and shutdown

● Application startup and shutdown

● Manual application reset and switchover

● Visual cues for resource and application fault isolation

● Fault clearing capability

● Graphs of application and resources

● Logs for nodes and applications

● Maintenance mode control

2.6   RMS components

The RMS product is made up of the following software components that run on each node in the cluster:

● Base monitor

● Detectors

● Scripts

2.6.1   Base monitor

The base monitor process is the decision-making segment of the RMS process group. It has the following functions:

● Stores the current configuration of resources as represented by objects, their attributes, and their interdependent relationships

● Receives user requests for specific actions from the Cluster Admin graphical user interface (GUI) or the RMS command line interface (CLI)

● Monitors the heartbeat from every node to keep track of each machine's status and its connectivity to the rest of the cluster

- Receives input from detectors and monitors state changes

- Launches scripts to bring applications and their dependent resources online or offline

- Dictates the sequencing of the resource state changes to ensure resources and applications are brought online or offline in the correct order

- Initiates and controls automatic application switchover in case of a resource or node failure, or when directed by a user request

- Performs various administrative functions

2.6.2   Detectors and states

Detectors are independent processes that monitor specific sets of resources in order to determine their state. The detector does not determine whether the current state of a resource is the correct state or not (for example, if a resource is Offline but is supposed to be Online)—that is the role of the base monitor.

Detectors can report the following states to the base monitor:

Faulted   Error condition encountered. The error may have occurred in the resource, in one of its children, or during script processing.

Offline   Disabled, not ready for use. The scripts have successfully disabled the resource.

Online   Enabled, ready for use. All required children are online, and no errors were encountered while scripts were processed.

Standby   Ready to be quickly brought Online when needed.

The following resource states may also be displayed in the GUI status area:

Deact   Applies to userApplication objects only. Operator intervention has deactivated the application throughout the cluster (such as for maintenance purposes).

Inconsistent   Applies to userApplication objects only. The object is Offline or Faulted, but one or more resource objects in its graph have their ClusterExclusive attribute set to 1 and are Online or Faulted.

OfflineFault   Fault that occurred in the past has not yet been cleared.

Unknown No information is available. Reported before object initialization is completed.

Wait   Temporarily in transition to a known state. An action has been initiated for the affected resource, and the system is waiting for the action to be completed

before allocating one of the above states.

Warning   Some warning threshold has been exceeded.

Maintenance   Manual, temporary mode of operation in which the state of an application is decoupled from the states of its dependent resources. This allows, for example, a file system to be taken offline for backup without disturbing the state of its parent application. An application in maintenance mode is usually marked with its **intended state**, which is the state that would be attained if the application were immediately taken out of maintenance mode. The maintenance mode intended states are Maintenance-Online, Maintenance-Offline, and Maintenance-Standby.

The interpretation of Offline and Faulted may depend on the resource type. For instance, a mount point resource can be either Online (mounted) or Offline (not mounted); in this case, the detector would never report the Faulted state. On the other hand, a detector for a physical disk can report either Online (normal operation) or Faulted (input or output error); it would never report Offline.

Detectors for common system functions are provided by the Wizard Tools. Additional application-specific detectors are included with the Wizard Kit.

Some objects provided by RMS, such as the `controller` object, have no detector. Instead, RMS calculates the state of the object based on factors such as transitory processes and the states of its dependent resources.

2.6.3   Scripts

RMS uses scripts to perform actions such as moving a resource from one state to another (for example, from `Offline` to `Online`). The two types of scripts are as follows:

● Request-triggered scripts initiate a state change to a resource.

   The request-triggered scripts are as follows:

   – `InitScript` —Runs only once when RMS is first started

   – `PreCheckScript`—Determines if `Online` or `Standby` processing is needed or possible

   – `PreOfflineScript`—Prepares a transition to an `Offline` state

   – `OfflineScript`—Transitions a resource to an `Offline` state

   – `PreOnlineScript`—Prepares a transition to an `Online` state

- `OnlineScript`—Transitions a resource to an `Online` state

● State-triggered scripts react to specific events.

The state-triggered scripts are as follows:

- `PostOnlineScript`—Reaction to the transition to the `Online` state

- `PostOfflineScript`—Reaction to the transition to the `Offline` state

- `OfflineDoneScript`—Reaction to a `userApplication` reaching the `Offline` state

- `FaultScript`—Reaction to a resource transitioning to the `Faulted` state

- `WarningScript`—Reaction to a detector reporting the `Warning` state (available only for selected resources)

Scripts for common system functions are included with the subapplications provided by the Wizard Tools.

2.6.4   RMS CLI

The primary interface for configuring RMS is the RMS Wizard Tools, and the primary interface for administering RMS is the Cluster Admin GUI. Both the RMS Wizard Tools and Cluster Admin call the RMS CLI, and, under certain conditions, you may find it useful to invoke the CLI directly.

For a list of all commands related to RMS, see the chapter "Appendix—List of manual pages".

i With few exceptions, RMS CLI commands require root privilege. The exceptions are noted in the following table.

Command	Function
hvassert	Tests an RMS resource for a specified resource state. It can be used in scripts when a resource must achieve a specified state before the script can issue the next command. Does not require root privilege.

Table 1: Available CLI commands

Command	Function
hvcm	Starts the base monitor and the detectors for all monitored resources. In most cases, it is not necessary to specify options to the hvcm command. The base monitor is the decision-making module of RMS. It controls the configuration and access to all RMS resources. If a resource fails, the base monitor analyzes the failure and initiates the appropriate action according to the specifications for the resource in the configuration file.
hvconfig	Either displays the current RMS configuration or sends the current configuration to an output file. The output of the hvconfig command is equivalent to the running RMS configuration file, but does not include any comments that are in the original file. Also, the order in which the resources are listed in the output might vary from the actual configuration file.
hvdisp	Displays information about the current configuration for RMS resources. Does not require root privilege.
hvdump	Gets debugging information about RMS on the local node.
hvlogclean	Either saves old log files into a subdirectory whose name is the time RMS was last started, or, if invoked with the -d option, deletes old log files. In either case, hvlogclean creates a clean set of log files even while RMS is running.
hvsetenv	Provides an interface for changing the following RMS environment variables on the local node: – HV_RCSTART controls the automatic startup of RMS. – HV_AUTOSTARTUP controls the automatic startup of all applications. Changes made by this command apply only to the command execution node. For more information about these environment variables, see "Appendix—Environment variables".

Table 1: Available CLI commands

Command	Function
hvshut	Shuts down RMS on one or more nodes in the configuration. The base monitor on the local node sends a message to other online nodes indicating which node or nodes will be shut down.
hvswitch	Manually switches control of a user application resource from one system node to another in the RMS configuration. The resource being switched must be of type userApplication. The system node must be of type SysNode.
hvutil	Provides general administration interface to RMS. It performs various resource administration tasks, such as dynamically setting logging levels, sending a resource Offline, clearing faulted resources or hung cluster nodes in the Wait state, setting detector time periods, setting Maintenance Mode, and so forth.

Table 1: Available CLI commands

2.7   Object types

An object type represents a group of similar resources that are monitored by the same detector (for example, all disk drives). Using the Wizard Tools, you can create configuration files that contain objects of various types, each repre-senting resources or groups of resources to be monitored by RMS. The supported types are as follows:

● SysNode

● userApplication

● gResource

● controller

● andOp

● orOp

Refer to the chapter "Appendix—Object types" for the supported types, their required attributes, and a brief description of each object.

| i | This information is provided for reference only. These objects are created by the Wizard Tools during the generation phase of the configuration process. The type of an object may be listed in diagnostic messages for use by RMS experts. |

2.8   Object attributes

An attribute is the part of an object definition that specifies how the base monitor acts and reacts for a particular resource during normal operation. An attribute can include a device name and configuration scripts. Users can specify attributes in any order in the object definition.

Refer to the chapter "Appendix—Attributes" for the supported types, their associated values, and a description of each attribute.

| i | This information is provided for reference only. The values are determined by the Wizard Tools during the generation phase of the configuration process. Refer to the chapter "Using the Wizard Tools interface (hvw)". |

2.9   Environment variables

RMS uses global and local environment variables:

● Global environment variables must have the same setting on all nodes in the cluster. RMS maintains global environment variables in the `ENV` object.

● Local environment variables can vary from node to node. RMS maintains local environment variables in the `ENVL` object.

RMS creates the `ENV` and `ENVL` objects dynamically from the contents of `hvenv` and `hvenv.local` when the base monitor starts up. Refer to the section "Setting environment variables" for more information.

| i | ● Global variable settings (`ENV`) are included in the configurations checksum that is common to the cluster. The checksum is verified on each node during startup of the base monitor. RMS will fail to start if it detects a checksum difference between the values on any two nodes. |

● Local variable settings (ENVL) are not included in the configurations checksum that is common to the cluster.

While RMS is running, you can display the environment variables with the hvdisp command which does not require root privilege. Use `hvdisp ENV` for global list and `hvdisp ENVL` for local list.

Refer to the chapter "Appendix—Environment variables" for a complete description of each of these variables.

2.9.1   Setting environment variables

When RMS starts, it reads the values of environment variables from `hvenv` and `hvenv.local` and initializes the ENV and ENVL objects respectively. To change the values of environment variables before RMS starts up, you need to specify them in the `hvenv.local` file.These settings will be overridden at installation.

> **i** The `/tmp` directory that is nearly full may result in RMS errors, because the base monitor uses the `sort(1)` command to sort RMS environment variables.

If you adjust a global variable in the `hvenv.local` file on one node, you must make the same adjustment to `hvenv.local` on every other node in the cluster. You will have to restart RMS for the change to take effect.

⚠ **Caution**

Do not explicitly set RMS environment variables in the user environment. Doing so can cause RMS to lose environment variables settings.

The values of environment variables are specified as `export` directives in each file. A typical `export` directive would appear as follows:

```
export SCRIPTS_TIME_OUT=200
```

Therefore, any changes you make to `hvenv.local` file will not take effect until the next time RMS starts up. While RMS is running, you can display the environment variables with the following commands:

● `hvdisp ENV`

● `hvdisp ENVL`

2.9.2   Script execution environment variables

When the RMS invokes a script on behalf of an object, it provides a set of variables in the script's environment that can be used for decision processing at runtime. Since these variables exist only within the context of the script while it is carrying out its tasks, they are not usually visible in the RMS user or administrator environment. In rare cases, they could appear in a diagnostic message in the system log or on the console.

The section "Script execution environment variables" provides a complete description of each of these variables.

2.10   RMS Directory structure

RMS software consists of a number of executables, scripts, files, and commands, all located relative to the directory specified in the `RELIANT_PATH` environment variable. Table 2 illustrates the directory structure of the RMS software after it has been correctly installed.

Name	Contents
`RELIANT_PATH`	Base directory. **Default:** `/opt/SMAW/SMAWRrms`
<RELIANT_PATH>`/bin`	Executables, including detectors, commands, and scripts.
<RELIANT_PATH>`/build`	Work area for configuration files.
<RELIANT_PATH>`/etc`	Miscellaneous files used by RMS and the configuration tools.
<RELIANT_PATH>`/include`	RMS include files (header files) used by detectors and configuration files.
<RELIANT_PATH>`/lib`	RMS runtime libraries.
<RELIANT_PATH>`/us`	RMS source files. The names of the files in this directory are reserved and should not be used to name any configuration files that the user may create.

Table 2: RMS base directory structure

As summarized in Table 3, RMS log files are located in the directory specified in the RELIANT_LOG_PATH environment variable.

Name	Contents
RELIANT_LOG_PATH	Contains files that can be used for RMS analyzing and debugging, including the RMS switchlog. The base monitor and detectors create log files here. **Default:** /var/opt/SMAWRrms/log
	The same directory has subdirectories that contain backup copies of the RMS log files. Each backup subdirectory has a name of the form *yyyy-mm-dd_HH:MM:SS* to indicate the date and time when the backup was created.

Table 3: Log directory structure

3 Using the Wizard Tools interface (hvw)

This chapter describes how to configure high availability for customer applications using the RMS Wizards.

● The section "Overview" gives a brief overall description of the configuration process and the RMS Wizards.

● The section "General configuration procedure" outlines the four major steps involved in every configuration procedure.

● The section "Creating and editing a configuration" describes the wizard interface and how it is used to specify a configuration.

● The section "Activating a configuration" describes how to activate a configuration after it has been created or modified.

● The section "Configuration elements" provides additional details about basic RMS elements specified in every configuration.

All the following procedures assume the Cluster Foundation (CF) software has been properly installed, configured, and started. See the *Cluster Foundation (CF) Configuration and Administration Guide* for details.

3.1 Overview

The chapter "Introduction" describes the components necessary for configuring applications for high availability. It is extremely important that you define applications and the resources that are used by them. Resources are entities like disks, file systems, processes, IP addresses, and so forth.

This definition also needs to include the following information:

● How the applications and their resources are related to each other

● What scripts bring resources online and offline

● Which detectors monitor the state of which resources

For example, if a node should fail to be available, the node that is to take its place must have been defined beforehand so that the applications depending on this node are able to continue operating with minimal interruption. Once the

necessary information is defined, you can then set up an RMS configuration. A configuration of this magnitude, however, requires a great deal of expert knowledge.

The RMS Wizards are tools that allow you to set up an RMS configuration in a way that is simple, flexible, and quality-tested. Furthermore, these tools conform to a well-documented, standard design.To configure RMS with the wizards, you supply information about the applications using a menu-driven interface. The wizards use this information to set up a complete RMS configuration.

The following sections describe these wizards and the way they are used to configure high availability from a general point of view.

3.1.1   RMS Wizard types

The RMS Wizards are divided into two categories:

● RMS Wizard Tools—This is a general-purpose package that includes the following components:

– The `hvw` menu-based configuration interface

– The GENERIC application wizard, which allows you to configure a wide range of applications

– The DEMO wizard, which provides a simple demonstration of the Wizard Tools and RMS

– The basic set of resource-oriented wizards, which provide scripts and detectors for basic resources such as file systems, volume managers, and IP addresses. They are used by the GENERIC and DEMO wizards as well as components in the Wizard Kit.

● RMS Wizard Kit—These application-oriented wizards are designed to cover complete applications and perform their tasks on the basis of the turnkey concept. The R/3 and ORACLE wizards are components of the Wizard Kit.

> **i** For information on the availability of the RMS Wizard Kit, contact your local customer support service.

3.1.1.1   Turnkey wizards

Turnkey wizards provide predefined structures of resources to monitor almost every basic operating system object. This relieves the user of the tedious task of linking system resources according to their dependencies.

Many turnkey wizards are designed to configure a specific type of application. The configuration described in the chapter "Configuration example" uses the GENERIC and DEMO turnkey wizards. Other examples are the R/3 wizard and the ORACLE wizard. By convention, turnkey wizards have names with all uppercase letters.

3.1.1.2   Resource wizards

Resource wizards (sometimes called sub-application wizards) configure lower-level resources such as file systems or IP addresses. They are invoked by turnkey wizards and are not designed to interact directly with the user. Resource wizards have names that begin with one uppercase letter followed by one or more lowercase letters.The following are some of the more important resource wizards:

● Cmdline—Configures any generic resource type by specifying `StartScript` (to bring the resource online), `StopScript` (to send the resource offline) and `CheckScript` (to check the state of a resource).

● Controller—Configures applications that control other applications.

● Fsystem—Configures local or remote file systems.

● Gds—Configures disk classes administrated by Global Disk Services (GDS).

● Gls—Configures the IP addresses administrated by Global Link Services (GLS).

● Ipaddress—Configures the IP addresses that are needed for communication over a LAN interface.

3.2   General configuration procedure

RMS configuration always involves these four steps:

▶ **Stop RMS.**

Refer to the section "Stopping RMS". You can use the Cluster Admin GUI or the command line interface from any node in the cluster.

► **Create or edit the configuration.**

The next section provides general information, and the chapter "Configuration example" walks through an example.

► **Activate the configuration.**

Activation includes generation and distribution. See the section "Activating a configuration".

► **Start RMS.**

Refer to the section "Starting RMS". You can use the Cluster Admin GUI or the command line interface from any node in the cluster.

i	To avoid network access problems, perform RMS configuration tasks as `root`, and ensure that CF is installed, properly configured, and running as described in the *Software Release Guide PRIMECLUSTER* and the *PRIMECLUSTER Installation Guide* for your operating system.

3.3   Creating and editing a configuration

You can bring up an existing Wizard Tools configuration that is currently activated on the host systems of a cluster. In this case, you might call up the configuration because it is to be modified using the wizards while RMS is stopped. On the other hand, you might want to use the wizards to set up a new configuration. The commands for starting the wizards are as follows:

● `hvw`

Runs RMS Wizard Tools using the last activated configuration stored in the *<RELIANT_PATH>*/etc/CONFIG.rms startup file. If this file does not exist or activation is being done for the first time, RMS creates the default configuration, `config`.

● `hvw -n` *configname*

Edits an existing configuration or creates a new configuration using the specified name. The configuration will be stored in the *<RELIANT_PATH>*/build/*configname*.us startup file.

The sample configuration used for demonstration purposes in this chapter shows how to set up a new configuration called mydemo using the DEMO turnkey wizard. This example would be called up as follows:

```
hvw -n mydemo
```

The hvw command is documented in the online manual pages. Refer to the chapter "Appendix—List of manual pages" for additional information.

3.3.1   Using the wizard menus

The hvw command produces character-driven menus that guide you in a way designed to be self-explanatory. The following are some of the most frequently used menu operations and items:

● Selecting items—This is normally done by typing the number of the item followed by the ⏎Enter⏎ or ⏎Return⏎ key. Within the menu, a prompting line indicates the kind of input that is required. A >> prompt indicates that a string of text should be entered.

● Responding to messages—Within the menus, several kinds of messages are displayed. One type of message might be to inform the user about the activities that the wizard has performed; for example, a consistency check that ended in a positive result. Other messages may prompt the user to continue the configuration procedure with a certain activity; for example, choosing an application name.

● *HELP*—This item provides user assistance and is available at the top of every wizard menu.

● *QUIT*—This quits the wizard menu system.

● *RETURN*—This moves one level upward in the menu system; that is, from a subordinate menu to the menu it was called from.

● *SAVE+EXIT* and *NOSAVE+EXIT*—These save or discard your input and then exit. *SAVE+EXIT* will be disabled in read-only mode, and it may be disabled if the configuration is inconsistent at that point.

● *NEXT* and *PREVIOUS*—If all items cannot be listed in a single screen due to many items to be selected, some are displayed in the next screen. Press *NEXT* to go on to the next screen, and press *PREVIOUS* to go back to the previous screen.

3.3.2   Main configuration menu

The *Main configuration menu* appears immediately after a configuration has been called up. This top-level menu shows the state of the RMS cluster by indicating either one the following:

● RMS is inactive

● The list of nodes where RMS is up and running

The *Main configuration menu* changes dynamically at run time depending on whether RMS is running in the cluster and whether the configuration being edited is the current configuration.

If RMS is running anywhere in the cluster, actions that could modify a running configuration are not available. Additionally, the menu items that are available are modified such that no changes can be made to the running configuration.

When RMS is running but the configuration being edited is not the same as the currently active one, the main menu is not restricted except that the *Configuration-Activate* menu option is not available.

3.3.2.1   Main configuration menu when RMS is not active

If RMS is not running anywhere, then the entire top level menu is presented without restrictions. Figure 10 shows the *Main configuration menu* window when RMS is inactive.

```
shasta1: Main configuration menu, current configuration: mydemo
No RMS active in the cluster
 1) HELP                            10) Configuration-Remove
 2) QUIT                            11) Configuration-Freeze
 3) Application-Create              12) Configuration-Thaw
 4) Application-Edit                13) Configuration-Edit-Global-Settings
 5) Application-Remove              14) Configuration-Consistency-Report
 6) Application-Clone               15) Configuration-ScriptExecution
 7) Configuration-Generate         16) RMS-CreateMachine
 8) Configuration-Activate         17) RMS-RemoveMachine
 9) Configuration-Copy
Choose an action:
```

Figure 10: Main configuration menu when RMS is not active

Menu items

The *Main configuration menu* can perform the following activities when RMS is not running anywhere in the cluster:

● *Application-Create*—Specifies which application to configure for high availability. In addition, this operation specifies all the relevant settings for the application so that it can run in a high-availability configuration monitored by RMS. Among the most important of these settings is the name of the application and the list of nodes on which the application may run.

The user application should be configured to run on multiple nodes for a high-availability configuration.

The wizard assists you by supplying menus with basic and non-basic attributes, assigns values to the attributes, and prompts you if an attribute is mandatory.

By choosing the appropriate turnkey wizard for an application, the wizard will then provide predefined elements, like scripts and detectors, for the application in question. These elements have been developed especially for the respective type of application.

The wizard will also carry out consistency checks at certain stages of the configuration procedure in order to prevent inconsistent applications from running in a high-availability configuration.

● *Application-Edit*—Modifies an existing application.

An existing application can be modified using this menu item. The following modes are available for editing an application:

– Turnkey mode (highly recommended)—Turnkey mode is the default mode. This mode is highly recommended because it simplifies complicated tasks like creating linkages between application and sub-applications.

– Non-turnkey mode —Do not use this mode for creating and editing a normal RMS configuration. Use it only when you are requested from field engineers.

● *Application-Remove*—Removes an existing application from the high-availability configuration.

● *Application-Clone*—Clones an application. This feature is provided for users who want to create a new application that differs only slightly from an existing one. To do this, clone an application and modify only the parts that are necessary to create a new one.

- *Configuration-Generate*—Performs the following:

    – Runs consistency checks on the configuration

    – Creates the RMS graph of the configuration and stores it in the *configname*.us file. The graph is a hierarchical description of objects that represent the nodes, applications, and resources used in the configuration.

    During the *Configuration-Generate* phase, the wizard indicates the progress with a series of dots on the screen. Each dot represents an application or resource that has been successfully generated.

    Normally, you would use *Configuration-Activate* (described below) to generate and distribute the configuration in one step. *Configuration-Generate* provides a way to generate and check a configuration without distributing it to the other nodes in the cluster. This may be useful for testing or debugging (see also the description for *Configuration-ScriptExecution* later in this list).

    > **i** *Configuration-Generate* is always available, whether RMS is running or not.

- *Configuration-Activate*—Generates and distributes a configuration.

    Selecting this item performs both the generation and distribution phases in one step. The generation phase is described above.

    The distribution phase prepares the cluster for RMS, ensuring that all the required data is put into place. The wizard copies the configuration data to every reachable node specified in the configuration and installs all necessary files. If one or more nodes is not available during the distribution phase, you can later use *Configuration-Push* (described in the next section) to update only those nodes.

    > **i**
    > - *Configuration-Activate* is not available if RMS is already running on one or more nodes.
    > - Do not execute the *Configuration-Activate* menu simultaneously on multiple nodes which constitute the cluster.

- *Configuration-Copy*—Produces a copy of an existing configuration. This is often used to make a backup before an existing, tested configuration is enhanced.

- *Configuration-Remove*—Removes (deletes) any existing configuration.

- *Configuration-Freeze*—Prevents further changes to a configuration. This marks the configuration as read-only so it can be viewed, but not modified.

| i | *Configuration-Freeze* is password protected: you will be prompted to create a password before the configuration is locked. |

● *Configuration-Thaw*—Releases the configuration from the frozen (read-only) state so it can be modified.

| i | *Configuration-Thaw* is password protected: you must enter the correct password before the configuration is unlocked. |

● *Configuration-Edit-Global-Settings*—Modifies settings that affect the entire configuration. This includes settings for the detectors and the operation mode of the hvw command. This item is also used to specify the alternate interconnects (AlternateIps) for the cluster.

● *Configuration-Consistency-Report*—Provides a consistency check that verifies whether an application is running within a high-availability configuration and has actually been created using the configuration data provided by the respective wizard.

The wizard compares the currently activated wizard checksum against the wizard database checksum. One checksum is called the *Live-Info*, the other is called the *BuildInfo*. If both checksums match for an application, it is certified that its running version conforms to what was configured by the wizard.

● *Configuration-ScriptExecution*—Allows administrators to run any script independent of RMS.

By selecting the resources configured for the application, the user can execute the scripts that are to bring the resources online or offline. To see the online scripts being executed, you can go through the resource list, which is displayed for this purpose, in ascending order. The return code indicates the proper functioning of the respective script.

● *RMS-CreateMachine*—Defines the list of machines which constitute the cluster. During the activation phase, the RMS configuration will be distributed to all the nodes in this list.

Applications managed by RMS must each be configured to run on one or more machines in this pool. Therefore, complete this step before creating any application.

● *RMS-RemoveMachine*—Removes machines from the list of cluster nodes.

3.3.2.2 Main configuration menu when RMS is running

Wizard Tools menus change dynamically according to whether or not RMS is running in the following locations:

- anywhere in the cluster

- on the local node

If RMS is running on any of the cluster machines, any operation which could potentially modify the currently active configuration is not allowed.

In particular, when RMS is running on the local node, the *Main configuration menu* changes as shown in Figure 11.

```
shasta1: Main configuration menu, current configuration: mydemo
RMS up on: shasta1RMS -- RMS down on: shasta2RMS
 1) HELP
 2) QUIT
 3) Application-View
 4) Configuration-Generate
 5) Configuration-Copy
 6) Configuration-Remove
 7) Configuration-Freeze
 8) Configuration-Edit-Global-Settings
 9) Configuration-Consistency-Report
10) Configuration-ScriptExecution
11) Configuration-Push
12) RMS-ViewMachine
Choose an action:
```

Figure 11: Main configuration menu when RMS is running

When RMS is running, the following entries either appear or change their behavior:

- *Application-View*—Views an existing application in read-only mode.

- *Configuration-Generate*—Same functionality as when RMS is not running.

- *Configuration-Copy*—Produces a copy of an existing configuration. This is often used to make a backup before an existing, tested configuration is enhanced.

    > **i** *Configuration-Copy* cannot overwrite the configuration that is currently running.

● *Configuration-Remove*—Removes (deletes) any existing configuration except the one that is currently running.

● *Configuration-Push*—Distributes a complete copy of the currently running configuration to a specific cluster node.

When a configuration is activated, some nodes may not be available. This menu item allows you to update individual cluster nodes that are brought up later, when RMS is already running. For example, if you changed the configuration while a node was down for maintenance, you could use *Configuration-Push* to update the node after it was restarted.

> **i** *Configuration-Push* is available only after the configuration has been activated.

● *RMS-ViewMachine*—Displays the list of nodes on which RMS is currently running.

3.3.3 Secondary menus

Each of the main menu items has a number of secondary menus. The secondary menus themselves can have sub-menus.

The *Creation: Application type selection menu* (Figure 12) is an example of a secondary menu. You see this menu after selecting *Application-Create* from the main menu.

```
Creation: Application type selection menu:
 1) HELP
 2) QUIT
 3) RETURN
 4) OPTIONS
 5) DEMO
 6) GENERIC
 7) LIVECACHE
 8) R3ANY
 9) R3CI
10) RTP
Application Type: 5
```

Figure 12: Application type selection

i	The list of available application types displayed in the menu depends on the packages installed on the local system. Some of the application types shown in this example may not be available in your market or for your platform.

This option allows you to select an application type to be assigned to the application in question. This is an important step in the configuration procedure since it invokes the specific application-type wizard to provide all the predefined elements (for example, scripts and detectors) that go with that application type.

The chapter "Configuration example" shows how to use some of the secondary menus. A more detailed description of these menus is given in the RMS Wizards documentation package.

3.3.4   Basic and non-basic settings

Basic and non-basic settings are designed to guide you safely through the configuration process, ensuring that all mandatory settings are configured.

Among the basic settings are the application name and the names of the nodes where it can run. For example, at the application type selection menu shown in the previous section, selecting *5) DEMO* produces the menu in Figure 13.

```
Consistency check ...
Yet to do: process the basic settings using Machines+Basics
Yet to do: choose a proper application name


Settings of turnkey wizard "DEMO"
1) HELP
2) NO-SAVE+EXIT
3) SAVE+EXIT
4) REMOVE+EXIT
5) ApplicationName=APP3
6) BeingControlled=no
7) Machines+Basics(-)
Choose the setting to process: 7
```

Figure 13: Menu leading to basic settings

If you select *7) Machines+Basics*, you can configure the basic settings using the menu in Figure 14.

```
Consistency check ...



Machines+Basics (app1:consistent)
 1) HELP                          14) (AutoStartUp=no)
 2) -                             15) (AutoSwitchOver=No)
 3) SAVE+EXIT                     16) (PreserveState=no)
 4) REMOVE+EXIT                   17) (PersistentFault=0)
 5) AdditionalMachine             18) (ShutdownPriority=)
 6) AdditionalConsole             19) (OnlinePriority=)
 7) Machines[0]=shasta1RMS        20) (StandbyTransitions=)
 8) (PreCheckScript=)             21) (LicenseToKill=no)
 9) (PreOnlineScript=)            22) (AutoBreak=yes)
10) (PostOnlineScript=)           23) (HaltFlag=no)
11) (PreOfflineScript=)           24) (PartialCluster=0)
12) (OfflineDoneScript=)          25) (ScriptTimeout=)
13) (FaultScript=)
Choose the setting to process:
```

Figure 14: Menu to configure basic settings

The menu displays the application's current attribute settings, some of which may be set automatically by the wizards. Attributes enclosed in parentheses are optional.

After you complete the configuration of the basic settings, the non-basic settings menu appears (Figure 15). Non-basic settings include specifications for resources such as file systems, IP adresses, disks, and so forth.

```
Consistency check ...
Yet to do: process at least one of the non-basic settings



Settings of turnkey wizard "DEMO"
 1) HELP                              11) RemoteFileSystems(-)
 2) -                                 12) IpAddresses(-)
 3) SAVE+EXIT                         13) RawDisks(-)
 4) -                                 14) RC-VolumeManagement(-)
 5) ApplicationName=APP1             15) VERITAS-VolumeManagement(-)
 6) Machines+Basics(app1)            16) EMC-RdfManagement(-)
 7) CommandLines(-)                   17) FibreCat-MirrorView(-)
 8) Controllers(-)                    18) Gds:Global-Disk-Services(-)
 9) DEMO(-)                           19) Gls:Global-Link-Services(-)
10) LocalFileSystems(-)
Choose the setting to process:
```

Figure 15: Menu to configure non-basic settings

> **i** The list of available subapplications displayed in the menu depends on the packages installed on the local system. Some of the subapplications shown in this example may not be available in your market or for your platform.

3.4   Activating a configuration

As described in section "General configuration procedure", activating a configuration is the third of the four fundamental steps required to set up a high-availability configuration. The activation phase comprises a number of tasks, among which are generation and distribution of a configuration.

> **i** You must stop RMS on all nodes in the cluster before you activate a configuration.

The starting point for the activation phase is the *Main configuration menu* (see Figure 16).

```
shasta1: Main configuration menu, current configuration: mydemo
No RMS active in the cluster
 1) HELP                            10) Configuration-Remove
 2) QUIT                            11) Configuration-Freeze
 3) Application-Create              12) Configuration-Thaw
 4) Application-Edit                13) Configuration-Edit-Global-Settings
 5) Application-Remove              14) Configuration-Consistency-Report
 6) Application-Clone               15) Configuration-ScriptExecution
 7) Configuration-Generate          16) RMS-CreateMachine
 8) Configuration-Activate          17) RMS-RemoveMachine
 9) Configuration-Copy
Choose an action: 8
```

Figure 16: Main configuration menu

► Select the *Configuration-Activate* item by entering the number 8.

The activation is performed by the wizard. No further input is required at this stage.

During the activation phase, the wizard executes a series of tasks and displays the status on the screen. The completion of a task is indicated by the word *done* or a similar expression (see Figure 17).

```
About to activate the configuration mydemo ...



Testing for RMS to be up somewhere in the cluster ... done.

Arranging sub applications topologically ... done.

Check for all applications being consistent ... done.

Running overall consistency check ... done.

Generating pseudo code [one dot per (sub) application]: ... done.

Generating RMS resources.......................... done


hvbuild using /usr/opt/reliant/build/wizard.d/mydemo/mydemo.us
About to distribute the new configuration data to hosts: shasta1RMS,shasta2RMS

The new configuration was distributed successfully.

About to put the new configuration in effect ... done.

The activation has finished successfully.
Hit CR to continue
```

Figure 17: Activating a configuration

Among the tasks carried out by *Configuration-Activate* are generation and distri-
bution of the configuration. The wizard performs a consistency check of the
graph created in the generation of the configuration before distributing the
configuration to all nodes specified in the configuration.

The test to see whether RMS is up on one of the nodes in the cluster is required
since activation cannot be performed if RMS is running. In this case, RMS would
need to be shut down first.

> **i** The *Configuration-Activate* process removes persistent status information
> on all affected nodes.

After the configuration has been activated successfully, you can return to the
Main configuration menu. From there, you can quit the configuration procedure.

► Press Enter to return to the *Main configuration menu* (see Figure 18).

```
shasta1: Main configuration menu, current configuration: mydemo
No RMS active in the cluster
 1) HELP                           10) Configuration-Remove
 2) QUIT                           11) Configuration-Freeze
 3) Application-Create             12) Configuration-Thaw
 4) Application-Edit               13) Configuration-Edit-Global-Settings
 5) Application-Remove             14) Configuration-Consistency-Report
 6) Application-Clone              15) Configuration-ScriptExecution
 7) Configuration-Generate         16) RMS-CreateMachine
 8) Configuration-Activate         17) RMS-RemoveMachine
 9) Configuration-Copy
Choose an action: 2
```

Figure 18: Quitting the Main configuration menu

► Select *QUIT* by entering the number 2.

This ends the activation phase of the configuration process. Usually, the next step is to start RMS to monitor the newly-configured application.

► Start RMS with the GUI or with the following command:

   hvcm −a

3.5 Configuration elements

This section discusses some basic elements that are part of a high-availability configuration. Most of them have been mentioned in previous sections. Additional details are provided here to assist you in understanding how they are used by the wizards.

> **i** Users do not have to deal with any of the items listed in this section directly. RMS Wizards manage all the basic elements for a high availability configuration. This section is provided only to help users better understand the configuration elements.

3.5.1 Scripts

Scripts are used in a high-availability configuration to perform several kinds of actions. Among the most important types of actions are the following:

● Bringing a resource to an `Online` state

● Bringing a resource to an `Offline` state

As an example of a script sending a resource `Offline`, you might think of a file system that has to be unmounted on a node where a fault occurs. An offline script would use the `umount` command to unmount the file system. Another script might use the `mount` command to mount it on a different node.

Besides such online and offline scripts, there are also pre-online and pre-offline scripts for preparing transition into the respective states, as well as a number of other scripts.

The RMS Wizards provide a complete set of scripts for several pre-defined application types such as R/3 or Oracle. If you assign your application to one of these standard types, you automatically take advantage of the built-in scripts.

> **i** The `hvexec` command executes scripts for a high-availability configuration monitored by RMS. For more details on the command `hvexec` please refer to the `primer.htm` document.

3.5.2　Detectors

Detectors are processes that have the task of monitoring resources. If there is a change in the state of a resource (for example, of a disk group) the detector in charge notifies the RMS base monitor. The base monitor may then decide to have a script executed as a reaction to this changed state.

Like the built-in scripts described in the previous section, the RMS Wizards provide built-in detectors for pre-defined application types. If you assign your application to one of these standard types, it automatically uses the built-in detectors.

3.5.3　RMS objects

A high-availability configuration can be seen as a set or group of objects with interdependencies. Any application or resource that is part of the configuration is then represented by one of the objects. The interdependences of objects can be displayed as a graph called the RMS graph.

These are the most important object types used in RMS configurations:

● `userApplication`—Represents an application to be configured for high-availability.

● `SysNode`—Represents a machine that is running as a node in a cluster.

● `gResource`—Represents a generic resource that is to be defined according to the needs of a customer application.

● `Controller`—Provides a dependency linkage so that a child application can act as a resource of the parent application.

In a typical configuration, one detector can be associated with all objects of the same type.

4 Configuration example

This chapter provides an example of the configuration process using the RMS Wizards. Two simple applications are configured for operation on a small cluster. The example includes the following steps:

● "Stopping RMS"

● "Creating a configuration"

● "Adding hosts to the cluster"

● "Creating an application"

● "Entering Machines+Basics settings"

● "Entering non-basic settings"

● "Specifying a display"

● "Activating the configuration"

● "Creating a second application"

● "Setting up a controlling application"

● "Specifying controlled applications"

● "Activating the configuration a second time"

● "Starting RMS"

An abbreviated version of this example appears in the *Software Release Guide PRIMECLUSTER* and the *PRIMECLUSTER Installation Guide* for your operating system.

> **i** To avoid network access problems, perform RMS configuration tasks as `root`, and ensure that CF is installed, properly configured, and running as described in the *Software Release Guide PRIMECLUSTER* and the *PRIMECLUSTER Installation Guide* for your operating system.

4.1 Stopping RMS

Before you create or edit a configuration, ensure that RMS is not active on any machine that would be affected by the changes. You can use the Cluster Admin GUI (see the section "Stopping RMS") or you can enter the following command to stop RMS on all nodes from any machine in the cluster:

```
# hvshut -a
```

4.2   Creating a configuration

► Enter the following command to generate the wizard menu for the configuration example, mydemo:

```
# hvw -n mydemo
```

This will create an RMS configuration file named mydemo.us in the /opt/SMAW/SMAWRrms/build/ directory. If you choose a different name and location, the combined length of the file name and path should not exceed 80 characters.

The *RMS configuration menu* appears, displaying the name of the configuration at the top of the menu (Figure 19).

```
shasta1: Main configuration menu, current configuration: mydemo
No RMS active in the cluster
 1) HELP                              10) Configuration-Remove
 2) QUIT                              11) Configuration-Freeze
 3) Application-Create                12) Configuration-Thaw
 4) Application-Edit                  13) Configuration-Edit-Global-Settings
 5) Application-Remove                14) Configuration-Consistency-Report
 6) Application-Clone                 15) Configuration-ScriptExecution
 7) Configuration-Generate           16) RMS-CreateMachine
 8) Configuration-Activate           17) RMS-RemoveMachine
 9) Configuration-Copy
Choose an action:
```

Figure 19: Main configuration menu

4.3   Adding hosts to the cluster

Before you configure an application, you must define the cluster so that it includes all hosts on which the application may run. The names of all possible RMS hosts should have already been added to the /etc/hosts file (see "Appendix—Site preparation").

Select the nodes to be included in the configuration. See "Appendix—Cluster planning worksheet" in the *Software Release Guide PRIMECLUSTER* and the *PRIMECLUSTER Installation Guide* for your operating system.

> **i** Set up the followings after adding hosts to the cluster of CF,Cluster Resource Management(CRM) and Shutdown Facility(SF). For more details on adding hosts to the cluster, refer to "PRIMECLUSTER Cluster Foundation (CF) Configuration and Administration Guide" for your operating system.

This example assumes /etc/hosts contains the following entries, which follow the RMS naming convention:

```
# host names for RMS
192.168.10.83    shasta1RMS
192.168.10.84    shasta2RMS
```

> **i** By default, RMS host names are of the form *<hostname>*RMS to follow the RMS naming convention. To override the default RMS name for a machine, edit that host's hvenv.local file and set the RELIANT_HOSTNAME variable to the desired name. The contents of that host's RELIANT_HOSTNAME variable must match the corresponding /etc/hosts entry on every host in the cluster. This must be done before you start the Wizard Tools (hvw). If RMS is running, you must also restart RMS.

In this step, you will add the RMS hosts to the cluster.

► At the *Main configuration menu*, enter the number 16. The *Add hosts to a cluster* menu appears (Figure 20).

```
Creation: Add hosts to a cluster:
Current set:
1) HELP
2) QUIT
3) RETURN
4) FREECHOICE
5) ALL-CF-HOSTS
6) shasta1RMS
7) shasta2RMS
Choose the host to add: 7
```

Figure 20: Add hosts to a cluster menu

This menu displays the current set of nodes and lists the machines that can be selected. If you select *5) ALL-CF-HOSTS*, the RMS Wizards add all nodes in /etc/cip.cf to this configuration. Otherwise, you can add hosts individually from the displayed list.

▶ Select *shasta1RMS* by entering the number 6. Select *shasta2RMS* by entering the number 7 (see Figure 20).

At this screen, you can also choose *4) FREECHOICE*, which will allow you to enter host names that are not listed in the menu.

▶ After all host names have been added, use *3) RETURN* to return to the *Main configuration menu*.

To remove a node, select *17) RMS-RemoveMachine* from the *Main configuration menu*. The *Remove hosts from a cluster* menu appears (Figure 21).

```
Removal: Remove hosts from a cluster:
Current set: shasta1RMS shasta2RMS
1) HELP
2) QUIT
3) RETURN
4) ALL
5) shasta1RMS
6) shasta2RMS
Choose the host to remove:
```

Figure 21: Remove hosts from a cluster menu

This menu lists all nodes currently in the cluster. Machines can be removed by selecting them individually or by selecting *4) ALL* from the menu. In either case, machines being used by one or more applications cannot be removed.

4.4   Creating an application

After you have defined the set of hosts that form the cluster, you can configure an application that will run on those hosts. In this step, we will first create the application using the DEMO turnkey wizard. Begin at the *Main configuration menu* (Figure 22).

```
shasta1: Main configuration menu, current configuration: mydemo
No RMS active in the cluster
 1) HELP                           10) Configuration-Remove
 2) QUIT                           11) Configuration-Freeze
 3) Application-Create             12) Configuration-Thaw
 4) Application-Edit               13) Configuration-Edit-Global-Settings
 5) Application-Remove             14) Configuration-Consistency-Report
 6) Application-Clone              15) Configuration-ScriptExecution
 7) Configuration-Generate         16) RMS-CreateMachine
 8) Configuration-Activate         17) RMS-RemoveMachine
 9) Configuration-Copy
Choose an action: 3
```

Figure 22: Main configuration menu

► Select *Application-Create* by entering the number 3. The *Application type selection menu* appears (Figure 23).

```
Creation: Application type selection menu:
 1) HELP
 2) QUIT
 3) RETURN
 4) OPTIONS
 5) DEMO
 6) GENERIC
 7) LIVECACHE
 8) R3ANY
 9) R3CI
10) RTP
Application Type: 5
```

Figure 23: Application type selection menu

| i | The list of available application types displayed in the menu depends on the packages installed on the local system. Some of the application types shown in this example may not be available in your market or for your platform. |

This example uses the *DEMO* application type, which has been designed to familiarize the user with the configuration process and is intended for demonstration purposes only: other than a few user-specified attributes, everything is preset and ready to run. To configure a real-world application, you would instead select the *GENERIC* application type, as described in the section "Creating a second application".

▶ Select the *DEMO* application type by entering the number 5.

You have now assigned the *DEMO* application type to your application. This means the *DEMO* turnkey wizard will provide the application with scripts and detectors that were developed for this application type.

There are, however, more parameters to specify before this application can run. One of them might be the application name; you can assign a name of your choice to any application that you configure for RMS. In this case, there is no need to specify an application name, as the *DEMO* wizard provides *APP1* as a default here.

APP1 is a simple application, developed specifically for this example, that generates an animated graphical figure on an X-window display. It will be used demonstrate how an application can be started, stopped, or switched, and how RMS performs failover when the application process is killed on the initial node.

After performing a consistency check, the wizard informs you what to do next (see Figure 24).

```
Consistency check ...
Yet to do: process the basic settings using Machines+Basics
Yet to do: choose a proper application name


Settings of turnkey wizard "DEMO"
1) HELP
2) NO-SAVE+EXIT
3) SAVE+EXIT
4) REMOVE+EXIT
5) ApplicationName=APP1
6) BeingControlled=no
7) Machines+Basics(-)
Choose the setting to process: 7
```

Figure 24: Prompting for further actions

At each step, the wizard checks the consistency of the application being configured. Only consistent applications are allowed to be part of the high-availability configuration.

If you want to specify a different application name, you could do it here by selecting *5) ApplicationName*. However, because we are using the default of APP1, the *Yet to do* message will disappear after you select *7) Machine+Basics*.

4.5   Entering Machines+Basics settings

▶   Select *Machines+Basics* by entering the number 7. The *Machines+Basics*
    menu appears (Figure 25).

```
Consistency check ...



Machines+Basics (app1:consistent)
 1) HELP                             14) (AutoStartUp=no)
 2) -                                15) (AutoSwitchOver=No)
 3) SAVE+EXIT                        16) (PreserveState=no)
 4) REMOVE+EXIT                      17) (PersistentFault=0)
 5) AdditionalMachine                18) (ShutdownPriority=)
 6) AdditionalConsole                19) (OnlinePriority=)
 7) Machines[0]=shasta1RMS           20) (StandbyTransitions=)
 8) (PreCheckScript=)                21) (LicenseToKill=no)
 9) (PreOnlineScript=)               22) (AutoBreak=yes)
10) (PostOnlineScript=)              23) (HaltFlag=no)
11) (PreOfflineScript=)              24) (PartialCluster=0)
12) (OfflineDoneScript=)             25) (ScriptTimeout=)
13) (FaultScript=)
Choose the setting to process: 5
```

Figure 25: Consistency check and Machines+Basics menu

At the top of the menu, the wizard shows you the result of the latest consistency
check. The application named *APP1*, which was indicated on the previous
screen, has proven to be consistent.

The *Machines[0]* menu item indicates the node where your application will first
attempt to come online. In this case, it is shasta1RMS.

> **i**   The RMS Wizards retrieve the default settings for *Machines[0]* from the
>   local node defined in RELIANT_HOSTNAME.

Subsequent *Machines[ ]* items, if any, indicate the list of failover nodes. If the
initial node fails, RMS will attempt to switch the application to a failover node,
trying each one in the list according to the index order.

At this point, only the initial node appears in the menu, so configure a failover
node for your application as follows:

► Select *AdditionalMachine* by entering the number 5. A menu containing the current list of available nodes appears (Figure 26).

```
1) HELP
2) RETURN
3) shasta1RMS
4) shasta2RMS
Choose a machine for this application: 4
```

Figure 26: List of nodes for failover procedure

> **i** The Wizards retrieve the default list of nodes from the CIP configuration file, /etc/cip.cf.

Since our application is presently configured for shasta1RMS, we will select shasta2RMS as the additional node:

► Select *shasta2RMS* by entering the number 4.

In the menu that follows (Figure 27) you will see your selection confirmed. *shasta2RMS* now appears under *Machines[1]* as the additional node. If there is a failure on shasta1RMS, your application is configured to switch over to shasta2RMS.

```
Consistency check ...



Machines+Basics (app1:consistent)
 1) HELP                         14) (FaultScript=)
 2) -                            15) (AutoStartUp=no)
 3) SAVE+EXIT                    16) (AutoSwitchOver=No)
 4) REMOVE+EXIT                  17) (PreserveState=no)
 5) AdditionalMachine            18) (PersistentFault=0)
 6) AdditionalConsole            19) (ShutdownPriority=)
 7) Machines[0]=shasta1RMS       20) (OnlinePriority=)
 8) Machines[1]=shasta2RMS       21) (StandbyTransitions=)
 9) (PreCheckScript=)            22) (LicenseToKill=no)
10) (PreOnlineScript=)           23) (AutoBreak=yes)
11) (PostOnlineScript=)          24) (HaltFlag=no)
12) (PreOfflineScript=)          25) (PartialCluster=0)
13) (OfflineDoneScript=)         26) (ScriptTimeout=)
Choose the setting to process: 16
```

Figure 27: Machines+Basics menu for additional nodes

At this point, the default value of *No* is specified for *16) AutoSwitchOver*. This means that to actually switch your application over, manual action would be required.

► To have the switchover procedure carried out automatically, select *16) AutoSwitchOver*

The *Set flags for AutoSwitchOver* menu opens (Figure 28).

```
Set flags for AutoSwitchOver:  Currently set: NO (N)
1) HELP
2) -
3) SAVE+RETURN
4) DEFAULT
5) NO(N)
6) HOSTFAILURE(H)
7) RESOURCEFAILURE(R)
8) SHUTDOWN(S)
Choose one of the flags: 6
```

Figure 28: AutoSwitchOver mode

▶ Set a flag by entering the number 6 for *HOSTFAILURE*. This means that RMS switches an application to another node automatically in the case of a node failure.

The first line of the updated menu indicates the *HOSTFAILURE* flag is now set (Figure 29).

```
Set flags for AutoSwitchOver:  Currently set: HOSTFAILURE (H)
1) HELP
2) –
3) SAVE+RETURN
4) DEFAULT
5) NO(N)
6) NOT:HOSTFAILURE(H)
7) RESOURCEFAILURE(R)
8) SHUTDOWN(S)
Choose one of the flags:
```

Figure 29: Setting flags for AutoSwitchOver mode

Continue with the following operations from the same menu:

▶ Enter the number 7 for *RESOURCEFAILURE*. This means that RMS switches an application to another node automatically in the case of a resource failure.

▶ Enter the number 3 for *SAVE+RETURN*.

You will be returned to the *Machines+Basics* menu (Figure 30). Note that item 16 now displays the *AutoSwitchOver* flags you just set.

```
Consistency check ...



Machines+Basics (app1:consistent)
 1) HELP
 2) -
 3) SAVE+EXIT
 4) REMOVE+EXIT
 5) AdditionalMachine
 6) AdditionalConsole
 7) Machines[0]=shasta1RMS
 8) Machines[1]=shasta2RMS
 9) (PreCheckScript=)
10) (PreOnlineScript=)
11) (PostOnlineScript=)
12) (PreOfflineScript=)
13) (OfflineDoneScript=)
14) (FaultScript=)
15) (AutoStartUp=no)
16) (AutoSwitchOver=HostFailure|ResourceFailure)
17) (PreserveState=no)
18) (PersistentFault=0)
19) (ShutdownPriority=)
20) (OnlinePriority=)
21) (StandbyTransitions=)
22) (LicenseToKill=no)
23) (AutoBreak=yes)
24) (HaltFlag=no)
25) (PartialCluster=0)
26) (ScriptTimeout=)
Choose the setting to process: 3
```

Figure 30: Saving settings

Save your settings now to complete the *Application-Create* process.

▶ Select *SAVE+EXIT* by entering the number 3.

4.6   Entering non-basic settings

The DEMO turnkey wizard performs another consistency check before
returning you to the wizard settings menu (Figure 31).

```
Consistency check ...
Yet to do: process at least one of the non-basic settings



Settings of turnkey wizard "DEMO"
 1) HELP                              11) RemoteFileSystems(-)
 2) -                                 12) IpAddresses(-)
 3) SAVE+EXIT                         13) RawDisks(-)
 4) -                                 14) RC-VolumeManagement(-)
 5) ApplicationName=APP1              15) VERITAS-VolumeManagement(-)
 6) Machines+Basics(app1)            16) EMC-RdfManagement(-)
 7) CommandLines(-)                   17) FibreCat-MirrorView(-)
 8) Controllers(-)                    18) Gds:Global-Disk-Services(-)
 9) DEMO(-)                           19) Gls:Global-Link-Services(-)
10) LocalFileSystems(-)
Choose the setting to process: 9
```

Figure 31: Non-basic settings

| i | The list of available subapplications displayed in the menu depends on the packages installed on the local system. Some of the subapplications shown in this example may not be available in your market or for your platform. |

The menu header indicates there is at least one more setting to specify, but it is
not a basic setting.

As described earlier, this application creates an animated graphical picture on
an X-window display. Therefore, a display setting for the DEMO wizard must be
added to the basic settings you have already entered.

► Select *DEMO* by entering the number 9. The *CommandLines* menu appears
   (Figure 32).

```
Consistency check ...
Yet to do: set a display


CommandLines (Dem_APP1:not yet consistent)
 1) HELP
 2) -
 3) SAVE+EXIT
 4) REMOVE+EXIT
 5) Display=
 6) StartCommands[0]='hvexec~-F~demo~-c'
 7) StopCommands[0]='hvexec~-F~demo~-u'
 8) CheckCommands[0]=hvdet_demo
 9) (Timeout=300)
10) (AutoRecover=no)
11) (MonitorOnly=no)
Choose the setting to process: 5
```

Figure 32: Prompting for display specification

The menu header indicates that a display still needs to be specified, and the status line tells you that *APP1* is not yet consistent; that is, *APP1* could not yet run with the present *mydemo* configuration.

Because the DEMO wizard has been customized for demonstration purposes, some of the items in the menu have been predefined. Items in the menu body indicate the scripts provided by the wizard for starting, stopping, and checking: see the lines beginning with *6) StartCommands[0]=*, *7) StopCommands[0]=*, and *8) CheckCommands[0]=*.

| **i** | For technical reasons, spaces are displayed as tildes (~) within the wizard menu commands. The actual commands do not have tildes. |

4.7   Specifying a display

Specify the display within the *CommandLines* menu as follows:

► Select *Display* by entering the number 5. A list of display options appears
  (Figure 33).

```
 1) HELP
 2) RETURN
 3) FREECHOICE
 4) shasta3ADM
 5) shasta1ADM
 6) shasta2ADM
 7) shastaRCA
 8) shastaSCON
 9) shasta1
10) shasta2
11) shasta1RMS
12) shasta2RMS
Choose a display for this application: 3
               >> 172.25.220.27
```

Figure 33: List of display options

You can choose from the list of detected hosts (all hosts in /etc/hosts), or you
can select *3) FREECHOICE* to specify an arbitrary host with a suitable display.

► Select *FREECHOICE* by entering the number 3.

At the >> prompt, enter the host name or IP address for the X-window display.
In this example, we use the IP address 172.25.220.27, but you should enter
an address in your LAN.

Completing the *FREECHOICE* step initiates another consistency check
(Figure 34).

```
Consistency check ...



CommandLines (Dem_APP1:consistent)
 1) HELP
 2) -
 3) SAVE+EXIT
 4) REMOVE+EXIT
 5) Display=172.25.220.27
 6) StartCommands[0]='hvexec~-F~demo~-c~~172.25.220.27'
 7) StopCommands[0]='hvexec~-F~demo~-u~~172.25.220.27'
 8) CheckCommands[0]=hvdet_demo
 9) (Timeout=300)
10) (AutoRecover=no)
11) (MonitorOnly=no)
Choose the setting to process: 3
```

Figure 34: Successful consistency check for APP1

The consistency check is successful: you can now use RMS to run *APP1* with the *mydemo* configuration.

Note that the wizard updated the display information for the scripts in items *6) StartCommands[0]* and *7) StopCommands[0]*.

This completes the specification of the non-basic settings. You can now save the non-basic settings and exit this part of the configuration procedure.

► From the *CommandLines* menu (Figure 34), select *SAVE+EXIT* by entering the number 3.

This will take you back to the *Settings of turnkey wizard "DEMO"* menu (Figure 35).

```
Consistency check ...



Settings of turnkey wizard "DEMO"
 1) HELP                              11) RemoteFileSystems(-)
 2) -                                 12) IpAddresses(-)
 3) SAVE+EXIT                         13) RawDisks(-)
 4) -                                 14) RC-VolumeManagement(-)
 5) ApplicationName=APP1              15) VERITAS-VolumeManagement(-)
 6) Machines+Basics(app1)            16) EMC-RdfManagement(-)
 7) CommandLines(-)                   17) FibreCat-MirrorView(-)
 8) Controllers(-)                    18) Gds:Global-Disk-Services(-)
 9) DEMO(Dem_APP1)                    19) Gls:Global-Link-Services(-)
10) LocalFileSystems(-)
Choose the setting to process: 3
```

Figure 35: Turnkey wizard DEMO

By specifying the basic and non-basic settings for your application and achieving a consistent result, you have successfully finished the *Application-Create* part of the configuration procedure.

► Select *SAVE+EXIT* by entering the number 3. This will take you back to the *RMS configuration menu*.

4.8   Activating the configuration

As described in the section "General configuration procedure", activating a configuration is the third of the four fundamental steps required to set up a high-availability configuration.

You must stop RMS before activating a configuration. In this example, we stopped RMS before creating the configuration.

The starting point for the activation phase is the *Main configuration menu* (Figure 36).

```
shasta1: Main configuration menu, current configuration: mydemo
No RMS active in the cluster
 1) HELP                            10) Configuration-Remove
 2) QUIT                            11) Configuration-Freeze
 3) Application-Create              12) Configuration-Thaw
 4) Application-Edit                13) Configuration-Edit-Global-Settings
 5) Application-Remove              14) Configuration-Consistency-Report
 6) Application-Clone               15) Configuration-ScriptExecution
 7) Configuration-Generate          16) RMS-CreateMachine
 8) Configuration-Activate          17) RMS-RemoveMachine
 9) Configuration-Copy
Choose an action: 8
```

Figure 36: Main configuration menu

▶   Select *Configuration-Activate* by entering the number 8.

No further input is required at this stage. As the Wizard completes each task in the activation phase, it displays status information as described in the section "Activating a configuration". You will be prompted to continue at the end of the process (see Figure 37).

```
The new configuration was distributed successfully.


About to put the new configuration in effect ... done.


The activation has finished successfully.
Hit CR to continue
```

Figure 37: Successful configuration activation

► Press the ⌈Enter⌋ or ⌈Return⌋ key to return to the *Main configuration menu* (Figure 38).

```
shasta1: Main configuration menu, current configuration: mydemo
No RMS active in the cluster
 1) HELP                           10) Configuration-Remove
 2) QUIT                           11) Configuration-Freeze
 3) Application-Create             12) Configuration-Thaw
 4) Application-Edit               13) Configuration-Edit-Global-Settings
 5) Application-Remove             14) Configuration-Consistency-Report
 6) Application-Clone              15) Configuration-ScriptExecution
 7) Configuration-Generate         16) RMS-CreateMachine
 8) Configuration-Activate         17) RMS-RemoveMachine
 9) Configuration-Copy
Choose an action: 2
```

Figure 38: Quitting the Main configuration menu

► Select *QUIT* by entering the number 2.

This ends the activation phase of the configuration process. At this point, RMS may be started to monitor the newly-configured application.

4.9   Creating a second application

In this section, the *mydemo* configuration is expanded by adding a second application. This example application differs from the first because duplicate configuration procedures are skipped to simplify the example. However, in other parts of the procedure, new features add to the complexity of the *mydemo* configuration.

The second application differs from the first as follows:

● The application uses a new application type, *GENERIC*, instead of *DEMO*. We will use the name APP2 for the second application.

● APP2 will control the first application (APP1). Therefore, APP2 must be configured with a controller sub-application.

Resume the configuration procedure as follows:

▶ Stop RMS if it is running.

▶ Return to the *Main configuration menu* with the following command:

    # **hvw -n mydemo**

The *Main configuration menu* opens (see Figure 39).

```
shasta1: Main configuration menu, current configuration: mydemo
No RMS active in the cluster
 1) HELP                          10) Configuration-Remove
 2) QUIT                          11) Configuration-Freeze
 3) Application-Create            12) Configuration-Thaw
 4) Application-Edit              13) Configuration-Edit-Global-Settings
 5) Application-Remove            14) Configuration-Consistency-Report
 6) Application-Clone             15) Configuration-ScriptExecution
 7) Configuration-Generate        16) RMS-CreateMachine
 8) Configuration-Activate        17) RMS-RemoveMachine
 9) Configuration-Copy
Choose an action: 3
```

Figure 39: Starting again with the Main configuration menu

You can add more machines to the cluster at this point, provided the required site preparation steps have been completed.

▶ To add machines, select *RMS-CreateMachine* by entering the number 16.
Follow the procedure described earlier and then return to the *Main
configuration menu* when finished.

▶ From the *Main configuration menu*, select *Application-Create* by entering the
number 3.

The *Application type selection* menu opens (see Figure 40).

```
Creation: Application type selection menu:
 1) HELP
 2) QUIT
 3) RETURN
 4) OPTIONS
 5) DEMO
 6) GENERIC
 7) LIVECACHE
 8) R3ANY
 9) R3CI
10) RTP
Application Type: 6
```

Figure 40: Application type selection menu

This time, assign the *GENERIC* application type to the application. This means
that the GENERIC turnkey wizard will be in charge of the configuration
procedure.

▶ Select the *GENERIC* application type by entering the number 6.

After the consistency check, you are prompted to configure the basic settings.
APP2 is the default value for the application name.

┌───┐
│ **i** │ If you want to change the name, select *5) ApplicationName* (see
└───┘ Figure 41).

```
Consistency check ...
Yet to do: process the basic settings using Machines+Basics
Yet to do: choose a proper application name


Settings of turnkey wizard "GENERIC"
1) HELP
2) NO-SAVE+EXIT
3) SAVE+EXIT
4) REMOVE+EXIT
5) ApplicationName=APP2
6) BeingControlled=no
7) Machines+Basics(-)
Choose the setting to process: 7
```

Figure 41: Prompting for further specification

▶ Select *Machines+Basics* by entering the number 7.

The consistency of APP2 is checked, and the result is positive. When the *Machines+Basics* menu appears, it shows that APP2 is initially configured to run on shasta1RMS (see item *7) Machines[0]* in Figure 42).

```
Consistency check ...

Machines+Basics (app2:consistent)
 1) HELP                        14) (AutoStartUp=no)
 2) -                           15) (AutoSwitchOver=No)
 3) SAVE+EXIT                   16) (PreserveState=no)
 4) REMOVE+EXIT                 17) (PersistentFault=0)
 5) AdditionalMachine           18) (ShutdownPriority=)
 6) AdditionalConsole           19) (OnlinePriority=)
 7) Machines[0]=shasta1RMS      20) (StandbyTransitions=)
 8) (PreCheckScript=)           21) (LicenseToKill=no)
 9) (PreOnlineScript=)          22) (AutoBreak=yes)
10) (PostOnlineScript=)         23) (HaltFlag=no)
11) (PreOfflineScript=)         24) (PartialCluster=0)
12) (OfflineDoneScript=)        25) (ScriptTimeout=)
13) (FaultScript=)
Choose the setting to process: 5
```

Figure 42: Machines+Basics menu

► Select *AdditionalMachine* by entering the number 5. A menu appears with the list of available machines (Figure 43).

```
1) HELP
2) RETURN
3) shasta1RMS
4) shasta2RMS
Choose a machine for this application: 4
```

Figure 43: List of nodes for failover procedure

As with the former application, the additional machine to be specified for the failover procedure is shasta2RMS.

► Select *shasta2RMS* by entering the number 4.

In the screen that follows you see your selection confirmed—menu item *8) Machines[1]* now displays shasta2RMS as the additional machine (Figure 44).

```
Consistency check ...



Machines+Basics (app2:consistent)
 1) HELP                            14) (FaultScript=)
 2) -                               15) (AutoStartUp=no)
 3) SAVE+EXIT                       16) (AutoSwitchOver=No)
 4) REMOVE+EXIT                     17) (PreserveState=no)
 5) AdditionalMachine               18) (PersistentFault=0)
 6) AdditionalConsole               19) (ShutdownPriority=)
 7) Machines[0]=shasta1RMS          20) (OnlinePriority=)
 8) Machines[1]=shasta2RMS          21) (StandbyTransitions=)
 9) (PreCheckScript=)               22) (LicenseToKill=no)
10) (PreOnlineScript=)              23) (AutoBreak=yes)
11) (PostOnlineScript=)             24) (HaltFlag=no)
12) (PreOfflineScript=)             25) (PartialCluster=0)
13) (OfflineDoneScript=)            26) (ScriptTimeout=)
Choose the setting to process: 3
```

Figure 44: Machines+Basics menu

Note that item 16 indicates `AutoSwitchOver=No`, so `APP2` will not be switched automatically to `shasta2RMS` if `shasta1RMS` fails: it must be switched manually with the GUI or CLI. To enable automatic switchover, select item 16.

Save your settings and exit this part of the configuration procedure:

▶ Select *SAVE+EXIT* by entering the number 3. This takes you to the non-basic settings menu.

4.10 Setting up a controlling application

The basic settings have been specified. However, we still need to set up `APP2` to control `APP1`. This will involve the following two steps, available in the non-basic settings:

● Create a controller object for `APP2`.

● Specify `APP1` as the application to be controlled.

The previous step has taken you to the non-basic settings menu (Figure 45).

```
Consistency check ...
Yet to do: process at least one of the non-basic settings


Settings of turnkey wizard "GENERIC"
 1) HELP                           10) RemoteFileSystems(-)
 2) -                              11) IpAddresses(-)
 3) SAVE+EXIT                      12) RawDisks(-)
 4) -                              13) RC-VolumeManagement(-)
 5) ApplicationName=APP2           14) VERITAS-VolumeManagement(-)
 6) Machines+Basics(app2)          15) EMC-RdfManagement(-)
 7) CommandLines(-)                16) FibreCat-MirrorView(-)
 8) Controllers(-)                 17) Gds:Global-Disk-Services(-)
 9) LocalFileSystems(-)            18) Gls:Global-Link-Services(-)
Choose the setting to process: 8
```

Figure 45: Non-basic settings

▶ Select *Controllers* by entering the number 8.

This creates a controller object for `APP2` and presents a menu that lets you specify the controller settings (Figure 46).

```
Consistency check ...
Yet to do: assign at least one application to control
Yet to do: configure at least one controlled application without the M flag

Settings of application type "Controller" (not yet consistent)
1) HELP
2) -
3) SAVE+EXIT
4) REMOVE+EXIT
5) ControlPolicy=FOLLOW
6) AdditionalAppToControl
7) (InParallel=)
8) (FaultScript=)
Choose the setting to process: 6
```

Figure 46: Assigning a controller

Note that item 5 allows you to change the controller type. This example will use
the default follow controller.

4.11   Specifying controlled applications

Once you specify a controller, the wizard needs to know which application to
control.

| i | When an application becomes the child of a controller, the Wizards adjust some of its attributes automatically. In the case of a Follow controller, the child's *Machines[ ]* entries are overridden by the settings of the parent. |

► Select *AdditionalAppToControl* by entering the number 6.

The menu that appears offers you a list from which to choose an application
(Figure 47).

```
1) HELP
2) RETURN
3) FREECHOICE
4) app1
Choose an application to control: 4
```

Figure 47: List of applications to be chosen as controlled applications

The controlled application is APP1, while APP2 is the controlling application. Choose the application to be controlled as follows:

► Select *APP1* by entering the number 4. The controller flags menu appears (Figure 48).

```
Set flags for (sub) application: app1
Currently set: AUTORECOVER,TIMEOUT (AT180)
1) HELP
2) -
3) SAVE+RETURN
4) DEFAULT
5) MONITORONLY(M)
6) NOT:AUTORECOVER(A)
7) TIMEOUT(T)
Choose one of the flags:
```

Figure 48: Menu for setting controller flags

There are a number of flags that can be set for a controlled application. In this example, the *A* (*AUTORECOVER*) flag has been set. The *A* flag means If the controlled application becomes offline, the controlling application tries to restart it. The *AUTORECOVER* menu item is now in the opposite state; that is, ready to be toggled to *NOT*.

The *T* (*TIMEOUT*) flag limits the amount of time tolerated while bringing the controlled application online. In this example, we will reduce the timeout period to 150 seconds.

► Change the timeout period by entering 7.

► In the menu that appears (Figure 49), select *FREECHOICE* by entering the number 3.

```
1) HELP
2) RETURN
3) FREECHOICE
4) 180
Set an appropriate timeout: 3
        >> 150
```

Figure 49: Changing controller timeout period

► At the >> prompt, enter 150 for the timeout period.

► Press ⌷Enter⌷ or ⌷Return⌷ to return to the menu for controller flags (Figure 50).

```
Set flags for (sub) application: app1
Currently set: AUTORECOVER,TIMEOUT (AT150)
1) HELP
2) -
3) SAVE+RETURN
4) DEFAULT
5) MONITORONLY(M)
6) NOT:AUTORECOVER(A)
7) TIMEOUT(T)
Choose one of the flags: 3
```

Figure 50: Saving flags for controller

After completing the settings, save them and return to the *Controllers* menu as follows:

► Select *SAVE+RETURN* by entering the number 3.

The *Controllers* menu shows that the controller settings are now consistent (Figure 51).

```
Consistency check ...



Settings of application type "Controller" (consistent)
1) HELP
2) -
3) SAVE+EXIT
4) REMOVE+EXIT
5) ControlPolicy=FOLLOW
6) AdditionalAppToControl
7) Controllers[0]=AT150:app1
8) (InParallel=)
9) (FaultScript=)
Choose the setting to process: 3
```

Figure 51: Indication of flags set for controller

Note that your settings are confirmed on item *7) Controllers[0]*: the *A* and *T* flags have been set for APP1.

▶ Select *SAVE+EXIT* by entering the number 3.

This takes you back to the *GENERIC* menu (Figure 52).

```
Consistency check ...



Settings of turnkey wizard "GENERIC"
 1) HELP                          10) RemoteFileSystems(-)
 2) -                             11) IpAddresses(-)
 3) SAVE+EXIT                     12) RawDisks(-)
 4) -                             13) RC-VolumeManagement(-)
 5) ApplicationName=APP2          14) VERITAS-VolumeManagement(-)
 6) Machines+Basics(app2)         15) EMC-RdfManagement(-)
 7) CommandLines(-)               16) FibreCat-MirrorView(-)
 8) Controllers(Ctl_APP2)         17) Gds:Global-Disk-Services(-)
 9) LocalFileSystems(-)           18) Gls:Global-Link-Services(-)
Choose the setting to process: 3
```

Figure 52: Menu with settings for GENERIC turnkey wizard

In the *GENERIC* menu, item *8 Controllers* now displays a controller assigned to APP2.

▶ Select *SAVE+EXIT* by entering the number 3. This takes you back to the *Main configuration menu* (Figure 53).

```
shasta1: Main configuration menu, current configuration: mydemo
No RMS active in the cluster
 1) HELP                          10) Configuration-Remove
 2) QUIT                          11) Configuration-Freeze
 3) Application-Create            12) Configuration-Thaw
 4) Application-Edit              13) Configuration-Edit-Global-Settings
 5) Application-Remove            14) Configuration-Consistency-Report
 6) Application-Clone             15) Configuration-ScriptExecution
 7) Configuration-Generate        16) RMS-CreateMachine
 8) Configuration-Activate        17) RMS-RemoveMachine
 9) Configuration-Copy
Choose an action:
```

Figure 53: Main configuration menu

This completes the creation of the second application.

4.12   Activating the configuration a second time

After returning to the *Main configuration menu*, you must activate the *mydemo* configuration for the second time. This has to be done because you have modified the configuration by adding another application.

RMS cannot be running while you activate a configuration. In this example, we stopped RMS before creating the second application.

To activate the configuration, begin at the *Main configuration menu* (Figure 54).

```
shasta1: Main configuration menu, current configuration: mydemo
No RMS active in the cluster
 1) HELP                            10) Configuration-Remove
 2) QUIT                            11) Configuration-Freeze
 3) Application-Create              12) Configuration-Thaw
 4) Application-Edit                13) Configuration-Edit-Global-Settings
 5) Application-Remove              14) Configuration-Consistency-Report
 6) Application-Clone               15) Configuration-ScriptExecution
 7) Configuration-Generate          16) RMS-CreateMachine
 8) Configuration-Activate          17) RMS-RemoveMachine
 9) Configuration-Copy
Choose an action: 8
```

Figure 54: Main configuration menu

► Select *Configuration-Activate* by entering the number 8.

No further input is required at this stage. As the Wizard completes each task in the activation phase, it displays status information as described in the section "Activating a configuration". You will be prompted to continue at the end of the process (Figure 55).

```
The new configuration was distributed successfully.

About to put the new configuration in effect ... done.

The activation has finished successfully.
Hit CR to continue
```

Figure 55: Activating the configuration for the second time

► Press the ⌐Enter⌐ or ⌐Return⌐ key to return to the *Main configuration menu* (Figure 56).

```
shasta1: Main configuration menu, current configuration: mydemo
No RMS active in the cluster
 1) HELP                          10) Configuration-Remove
 2) QUIT                          11) Configuration-Freeze
 3) Application-Create            12) Configuration-Thaw
 4) Application-Edit              13) Configuration-Edit-Global-Settings
 5) Application-Remove            14) Configuration-Consistency-Report
 6) Application-Clone             15) Configuration-ScriptExecution
 7) Configuration-Generate        16) RMS-CreateMachine
 8) Configuration-Activate        17) RMS-RemoveMachine
 9) Configuration-Copy
Choose an action: 2
```

Figure 56: Return to Main configuration menu

► Select *QUIT* by entering the number 2.

This ends the activation phase of the configuration process.

4.13 Starting RMS

 At this point, you are ready to start RMS on all nodes in the cluster to monitor both applications. You can use the Cluster Admin GUI (see the section "Starting RMS") or you can enter the following command from any machine in the cluster:

hvcm -a mydemo

Note that you do not have to specify "mydemo" on the hvcm command line if it is the configuration that was most recently activated.

This ends the configuration example.

5 Using the Cluster Admin GUI

This chapter describes PRIMECLUSTER administration using the Cluster Admin graphical user interface (GUI). In addition, some command-line interface (CLI) commands are discussed.

Chapter contents:

● "Overview"

● "Starting the Cluster Admin GUI"

● "Monitoring RMS with Cluster Admin"

5.1 Overview

RMS administration can be done by means of the Cluster Admin GUI or by the CLI. The procedures in this chapter focus on the Cluster Admin GUI.

Most of the GUI examples in this chapter show clusters with typical PRIMECLUSTER product installations. The appearance of some tabs and menus in the GUI display will change according to the products installed for your platform or market.

> **i** PRIMECLUSTER Configuration Services (PCS) is not installed when the Wizard Tools are installed.

CLI procedures

Some of the operations described in this chapter describe the equivalent CLI procedure. However, we recommend that the CLI be used only by expert system administrators, or in those cases where a browser is not available. If you decide to use a CLI procedure, please note the following:

● The commands are located in the *<RELIANT_PATH>*/bin directory.

● All RMS CLI commands accept both CF node names and RMS node names for SysNode objects when the RMS naming convention has been followed (that is, when the names are of the form *<CF-name>*RMS).

● RMS CLI commands accept case-insensitive application names.

● The CLI procedures presented here are not intended to be a complete description of the commands that are employed. Other options may be available. For more information about any CLI command, see its online manual pages. For the complete list of online documentation related to RMS operation, see "Appendix—List of manual pages".

CLI status codes

In general, RMS `hv*` commands send a request to the base monitor and then return immediately without waiting for this request to be processed. They exit with a status code of 0 (success), which indicates a request has been **sent** to the base monitor successfully. However, this does not guarantee that the request was **processed** successfully.

Important exceptions include the '`hvshut`' and '`hvutil -[mM]`' commands, which remove nodes and applications from RMS control. These commands may return status codes that indicate failure. This type of command also provides an option for "forced" operation, but this should be used with great care and only when absolutely necessary.

5.2   Starting the Cluster Admin GUI

The following sections discuss how to use the RMS portion of the GUI.

i	Windows desktop systems require the Java plugin as specified in "PRIMECLUSTER Web-Based Admin View Operation Guide" for your operating system.

5.2.1   Web-Based Admin View

► Open a Java-enabled browser and enter the following URL in the *Address* location:

>   `http://`*<hostname>*`:8081/Plugin.cgi`

The *hostname* should be the name or IP address of the primary or secondary management server. For example, if a cluster has `shasta1` and `shasta2` as its primary and secondary management servers, the URL would be either one of the following:

● `http://shasta1:8081/Plugin.cgi`

● `http://shasta2:8081/Plugin.cgi`

The `Plugin.cgi` URL always attempts to contact the primary management server; after contacting the host, the browser changes the URL suffix from '.cgi' to '.html'. If you instead use the `Plugin.html` form, Cluster Admin will attempt to make direct contact with the server specified in the URL.

For details on the primary and secondary management servers, refer to "PRIMECLUSTER Web-Based Admin View Operation Guide" for your operating system.

5.2.2   Login

Before logging in, make sure you have a user name and password with the appropriate privilege level. Cluster Admin has the following privilege levels:

● Root privileges—Can perform all actions including configuration, administration, and viewing tasks.

● Administrative privileges—Can view and execute commands, but cannot make configuration changes.

● Operator privileges—Can only perform viewing tasks.

For more details on privilege levels, refer to the *PRIMECLUSTER Installation and Administration Guide* for your operating system.

After the Web-Based Admin View login window appears (Figure 57), log in as follows:

▶ Enter the user name and password for a user with the appropriate privilege level.

▶ Click the *OK* button.
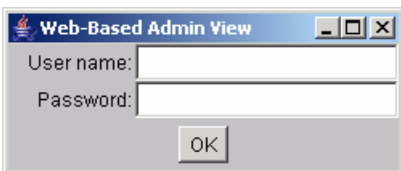
Figure 57: Web-Based Admin View login

After you log in, the Web-Based Admin View window appears (Figure 58).
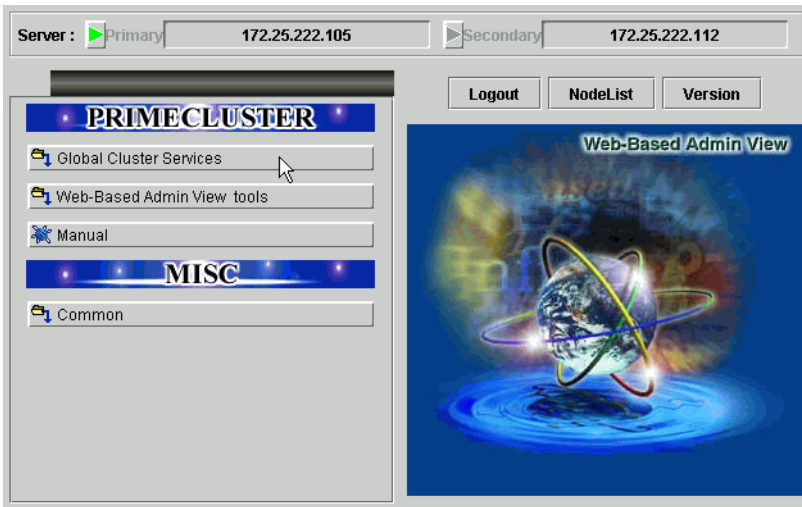
Figure 58: Invoking the Cluster Services GUI

► Click the *Global Cluster Services* button to advance to the next view (Figure 59).
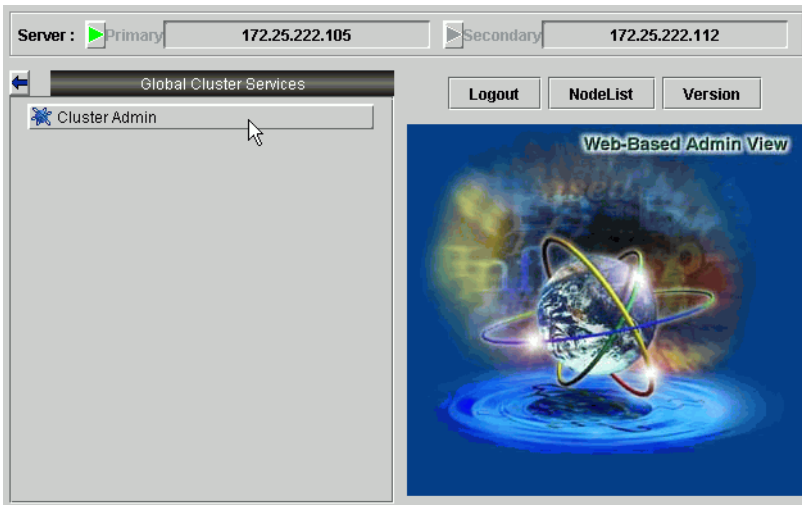
Figure 59: Invoking Cluster Admin

► Click the *Cluster Admin* button.

The *Choose a node for initial connection* window appears (Figure 60).

Figure 60: Cluster Admin initial connection menu

The nodes are displayed in alphabetical order, and the first one is selected by default. In most cases, the node you choose is immaterial for administrative tasks.

► Select the desired node for the connection, and click *OK*.

The next window that appears depends on how you have set the trust levels for the Cluster Admin Java applets. If you have already chosen to use trusted applets for all sessions, you can skip the following description.

Trusted applets

For platform independence and security, the Cluster Admin GUI uses Java applets. When the Java applets run in trusted mode, they are allowed to use some client system resources, such as the clipboard. If you intend to copy and paste text between the Java window and other applications on your workstation, you must run the applets in trusted mode.

| **i** | PRIMECLUSTER trusted applets are signed by VeriSign, Inc. The first time you start the Cluster Admin applet, a *Java Security Warning* dialog allows you to choose the security level for the current and future sessions. |

5.2.3   Main Cluster Admin window

When Cluster Admin opens, the initial view is similar to Figure 61.
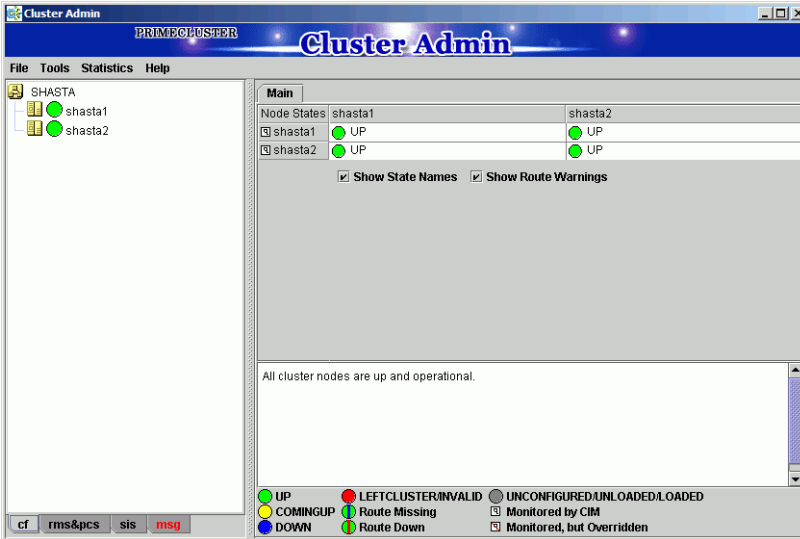
Figure 61: Main Cluster Admin window—Initial view

The following tabs appear at the bottom of the left pane:

● *cf*

● *rms & pcs* (or simply *rms* if PCS is not installed)

● *sis*

● *msg* (message window)

Clicking a tab switches the view to the corresponding product. Initially, the CF view is selected. All views have the following common features:

● **Menu bar**—The pull-down menus contain generic administrative functions as well as items specific to the PRIMECLUSTER products. Some entries may be disabled (grayed out) according to the item selected in the current view.

- ● **Configuration tree**—The left pane contains product-specific configuration information in a hierarchical display. Clicking on an item in the tree will display the item's properties and, in some views, allow you to change them. Right-clicking on an item generally brings up a context menu specific to that item.

- ● **Input and message area**—The large pane on the right is the main work and information area. The content varies according to the product being administered and the functions selected from the menus or tree.

This chapter focuses on RMS administration. For information about operations available on the other tabs, refer to the corresponding product documentation.

5.2.4   Cluster Admin message view

Error and debug messages related to Cluster Admin can be displayed at any time:

► Click the *msg* tab on the bottom of the RMS tree pane. (The tab label is red if a new message has been added to the text area since it was last viewed.) The *Admin Errors and Messages* view appears (Figure 62).
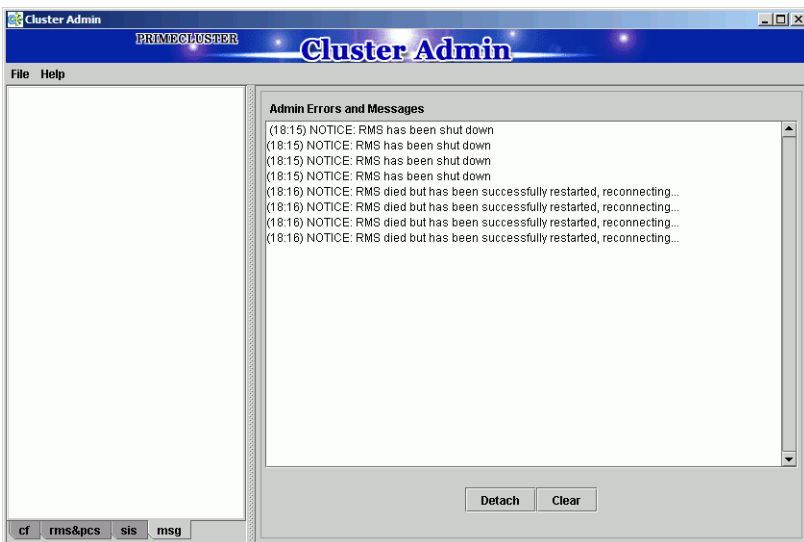
Figure 62: Main Cluster Admin window—message view

The message pane can be detached or re-attached using the buttons at the bottom of the display. Use the *Clear* button to delete all messages in the display.

5.3   Monitoring RMS with Cluster Admin

The procedures in this section allow you to view information about the RMS cluster as well as individual nodes, applications, and resources. These procedures are passive: they display data, but they do not change the operation of the configuration.

► To enter the RMS portion of the Cluster Admin GUI, click on the *rms&pcs* tab.

A typical RMS view is shown in Figure 63.
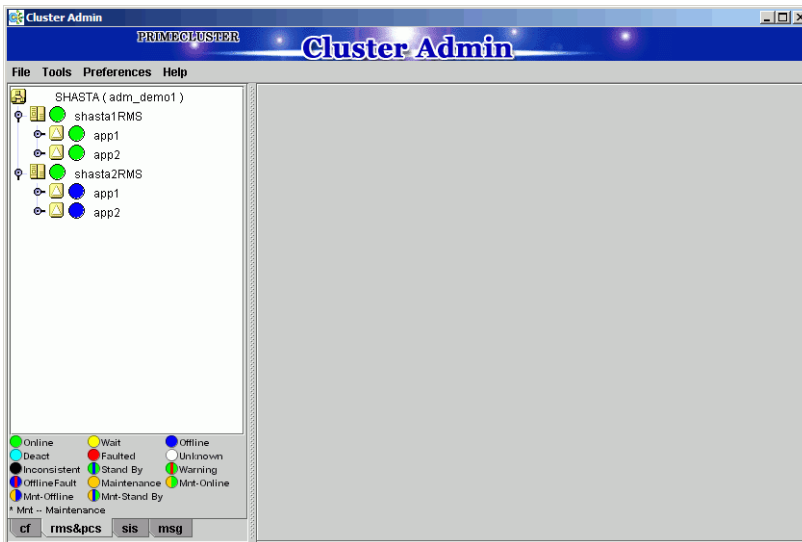
Figure 63: Main Cluster Admin window—RMS view

The main window area is split into two major areas: the left pane contains the RMS tree; the right pane displays configuration information, properties of nodes and objects, RMS logs, or other items. The information displayed depends on what has been selected RMS tree and which operation, if any, has been invoked.

5.3.1 RMS tree

The Cluster Admin RMS tree displays the configuration information for the cluster in a simple hierarchical format. The tree has the following levels:

- Root of the tree—Represents the cluster. The root is labeled with the cluster name, followed by the RMS configuration name in parentheses. The cluster name is determined by your CF configuration.

- First level—Represents the system nodes forming the cluster.

- Second level—Represents the userApplication objects running on each of the system nodes.

- Third level—Represents subapplications, if any. Also contains non-affiliated groups of objects (see fourth level description).

- Fourth level—Represents the resources necessary for each of the subapplications. Also contains non-affiliated objects.

> **i** Non-affiliated andOP and orOP objects are intended for use by RMS experts. These items provide logical dependencies and group connectivity between nodes, applications, and subapplications.

If an application has subapplications, the fourth level represents resources used by that subapplication. If an application does not have subapplications, then the third level represents all the resources used by the userApplication.

Applications always appear at the second level of the tree, even if some of them are controlled by others. Dependencies between applications are depicted in the RMS tree by the presence of controller objects. An example of an RMS tree with a controller object is shown in Figure 64.
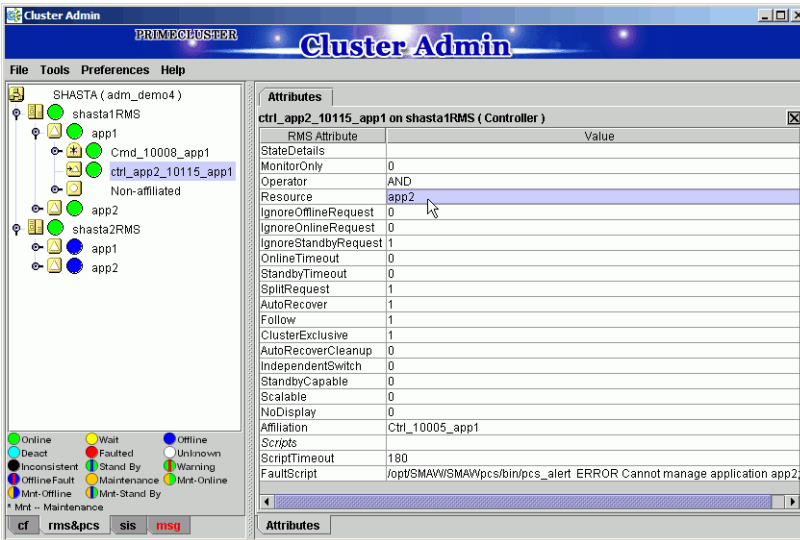
Figure 64: RMS tree with a controller object

In the example shown in the figure, the *Resource* attribute of the highlighted controller object in *app1* contains the name of the *app2* application: this indicates the dependence of *app1* on *app2*.

A complete visual display of object dependence is available from the RMS graph. See the section "Using RMS graphs" for a description of the procedure.

5.3.2   Pop-up context menus

Each object in the Cluster Admin RMS configuration tree has a pop-up context menu that provides quick access to commonly-used operations. Invoke the context menu by right-clicking on any object. The first item on the menu displays the selected object's name in grayed-out text. The remaining items list the available operations, which vary according to the object's type and current state (Figure 65).
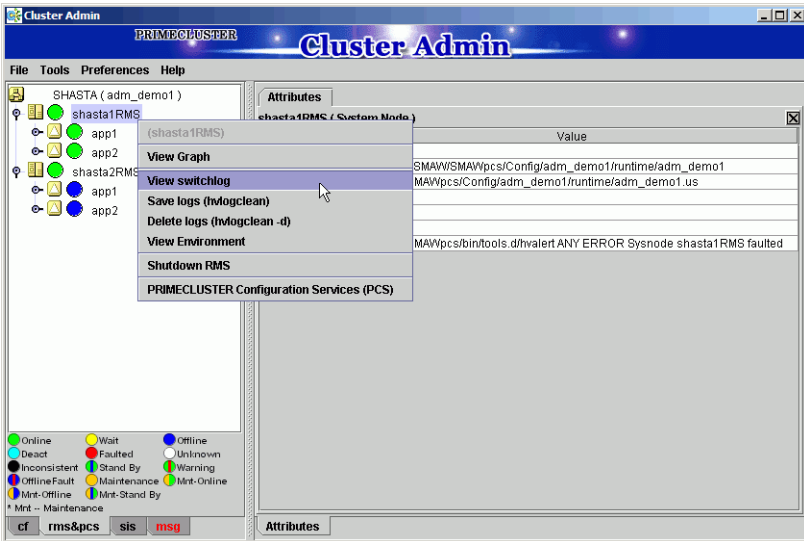
Figure 65: Pop-up context menu for a node

Items that affect object states, node states, or the entire RMS configuration appear toward the bottom of the menu. These actions are described in the final sections of this chapter.

> **i** The *PRIMECLUSTER Configuration Services (PCS)* item on node context menus (shown at the bottom of the menu in the above figure) will not appear if the RMS Wizard Tools are installed.

> **i** To close the context menu without performing an operation, click the grayed-out object name in the menu or press the ⎡Esc⎤ key.

The menu offers different operations for a node object compared to an application object. It also offers different options for an application object in the online state (Figure 66) compared to the offline state (Figure 67).
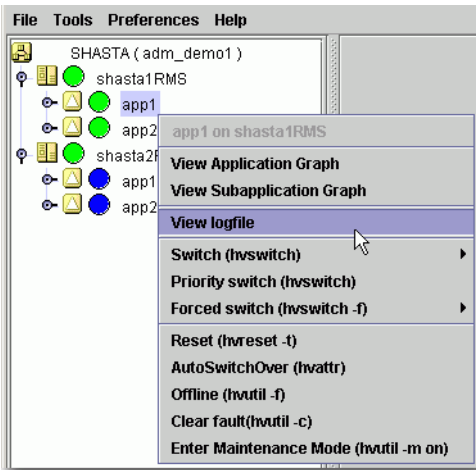
Figure 66: Pop-up context menu for an online application
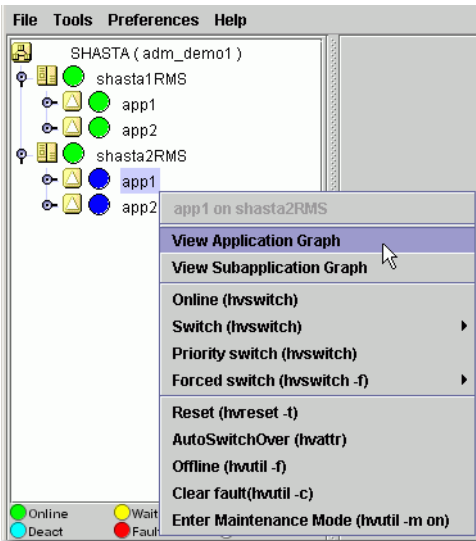
Figure 67: Pop-up context menu for an offline application

5.3.3   Pop-up confirmation dialogs

When you select an item in an object's context menu that can cause state changes to that object, a pop-up confirmation dialog appears (Figure 68). To proceed with the action described in the warning message, click *Yes*; to cancel the action, click *No*.
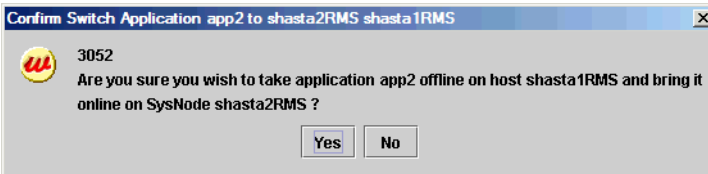
Figure 68: Pop-up confirmation dialog

5.3.4   Displaying environment variables

Display the global (clusterwide) environment variables as follows:

► Right-click on a cluster in the RMS tree window and select *View Environment* from the context menu (Figure 69).
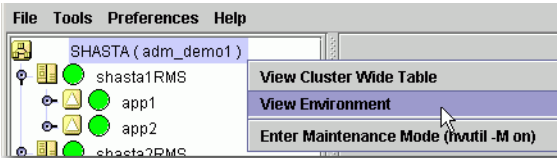
Figure 69: Displaying global environment variables

The global variables will appear under a separate tab in the right pane (Figure 70).
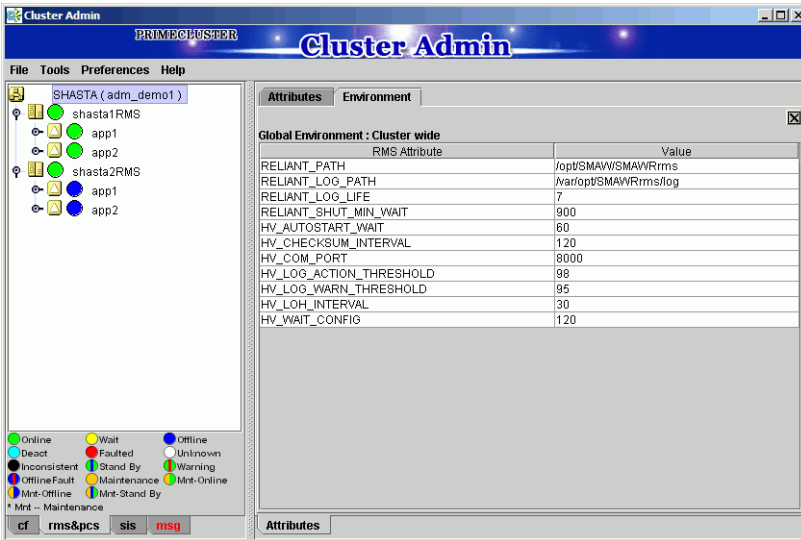
Figure 70: Global environment variable view

| **i** | To close a tab view, use the button in the upper-right corner of the view. |

Display local environment variables as follows:

► Right-click on a node in the RMS tree window and select *View Environment* from the context menu (Figure 71).
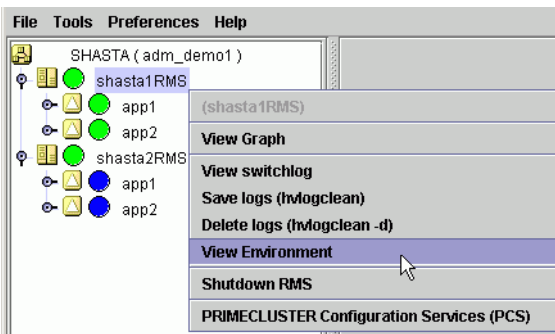
Figure 71: Displaying local environment variables

Both local and global variables appear on the same tab in the right pane
(Figure 72).
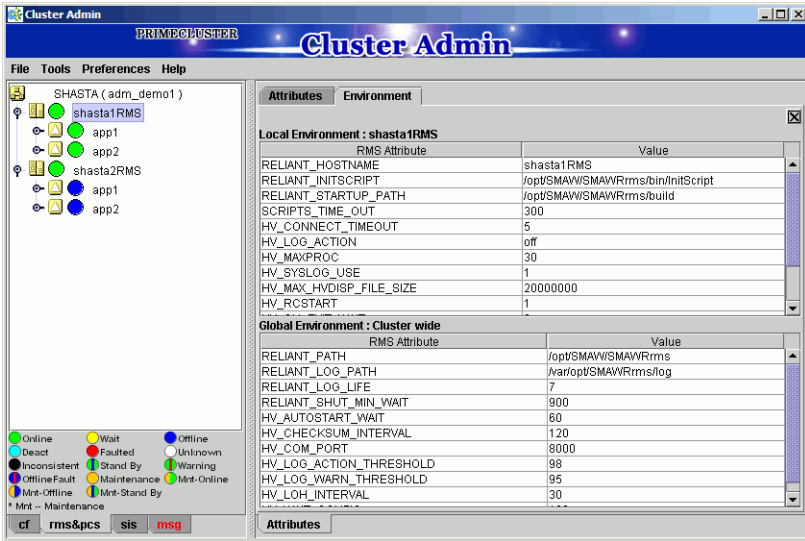
Figure 72: Local environment variables view

CLI: hvdisp

Display the environment variables with the `hvdisp` command, which does not
require root privilege:

```
hvdisp ENV
hvdisp ENVL
```

5.3.5   Displaying object states

The state of each RMS object is indicated by the color of its circular status icon,
located immediately to the left of the object's name in the configuration tree. The
legend for the object states appears below the RMS tree in the left pane of the
RMS view (Figure 73).
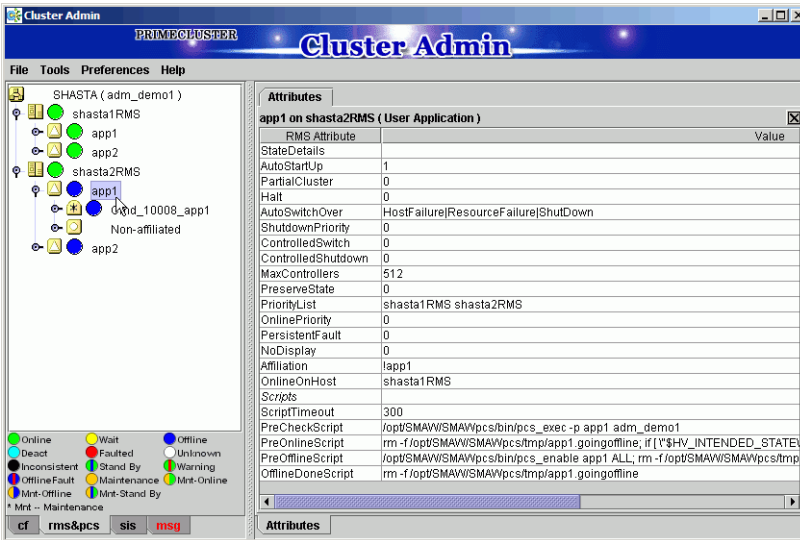
Figure 73: Displaying application and object states

In the example above, the application `app2` is online (green status icon) on node `shasta1RMS`, but offline (blue status icon) on node `shasta2RMS`.

See "Appendix—States" for a description of RMS object states.

CLI: hvdisp

The syntax for the CLI is as follows:

> hvdisp {−a | −c} [−o *out_file*]

Options:

−a   Displays the object name, the object type, the object's `SysNode` name, and the object state for each object in the configuration (automatically generated connectors are not shown)

−c   Displays information in compact format

−o   Sends the output to the designated file

The `hvdisp` command only works when RMS is running and does not require root privilege.

5.3.6   Configuration information or object attributes

View configuration information for individual objects by left-clicking with the mouse on the object in the tree. The properties are displayed in a tabular format on the right pane of the RMS main window (Figure 74).
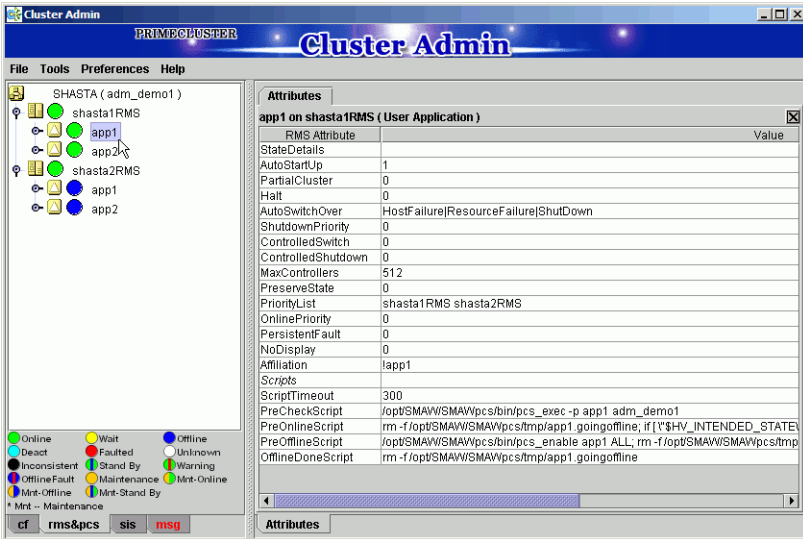
Figure 74: Configuration information or object attributes

6 Additional administrative tools

The Cluster Admin GUI provides several additional tools to help you manage RMS operation.

Chapter contents:

- "Using the RMS clusterwide table"

- "Using RMS graphs"

- "Interpreting display changes"

- "Viewing RMS log messages"

6.1 Using the RMS clusterwide table

The RMS clusterwide table displays the state of each application on each of the system nodes in a concise table.

▶ To open the clusterwide table, right-click the cluster name (the root of the RMS tree in the left pane) and then select *View Cluster Wide table* from the context menu (Figure 75).
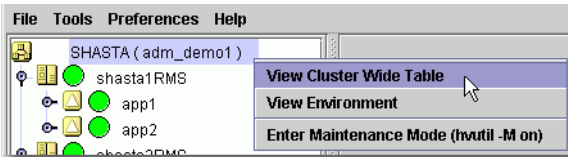
Figure 75: Opening the clusterwide table

The clusterwide table appears in a separate window (Figure 76).
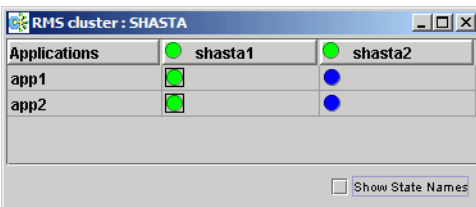
Figure 76: Clusterwide table

► To display the corresponding state name next to each status icon, click the *Show State Names* checkbox at the lower-right corner of the window (Figure 77).
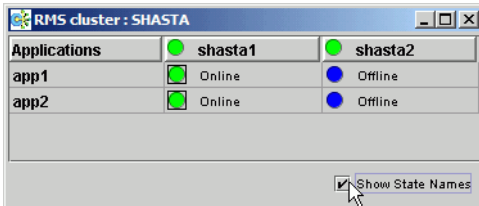
Figure 77: Clusterwide table with state names

You can increase or decrease the size of the clusterwide table window and the size of the columns by using the mouse. If the window is already large enough to fully display all of the table elements, then you will not be allowed to further increase its size.

A square surrounding the colored state circle indicates the primary node for the application. Figure 77 above shows that shasta1 is the primary node for all of the applications.

Normally, the clusterwide table displays applications in alphabetical order from top to bottom. However, Faulted applications are handled specially. If an application is in the Faulted state on any node in the cluster, then it is displayed at the top of the table, and the application's name is highlighted by a pink background (Figure 78). This allows the System Administrator to easily spot any Faulted applications.
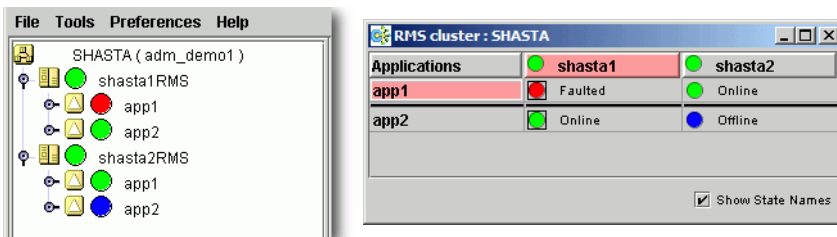
Figure 78: Faulted applications in the clusterwide table

The clusterwide table also makes special provisions for applications that are not online anywhere in the cluster. These applications are also displayed at the top of the table, with the application's name highlighted in light blue (Figure 79). This alerts the system administrator that some applications are not running anywhere and should probably be brought online on some node.
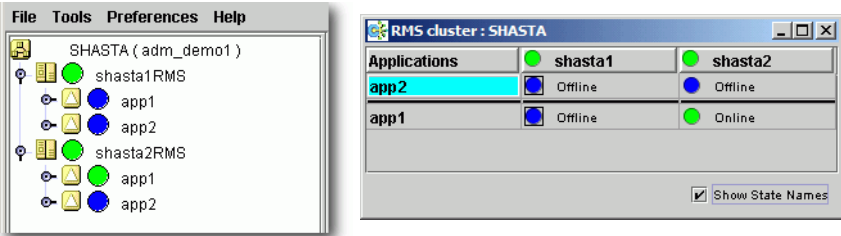
Figure 79: Offline applications in the clusterwide table

If there are both faulted applications and applications that are not online anywhere, then the faulted applications are listed first (Figure 80).
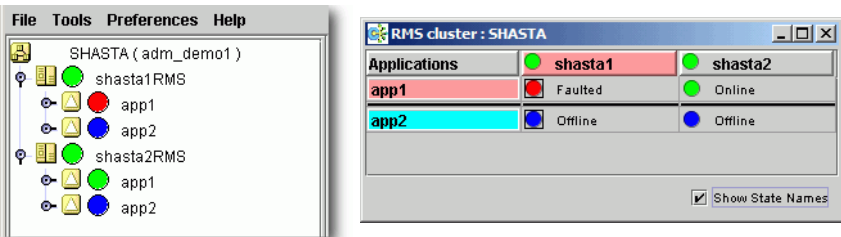
Figure 80: Faulted and offline applications in the clusterwide table

If there is a split-brain condition in the cluster on both the clusterwide table and the RMS tree, then colored exclamation marks will appear after the status icons (colored circles) of the nodes. A colored exclamation mark indicates that the state of that SysNode is different from what another SysNode views it as being. The color of the exclamation mark indicates the state that the other node thinks that the SysNode is in. If there are multiple nodes that see a SysNode in different states, you will see multiple exclamation marks after the colored circle. Exclamation marks are sorted according to the severity of the states.

Figure 81 shows a clusterwide table with an application of a split-brain condition. Note the yellow exclamation mark before the second node name.
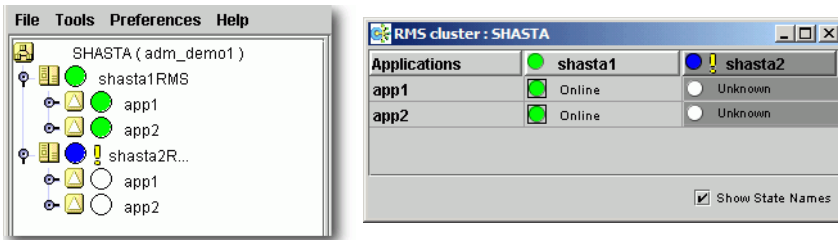
Figure 81: Split-brain conditions in the clusterwide table

| **i** | Momentary split-brain conditions may be indicated while a node starts up or shuts down. |

6.1.1   Using context menus from the clusterwide table

Each object in the clusterwide table has a pop-up context menu that provides quick access to commonly-used operations.

► To display a context menu, right-click on any object with a status icon, where column headers represent hosts and cells represent applications.

The first item on the menu displays the selected object's name in grayed-out text. The remaining items list the available operations, which vary according to the object's type and current state (Figure 82).
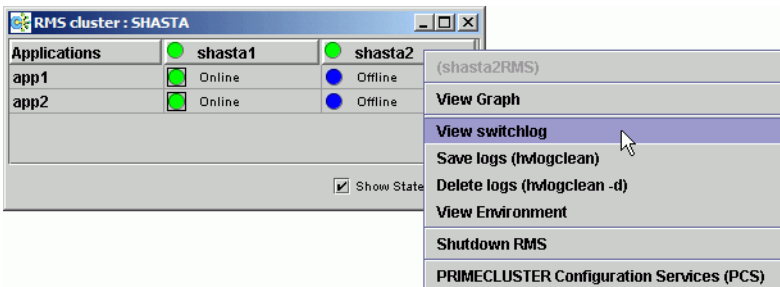
Figure 82: Using context menus in the clusterwide table

Items that affect object states, node states, or the entire RMS configuration appear toward the bottom of the menu. These actions are described in the final sections of this chapter.

▶ To close the context menu without performing an operation, click the grayed-out object name in the menu or press the $\boxed{\text{Esc}}$ key.

6.2 Using RMS graphs

Cluster Admin provides an alternate way of viewing the RMS configuration hierarchy called **graphs**. A graph represents the configuration in a true tree structure, where the branches indicate the dependencies that are not generally visible in the RMS configuration tree described earlier. The following types of graphs are available:

● Full graph—Displays the complete cluster configuration.

● Application graph—Shows all of the resources used by an application and can be used to look at specific resource properties.

● Subapplication graph—Lists all of the subapplications used by a given application, and it shows the connections between the subapplications.

● Composite subapplications graph—Shows all the subapplications that the application depends on directly or indirectly.

The following sections describe each type in more detail, as well as these graph-related features:

● Obtaining configuration information

● Using command context (pop-up) menus

● Displaying various levels of detail

● Interpreting display changes in the clusterwide table and graphs

6.2.1 RMS full graph

▶ To display the RMS full graph, right-click on any system node and select *View Graph* from the context menu (Figure 83).
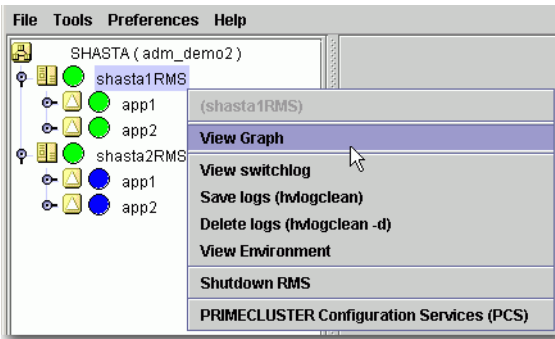
Figure 83: Viewing the RMS full graph on a node

> **i** The *View Graph* menu item is not available if a graph is already open for that node.

By default, each graph appears as a separate tab in the right pane of the Cluster Admin view (Figure 84).
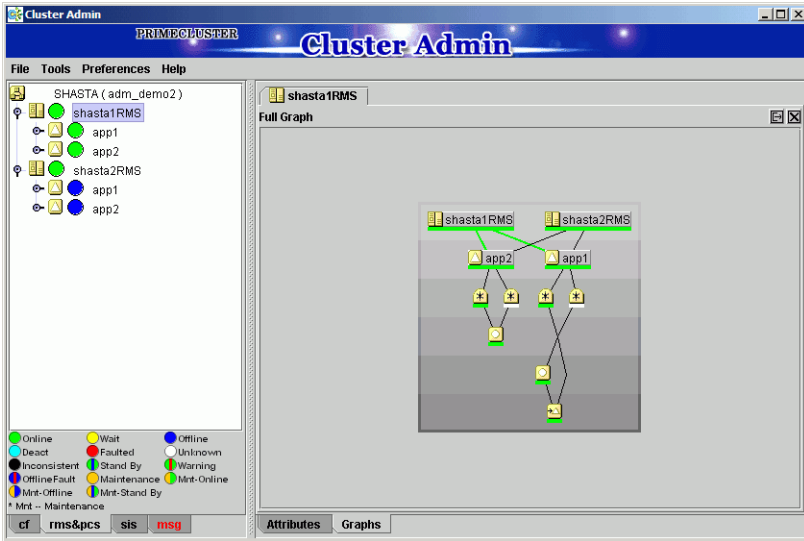
Figure 84: RMS full graph on a node—tab view

► To view any tab in a separate window, click the **detach** control button. The detach button is located next to the **close** control button in the upper-right corner of the view (Figure 85).
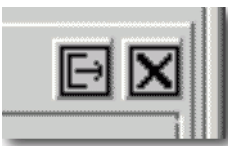
Figure 85: Detail of tab view showing detach button

The detached view contains the same information as the tabbed view (Figure 86).
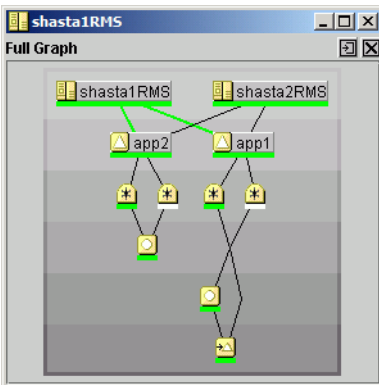
Figure 86: RMS full graph on a node—separate window view

► To rejoin the detached window to the Cluster Admin view, click the **attach** control button. The attach button is located next to the view's **close** control button in the upper-right corner, just below the standard window control buttons (Figure 87).
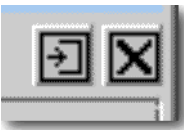
Figure 87: Detail of separate window view showing attach button

The RMS full graph displays the complete RMS configuration of the cluster and represents the following items:

● Node where each application is currently online, indicated by green lines between the node and application objects

● Object types, indicated by the object's icon

● Current object state, indicated by the colored bar beneath each icon

● Relationships between objects

● Dependencies of objects

The RMS graph is drawn from the perspective of the selected node; that is, the state information of all other objects is displayed according to the reports received by that node. The node name in the title bar of the graph identifies the node that is supplying the state information. You can create an RMS graph from the perspective of any node in the cluster.

The background of the graph is shaded from top to bottom with progressively darker gray bands. In large, complicated graphs, this can help to locate objects and identify their dependency level.

If you position the mouse cursor over an object in the graph, the cursor changes to a crosshair and the object's name appears as a tool tip. Also, the connector lines radiating from the object are highlighted with yellow to indicate its parent and child dependencies (Figure 88).
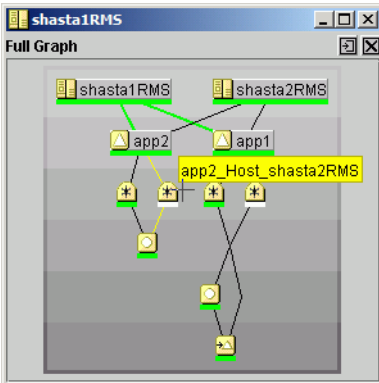
Figure 88: RMS full graph—object tooltip

Clicking on the object brings up a window with further details such as the object's attributes (Figure 89).
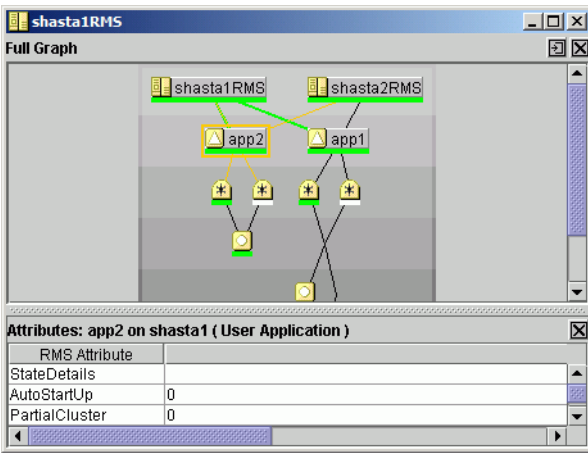
Figure 89: RMS full graph—object details

6.2.2 Application graph

► To display the graph for a single application, right-click on the application object and select *View Application Graph* from the context menu (Figure 90).
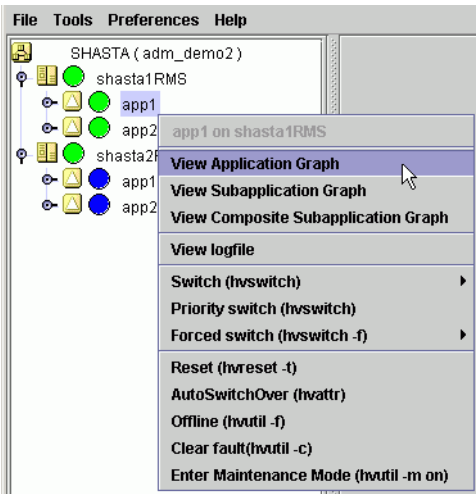
Figure 90: Viewing an RMS application graph

The application graph shares the same features as the full graph, except that it shows only the selected application and its resources (Figure 91). Like the full graph, the application graph is shown from the perspective of the selected node, and tooltips and details are available for every object.
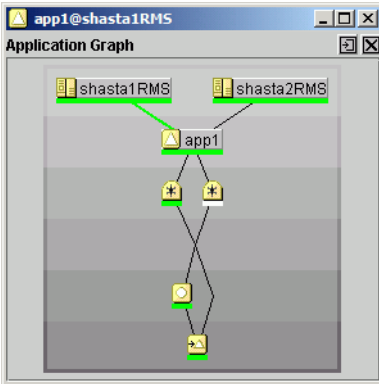
Figure 91: Typical RMS application graph

6.2.3   Subapplication graph

▶   To display a subapplication graph, right-click on the parent application and select *View Subapplication Graph* from the context menu (Figure 92).
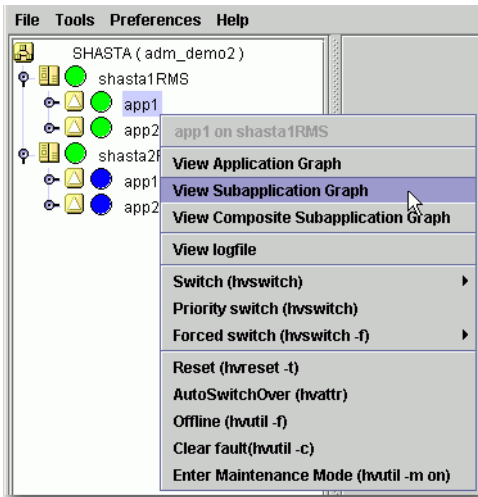
Figure 92: Viewing an RMS subapplication graph

This graph displays all the subapplications used by the selected application, showing the connections between the subapplications (Figure 93).
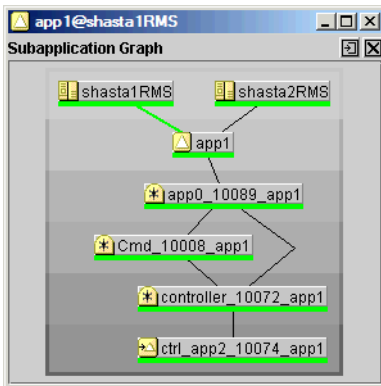
Figure 93: Typical RMS subapplication graph

For clarity, names of the subapplication objects are shown as labels rather than tooltips, and various abstractions such as non-affiliated objects are not included. Like other graphs, clicking on an object brings up a window that displays its attributes.

6.2.4   Composite subapplication graph

i The composite subapplication graph is available only for applications with controller objects.

When the configuration includes a controller object, the full graph of the node (or of either application) shows the parent and child applications in separate branches (Figure 94).
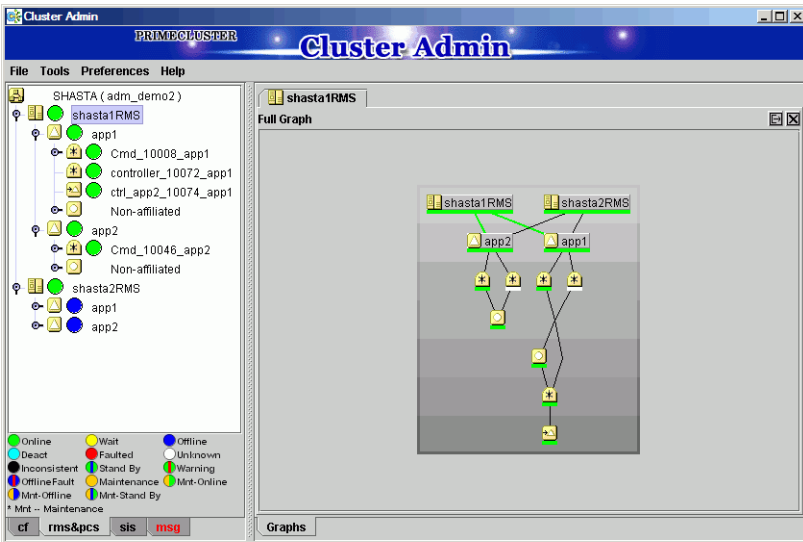
Figure 94: Standard view of configuration with controlled application

In some cases, it is convenient to view the child application as a resource of the parent so they both appear in the same branch. This is called a **composite subapplication graph**.

► To view a composite subapplication graph, right-click on the parent application and select *View Composite Subapplication Graph* from the context menu (Figure 95).
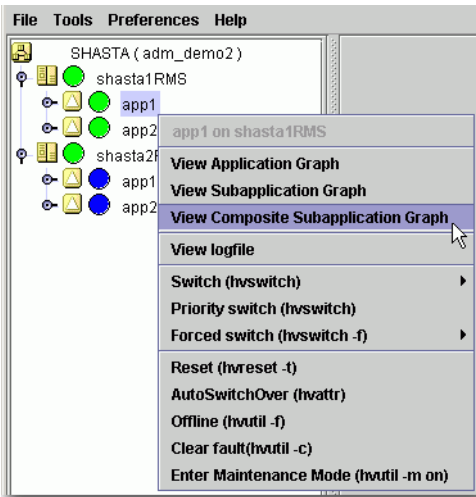
Figure 95: Viewing an RMS composite subapplication graph

For every controller object in the subapplication graph, the graph of its
controlled application is inserted with a dotted line connection to the parent
controller. For example, note where *app2* appears in the composite subappli-
cation graph in Figure 96 below, and compare this to the standard graph shown
earlier in Figure 94.
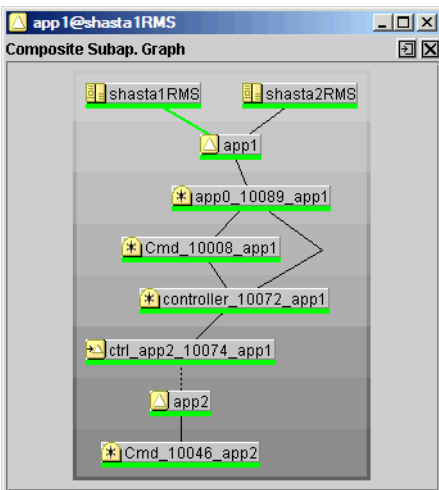
Figure 96: Typical composite subapplication graph

If the controlled application has its own controller objects, then the process is recursively repeated. This gives a composite view of all the subapplications that the selected parent application depends on, whether directly or indirectly.

6.2.5   Using pop-up context menus from the graph

Each object in the RMS graph has a pop-up context menu that provides quick access to commonly-used operations.

▶   To display a context menu, right-click on any object in a graph.

The first item on the menu displays the selected object's name in grayed-out text. The remaining items list the available operations, which vary according to the object's type and current state (Figure 97).
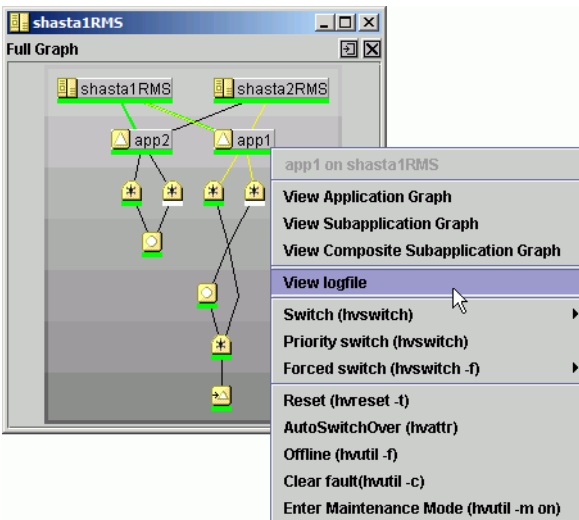
Figure 97: Using a pop-up context menu from the RMS graph

Items that affect object states, node states, or the entire RMS configuration appear toward the bottom of the menu. These actions are described in the final sections of this chapter.

▶   To close the context menu without performing an operation, click the grayed-out object name in the menu or press the ⎡Esc⎤ key.

6.2.6   Changing the displayed detail level

By default, the RMS graph does not display the resource (object) names on the graphs. These are available as tool tips and can be seen by placing the mouse over a particular object.

► To add any combination of resource names, affiliation names, "NoDisplay" nodes to the graphs, use the checkboxes on the *Preferences* menu.

Figure 98 shows the preference setting and a corresponding graph that displays resource names.
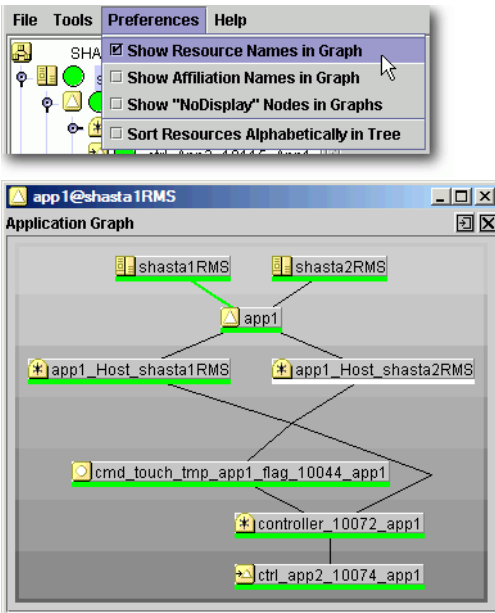
Figure 98: Displaying an RMS graph with affiliation names

Figure 99 shows the preference setting and a corresponding graph that displays affiliation names.
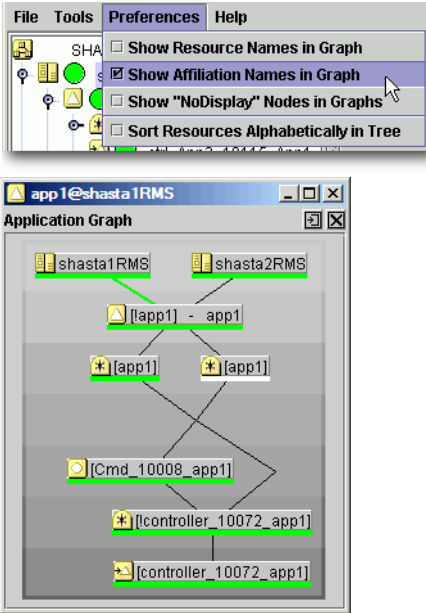
Figure 99: RMS graph with resource names

Figure 100 shows the preference setting and a corresponding graph that displays "NoDisplay" objects. These are typically logical AND/OR objects that are automatically generated by the configuration tool.
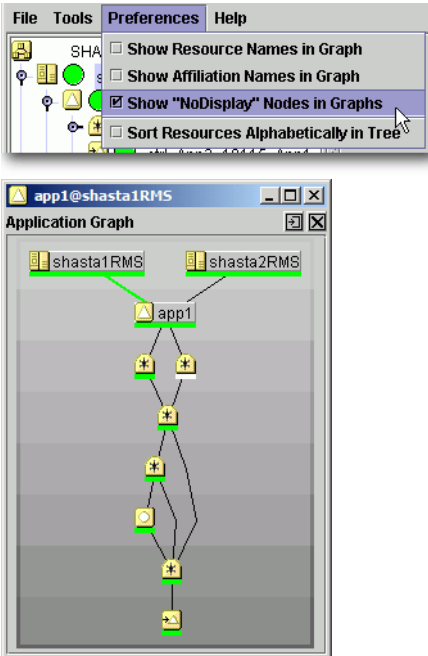
Figure 100: RMS graph with resource names

If two or more display options are selected, some of the object and resource names will consume more on-screen space (Figure 101). Horizontal scrollbars will appear when the graph is larger than the width of the screen, but the graph may still be difficult to read.
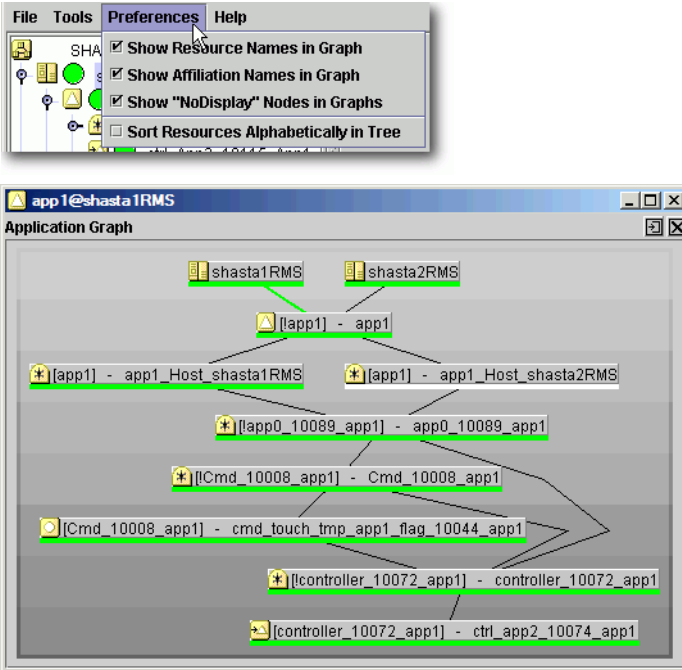
Figure 101: RMS graph with resource and affiliation names

In a complicated graph, it may help to sort the resource names alphabetically (Figure 102).
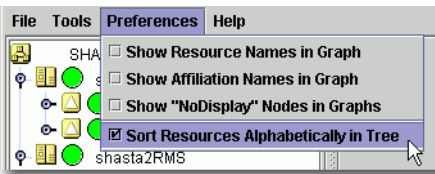
Figure 102: Sorting object names in the graph

6.3   Interpreting display changes

The Cluster Admin view, the graphs, and the clusterwide table all have methods to denote the state of individual nodes as well as the overall configuration.

6.3.1   Display during RMS configuration changes

When you stop and restart RMS with a different configuration, the RMS tree, the clusterwide table, and the node graphs are redrawn in the same windows.

Figure 103 illustrates a Cluster Admin view overlaid by two individual node graphs and the clusterwide table, all displaying the state of a running configuration that monitors app1 and app2.
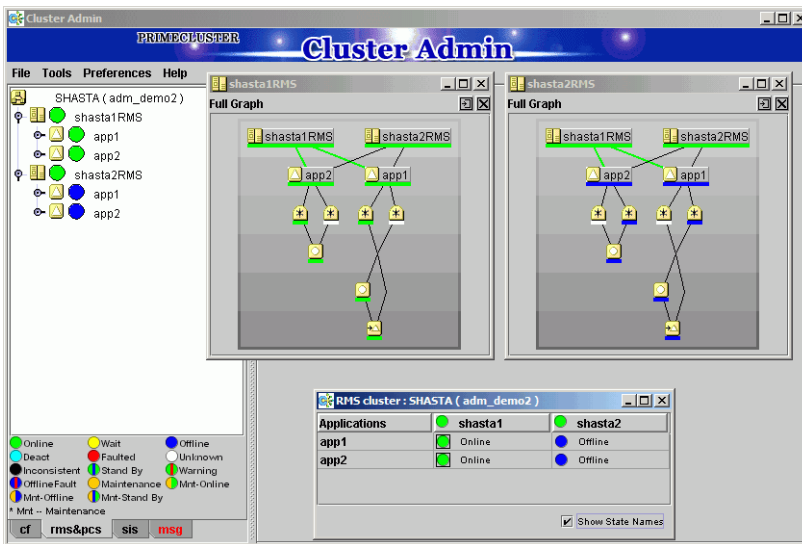
Figure 103: Cluster state before RMS is shut down

Figure 104 shows the same windows after RMS has been restarted, this time with a different configuration that monitors appA and appB.
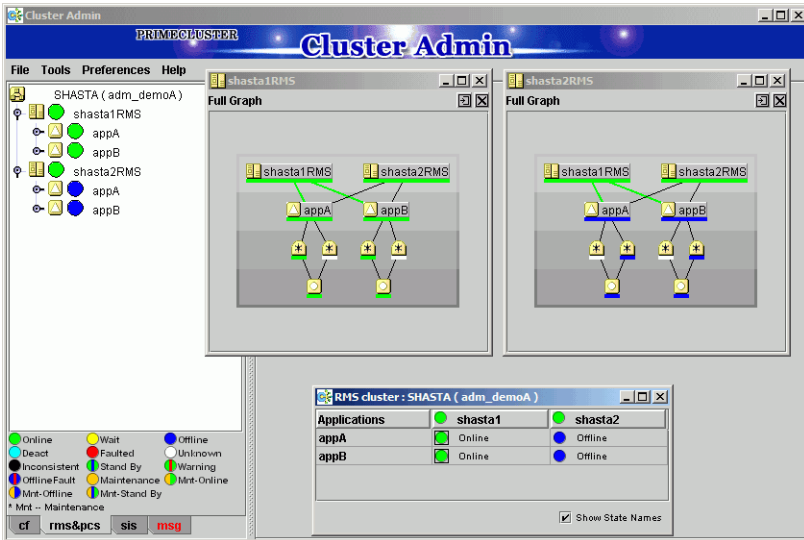
Figure 104: Cluster state after RMS restart with different configuration

The graphs and clusterwide table display the status of the same `SysNode` objects (`shasta1` and `shasta2`) before and after the restart, so the windows remain open.

6.3.2   Display after RMS shutdown

After RMS is shut down, the background of RMS graph windows become dark gray on the node from which they are getting their information. In this condition, all the states are white, indicating that the states are unknown. The main view and the clusterwide table will continue to show the active application states until RMS is shut down on all nodes.

For example, suppose RMS is shut down only on one node (`shasta1`) of our example cluster. The graph on that node and the corresponding column of the clusterwide table will be shaded with a dark gray background (Figure 105).
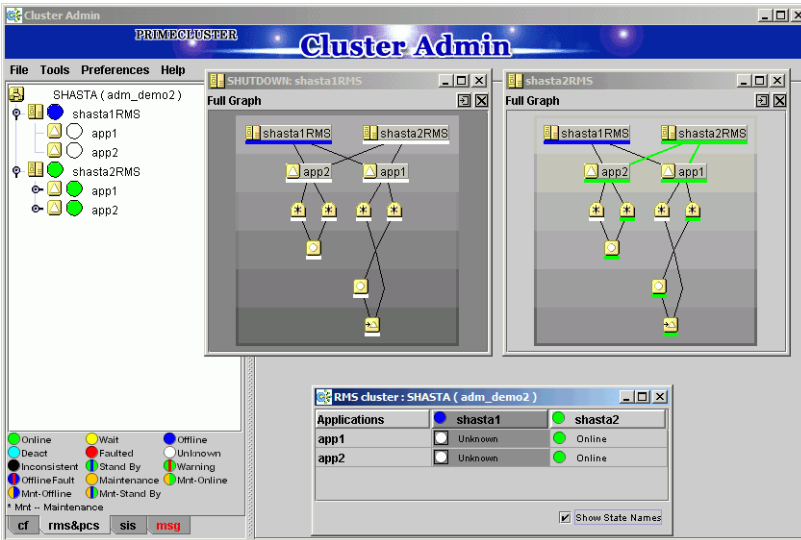
Figure 105: RMS main view, graphs, and clusterwide table after shutdown on one node

However, as long as RMS continues to run on the remaining node, shasta2, the RMS main view, the shasta2RMS graph, and the shasta2 column in the clusterwide table will indicate the online objects.

Application and subapplication graphs

When a node shuts down, any application or subapplication graph on that node would also change to the shutdown state (Figure 106).
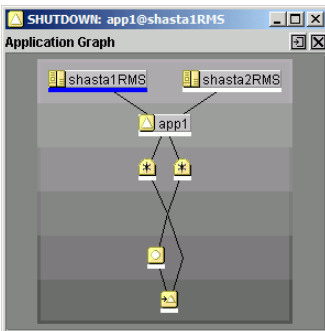
Figure 106: Application graph on shutdown node

The graph retains this appearance until the node restarts with the same config-uration, at which point it returns to its normal display.

If the node restarts with a different configuration, the graph contents are deleted, because the object(s) no longer exist in the new configuration. However, the empty graph window remain opens until explicitly closed (Figure 107).
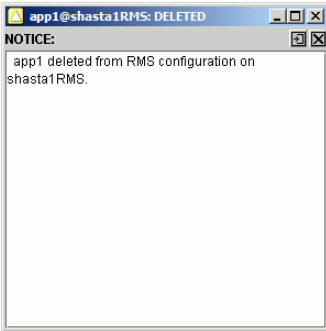
Figure 107: Graph window for deleted object

●

6.4 Viewing RMS log messages

The Cluster Admin interface provides a log viewer that lets you view and filter entries in the RMS switchlog and individual application logs on any node.

> **i** All RMS log files, which normally reside in /var/opt/SMAWRrms/log/, can be viewed directly using a standard UNIX editor like vi.

> **i** For meanings of error messages and possible corrective actions, refer to the *RMS Reference Guide*.

View the switchlog for a system node as follows:

► Right-click on the system node and select *View Switchlog* from the pop-up context menu (Figure 108). Alternatively, select a node and use *Tools –> View switchlog* (Figure 109).
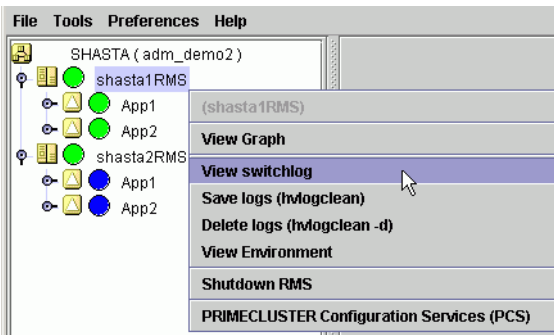
Figure 108: Viewing the RMS switchlog file using a context menu
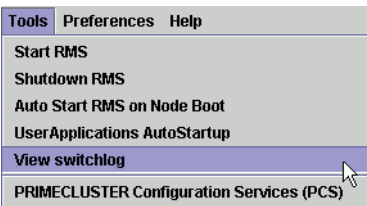
Figure 109: Viewing the RMS switchlog file using the Tools menu

View an application log as follows:

▶ Right-click on an application on the RMS tree and choose *View logfile* from the pop-up context menu (Figure 110).
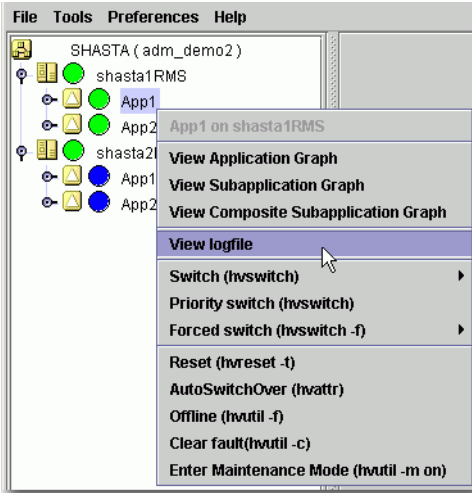
Figure 110: Viewing an application log using a context menu

> **i** You can invoke equivalent context menus for an object from the Cluster Admin view, from the clusterwide table, or from any RMS graph containing that object.

By default, each log file is displayed in a separate tab in the right pane
(Figure 111).
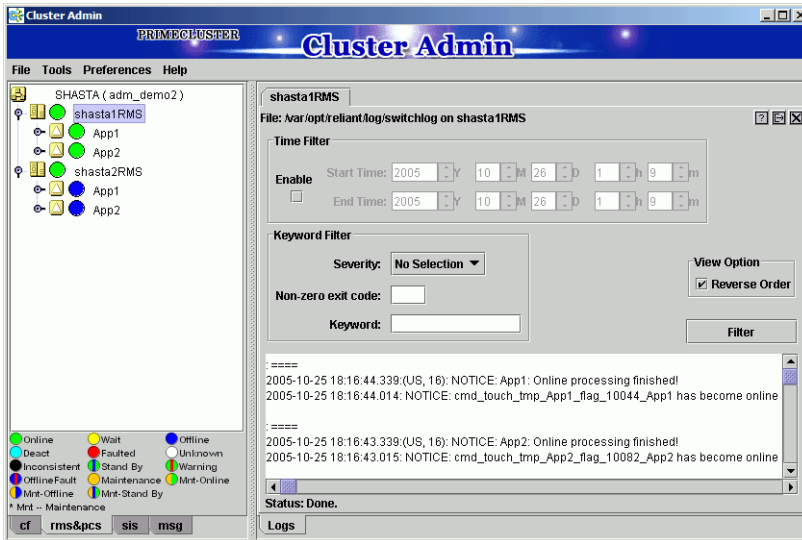
Figure 111: RMS switchlog in tab view

► To view any tab in a separate window, click the **detach** control button. The
detach button is located between the **help** and **close** control buttons in the
upper-right corner of the view (Figure 112).

Figure 112: Detail of tab view showing detach button

The detached view contains the same information as the tabbed view
(Figure 113).
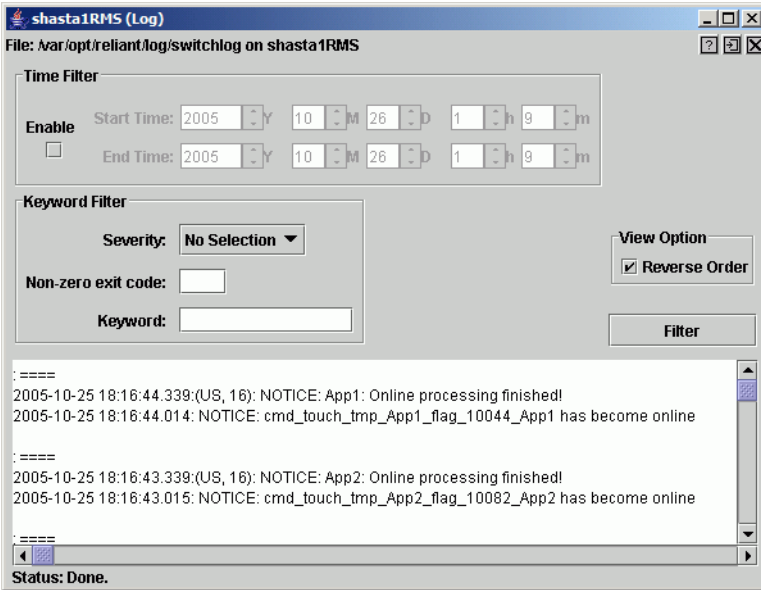
Figure 113: RMS switchlog in detached view

► To rejoin the detached window to the Cluster Admin view, click the **attach**
control button. The attach button is located between the view's **help** and
close control buttons in the upper-right corner, just below the standard
window control buttons (Figure 114).

Figure 114: Detail of detached window view showing attach button

> **i** While in detached mode, the view's close button and the standard
> window **close** button serve the same purpose: they both close the
> detached window.
>
> In attached mode, the tabbed view's **close** button closes only the visible
> tab. All other tabs remain open.

6.4.1 Common procedures for switchlog and application log

By default, the entire log is available in the scrolled area at the bottom of the window. You can restrict the entries displayed with the following filters, which are described in subsections below:

● *Time Filter*—defines the time period of interest.

● *Keyword Filter*—selects a particular resource name (for an application only), error message severity level, non-zero exit code, or keyword.

> **i** Refer to the *RMS Reference Guide* for a complete description of severity levels and exit codes.

► After you enter your filter criteria, click the *Filter* button to display the filtered log entries.

> **i** All the selected and non-blank *Time Filter* and *Keyword Filter* controls are combined with a logical AND operation.

At any time, you can sort the displayed switchlog entries according to increasing or decreasing time by checking or unchecking the *Reverse Order* checkbox in the log viewer window.

6.4.2   Time filter

The controls in the *Time Filter* panel allow you to limit the entries displayed in the log pane according to their date and time (Figure 115).
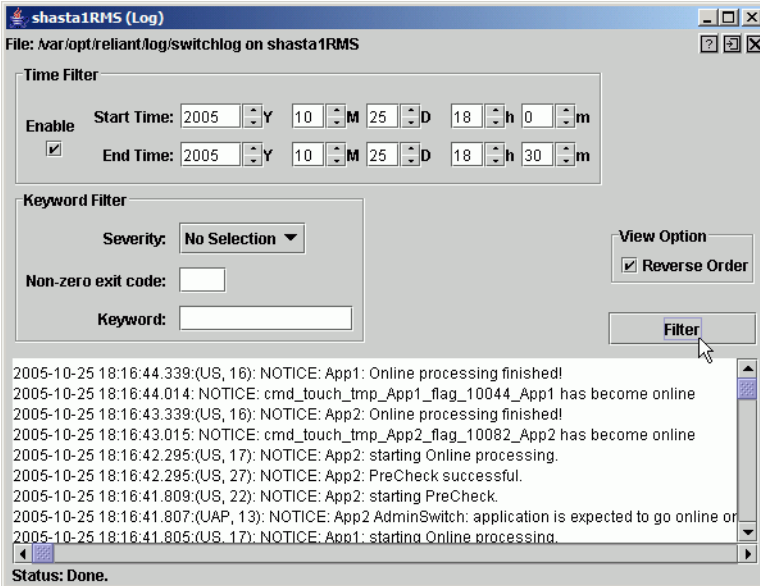
Figure 115: Search based on date and time range

▶ Select the *Start Time* and *End Time* using the scrolling input boxes (you can also type in the values directly) and then check the *Enable* checkbox.

The controls take effect the next time you click the *Filter* button.

▶ To remove the time filter, uncheck *Enable* and then click *Filter*.

6.4.3   Keyword filters

The following items are available in the *Keyword Filter* panel.

6.4.3.1   Resource Name

i   The *Resource Name* control is available only for application logs.

► Select a resource name from the dropdown list (Figure 116) and then click
Filter.
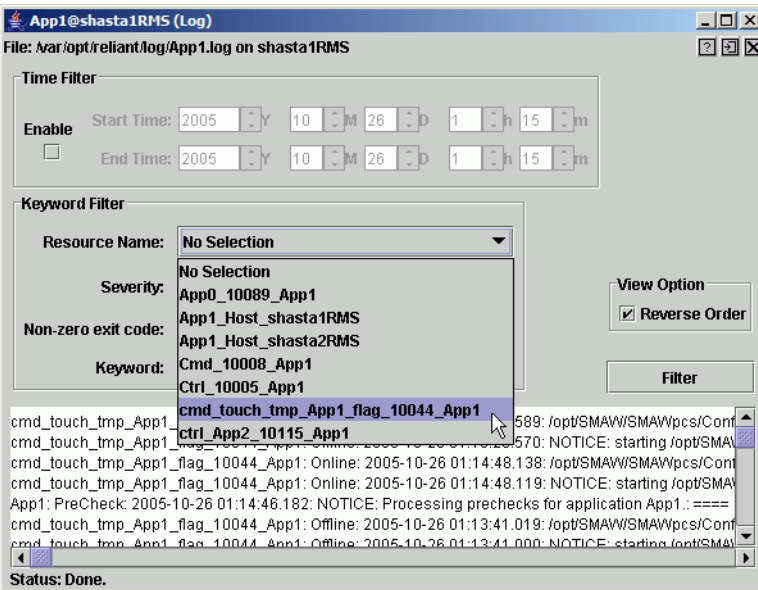
Figure 116: Search based on resource name

► To remove the resource name filter, select *No Selection* from the dropdown
list and then click *Filter*.

6.4.3.2   Severity

► Select an message severity level from the dropdown list (Figure 117) and then click *Filter*.
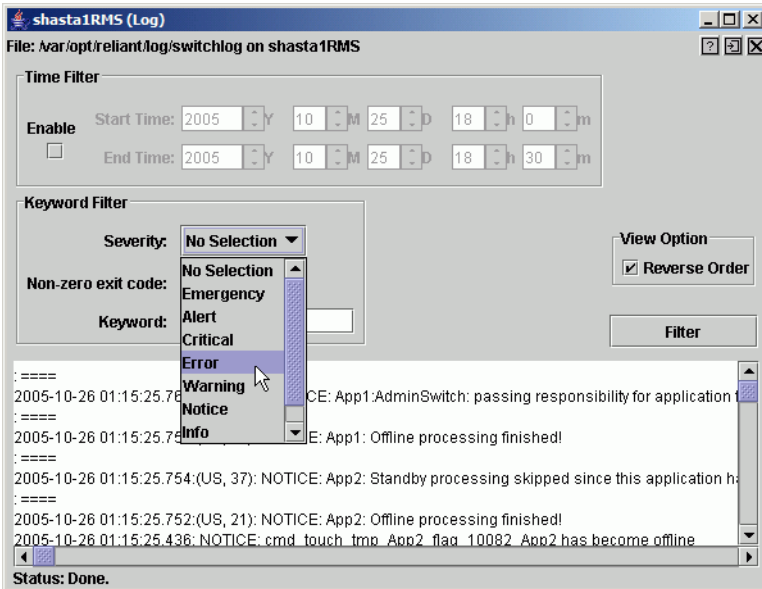
Figure 117: Search based on severity level

Table 4 summarizes the RMS message log viewer severity levels.

Severity level	Description
Emergency	Systems cannot be used
Alert	Immediate action is necessary
Critical	Critical condition (fatal error)
Error	Error condition (non-fatal error)
Warning	Warning condition
Notice	Normal but important condition
Info	Miscellaneous information
Debug	Debug messages

Table 4: RMS severity level description

► To remove the severity level filter, select *No Selection* from the dropdown list and then click *Filter*.

6.4.3.3   Non-zero exit code

► Enter a numeric exit code in the *Non-zero exit code* input box and then click *Filter*.

6.4.3.4   Keyword

► Enter a string in the *Keyword* box (Figure 118) and then click *Filter*.
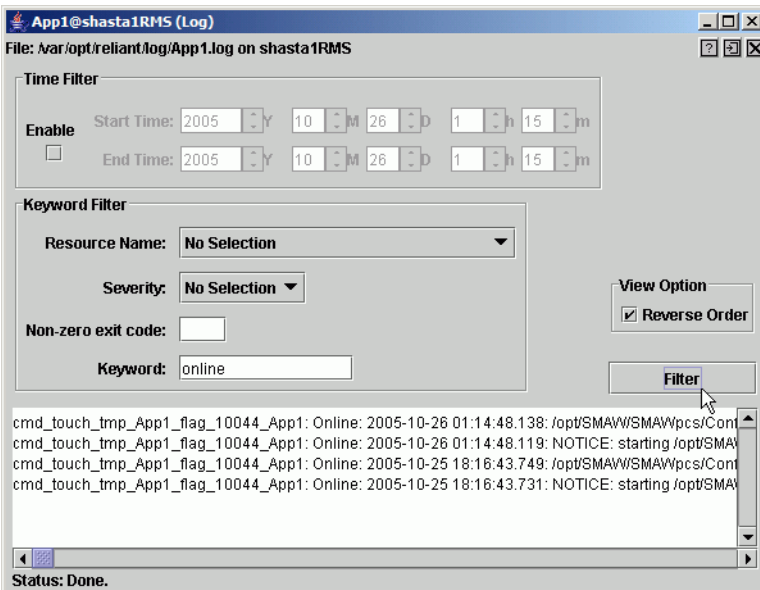
Figure 118: Search based on keyword

> **i**   Special characters and spaces are valid, but wildcards are not inter-preted. This search is **not case-sensitive**.

► To remove the keyword filter, clear the text in the *Keyword* box and then click *Filter*.

6.4.4   Text search

You can search the text in the application log by right-clicking on the displayed text. A pop-up dialog with a *Find* entry allows you to perform a **case-sensitive** search in the direction you specify (Figure 119).
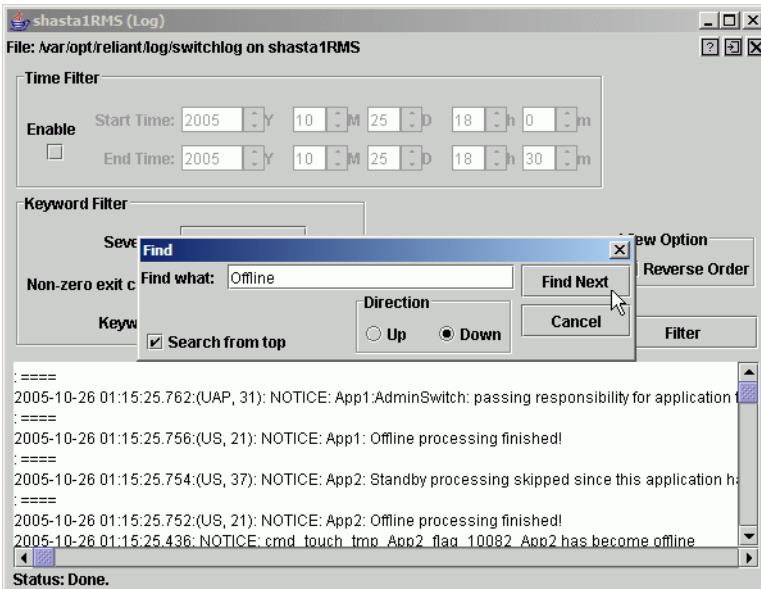
Figure 119: Using the pop-up Find dialog in log viewer

| **i** | The *Find* search string is processed literally You can include spaces and special characters, but wildcards are not interpreted. |

6.4.5   Removing filters

To remove all filters, take the following steps:

● Uncheck the time filter *Enable* box.

● Set drop-down lists to *No Selection*

● Clear text from input boxes

● Click the *Filter* button

The unfiltered view will be restored.

7 Controlling RMS operation

Chapter contents:

● " Managing RMS nodes"

● "Managing RMS applications"

● "Using maintenance mode"

7.1 Managing RMS nodes

This section describes basic procedures to control the operation of RMS, including how to start and stop individual nodes or the entire cluster. Procedures in this section are active: they change the state of the RMS cluster and may have a direct effect on the disposition of data.

As stated in the overview of this chapter, the primary means of administration is through the Cluster Admin GUI. This method should be used whenever possible. However, each procedure in this section includes a CLI alternative.

7.1.1 Starting RMS

When you use the GUI, you can only start the most recently activated configuration. To start a different configuration, you must first use the Wizard Tools to activate that configuration.

By default, the GUI will start RMS on all the nodes in the cluster. Alternatively, you can start RMS only on a single node that you select.

► From the Cluster Admin *rms&pcs* (or *rms*) tab, select *Tools –> Start RMS* (Figure 120).
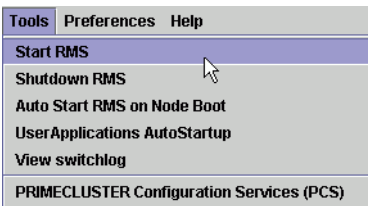
Figure 120: Starting RMS from the main menu

The *RMS Start Menu* window opens.

▶ To start RMS on all nodes, click the *all available nodes* radio button and then click *OK* (Figure 121).
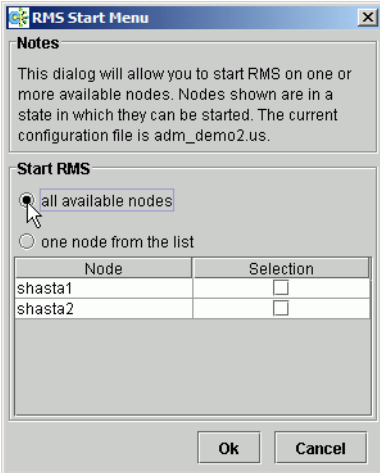
Figure 121: RMS Start Menu for all nodes

▶ To start RMS only on a single node, click the *one node from the list* radio button, and then choose the node using a checkbox in the *Selection* column. After making your selections, click *OK* (Figure 122).
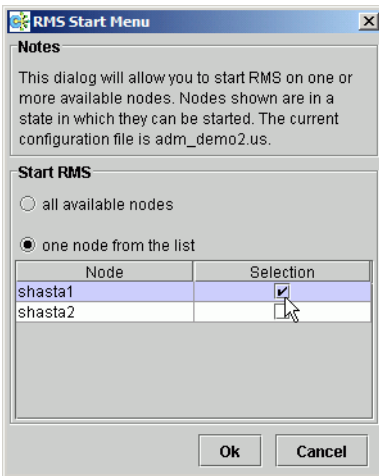
Figure 122: RMS Start Menu for individual nodes

Alternatively, you can start RMS on individual nodes directly from the *Cluster Admin* window:

1. In the left pane, click the *rms&pcs* tab to view the cluster tree.

2. Right-click on the node and select *Start RMS* from the pop-up menu (Figure 123).
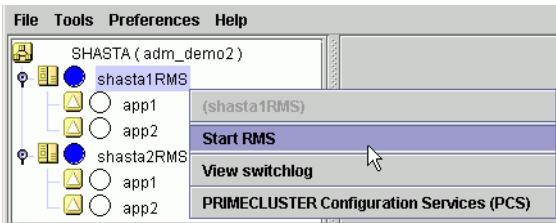
Figure 123: Starting RMS on individual nodes

CLI: hvcm

The hvcm command starts the base monitor and the detectors for all monitored resources. The CLI syntax for hvcm has two possible formats:

Format 1

> hvcm [−a | −s *SysNode*]

Format 2

> hvcm −c *config_name* [−a | −s *SysNode*] [−h *timeout*] [−l *loglevels*]

The options valid for both formats are:
- −a   Start RMS on all nodes in the configuration
- −s   Start RMS only on the specified node

If neither '−a' nor '−s' is specified, hvcm starts RMS only on the local node.

The options valid only for *Format 2* are:
- −c   Use the specified configuration file
- −h   Use the specified heartbeat recovery timeout
- −l   Activate diagnostic output according to the specified level(s)

> **i**   To start a configuration other than the one most recently activated, change the heartbeat recovery timeout, or set the diagnostic level, you must use the CLI. These parameters cannot be adjusted from the Cluster Admin GUI.

Notes for Format 1

When the '−c' option is not present, hvcm reads the default CONFIG.rms startup file. hvcm looks for the default startup file in *<RELIANT_PATH>*/etc/CONFIG.rms. If the default for the environment variable RELIANT_PATH has not been changed, this resolves to /opt/SMAW/SMAWRrms/etc/CONFIG.rms. Note that the search is always confined to the local node, even if you specify the '−a' or '−s' option to start RMS remotely.

The CONFIG.rms file contains either of the following:

- A simple configuration name, optionally with a '.us' suffix.

- An hvcm command compliant with *Format 2* that starts the most recently activated configuration.

You cannot specify the '−h' or '−l' options in *Format 1*. You can, however, edit the CONFIG.rms file and insert the options to comply with *Format 2*.

Notes for Format 2

When the '−c' option is present, and the configuration file is not an absolute path, hvcm looks for the first match in <*RELIANT_STARTUP_PATH*>. If the default for the environment variable RELIANT_STARTUP_PATH has not been changed, this resolves to /opt/SMAW/SMAWRrms/build/<*config_name*>.us (hvcm adds the '.us' extension if it is not specified as part of the configuration file name). If an absolute path is specified, hvcm attempts to read only that file. Note that the search is always confined to the local node, even if you specify the '−a' or '−s' option to start RMS remotely.

When hvcm locates the specified configuration file, it checks to see if the default CONFIG.rms file is also present. If it is, hvcm compares the configuration names defined by both files to make sure they agree. If they do not agree, hvcm aborts the startup processing.

The '−h' option sets the UDP heartbeat recovery timeout for the cluster. See "Nodes and heartbeats". The default timeout depends on which method RMS uses to monitor the cluster:

● By default, RMS monitors the state of each node with the Enhanced Lock Manager (ELM), which is implemented in the Cluster Foundation. ELM is not a polling method. Instead, it relies on locks that are held by each node and released when the node or its base monitor go down. When ELM is enabled, the UDP heartbeat timeout defaults to 600 seconds.

● ELM can be disabled for rolling upgrade or debugging operations by setting the HV_USE_ELM environment variable to zero (0). See "HV_USE_ELM". When ELM is disabled, the UDP heartbeat defaults to 45 seconds.

> **i** Specifying a heartbeat timeout shorter than the default may cause premature node kills. No data loss will occur, because a node kill begins with a graceful shutdown. However, cluster performance may suffer due to latency as applications are switched to different nodes. Excessively short heartbeat timeouts may also interfere with CF event timeouts.

The '−l' option sets the diagnostic output level at startup time. The *loglevels* specification consists of one or more individual numeric levels or hyphen-delimited ranges, each separated by a comma. For more information about this expert-level feature, see the *RMS Reference Guide*

7.1.2 Starting RMS automatically at boot time

i   This setting takes effect at the next system startup

You can use the following procedure to activate or deactivate automatic RMS startup when the system boots up.

► From the Cluster Admin *rms&pcs* (or *rms*) tabbed view, select *Tools > Auto Start RMS on Node Boot* (Figure 124).
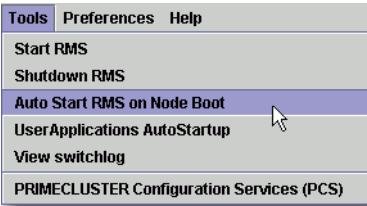
Figure 124: Controlling automatic RMS startup—step 1

You can then choose to activate (or deactivate) the automatic RMS startup on all nodes, or on just one node (Figure 125).
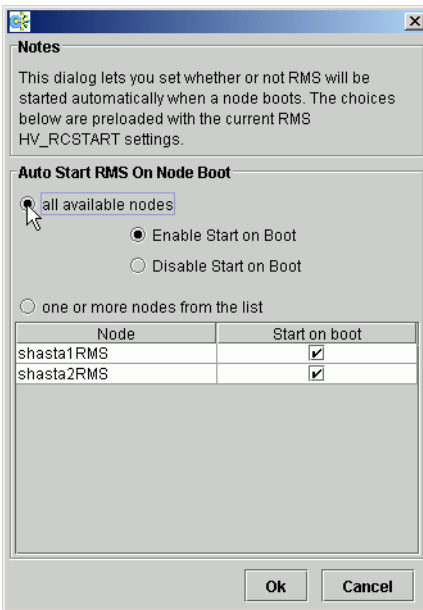
Figure 125: Controlling automatic RMS startup—step 2

CLI: hvsetenv

At system startup, the RMS `rc` script checks the environment variable settings: if the `HV_RCSTART` environment variable is set to 1, the `rc` script will attempt to start RMS using the `CONFIG.rms` file. You can set the `HV_RCSTART` variable with the `hvsetenv` command as follows:

```
hvsetenv HV_RCSTART [0|1]
```

The allowable values are:

    0   Do not start RMS at boot time
    1   Start RMS at boot time (default)

If no value is specified, the command reports the current value of the `HV_RCSTART` environment variable.

7.1.3   Stopping RMS

You can stop RMS on all nodes or on a subset that you select.

Use the *Tools* pull-down menu (Figure 126) and select *Shutdown RMS*.
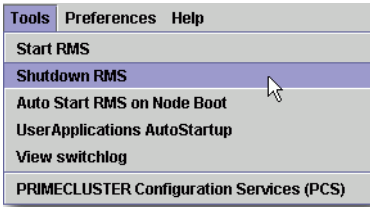
Figure 126: Using the Tools menu to stop RMS

▶ To stop RMS on all nodes, click the radio button for *all available nodes* and then click *Ok* (Figure 127).
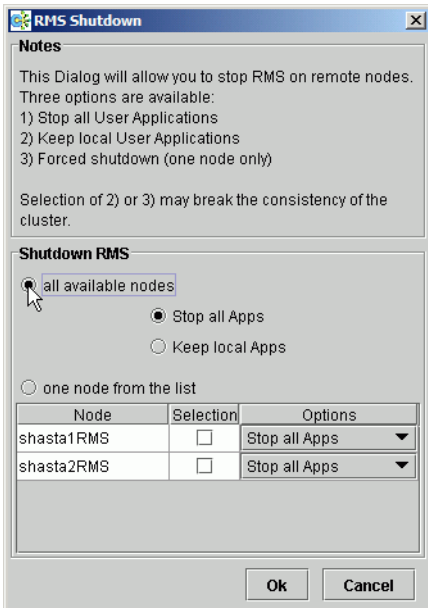
Figure 127: Stopping RMS on all available nodes

When you shut down *all available nodes*, two radio buttons allow you to choose how you want to handle the applications:

● *Stop all Apps*—Stops all user applications

● *Keep local Apps*—Leaves the applications running

⚠ **Caution**

Leaving the applications running after stopping RMS can lead to data inconsistencies or corruption.

To stop RMS on one specific node, select the radio button for *one node from the list*, and then click the checkbox of the node you want to shut down (Figure 128).
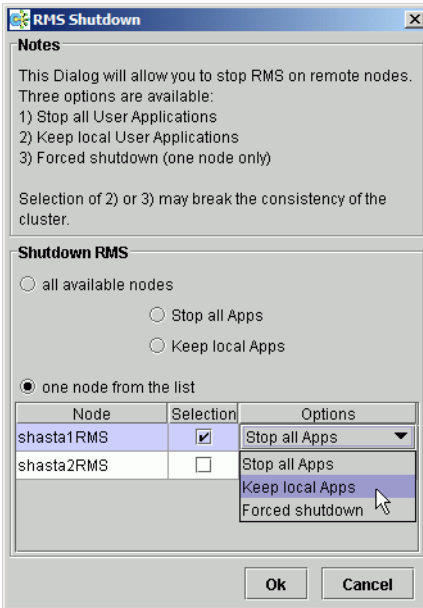
Figure 128: Stopping RMS on one node from the list

Each node has a dropdown list in the *Options* column to provide additional control:

● *Stop all Apps*—Stops all user applications on the selected node

● *Keep local Apps*—Leaves the applications running on the selected node

● *Forced shutdown*—Performs a forced shutdown of RMS

⚠ **Caution**

– Leaving the applications running after stopping RMS or using a forced shutdown can cause data inconsistencies or corruption.

- Do not stop RMS while RMS is running. Heartbeats between nodes are interrupted and the node where RMS is stopped may be forcibly shut down.

► Click the *Ok* button to initiate the shutdown with your selections.

Stop all Apps is the default option for shutting down RMS on all nodes or on one node. If you select an option other than the default, you will be prompted to confirm the operation (Figure 129 and Figure 130).
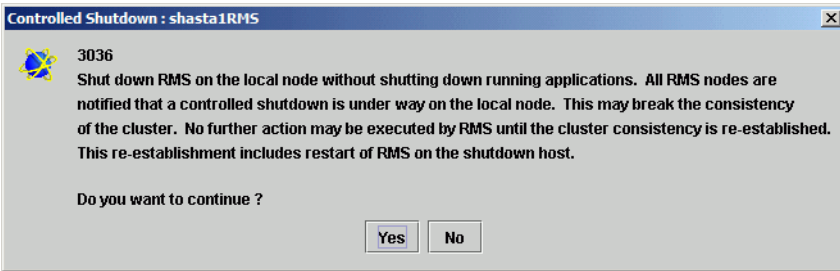
Figure 129: Stopping RMS while keeping applications—confirmation
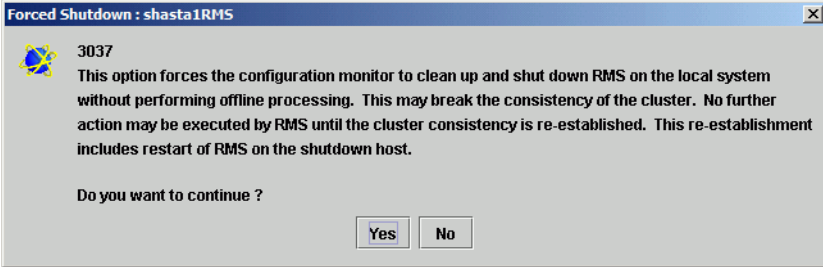
Figure 130: Forced shutdown of RMS—confirmation

You can also stop RMS on a single node by right-clicking on the node in the RMS tree and then selecting *Shutdown RMS* from the context menu (Figure 131).
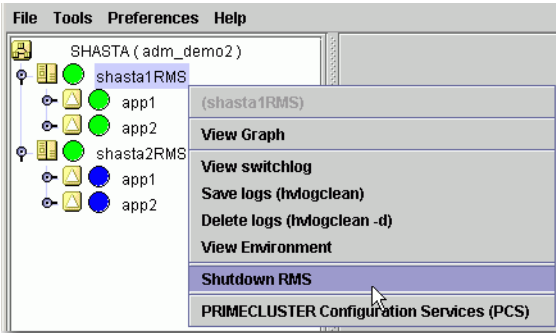
Figure 131: Using the context menu to stop RMS on one node

Only one node will appear in the confirmation window (Figure 132).
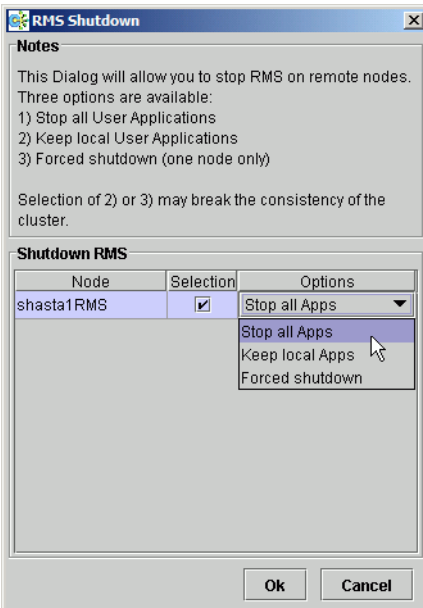
Figure 132: Stopping RMS on one node

CLI: hvshut

The syntax for the CLI is as follows:

    hvshut {-a | -A | -f | -l | -L | -s *SysNode*}

Options:
- −a   Shut down RMS and applications on all nodes
- −A   Shut down RMS on all nodes without shutting down applications
- −f   Forced (emergency) shutdown of RMS on the local node
- −l   Shut down RMS and applications on the local node
- −L   Shut down RMS on the local node without shutting down applications
- −s   Shut down RMS only on the specified node

The hvshut command shuts down the RMS software on one or more nodes in the configuration. The base monitor on the local node sends a message to other online nodes indicating which node or nodes are to be shut down. The hvshut command disables all error detection and recovery routines on the nodes being shut down, but does not shut down the operating system.

If any userApplication objects are online when the −A, −f, or −L options are used, the applications remain running but are no longer monitored by RMS. Both The −f and −L options affect only the local node, but the −f option is for emergencies (when other hvshut options do not work).

When you choose to shut down RMS without shutting down the monitored applications, you will be prompted to confirm the operation.

⚠ **Caution**

Use the hvshut −A, −f, and −L options carefully as they could result in inconsistencies or data corruption.

7.1.4   Clearing a SysNode Wait state

A node is put in the Wait state when the Shutdown Facility (SF) attempted to kill the node but could not successfully complete the shutdown within the timeout period. When you manually clear a wait state, RMS marks the specified node as being in either the online or offline state according to your selection, regardless of its state before the entering the wait state.

⚠ **Caution**

When you clear a wait state, be sure that the target state corresponds to the actual, physical state of the node. RMS, CF, and SF mark the node as being in the confirmed state you select. If the node is not truly in that state, conflicts and data corruption may occur.

If the Shutdown Facility (SF) timeout for a node has not yet been exceeded, requests to clear a wait state for that node will be ignored.

Clear any node in the `Wait` state as follows:

▶ Right-click on the node and select the *Online* or *Offline* option from the context menu.

CLI: hvutil -o and hvutil -u

Use the `hvutil` command to clear a wait state. To put the node in the online state, use the $-o$ option:

    hvutil -o *SysNode*

To put the node in the offline state, use the $-u$ option:

    hvutil -u *SysNode*

7.2   Managing RMS applications

This section describes basic procedures related to starting, stopping, and clearing special states of individual applications. Procedures in this section are active: they change the state of the RMS cluster and may have a direct effect on the disposition of data.

As stated in the overview of this chapter, the primary means of administration is through the Cluster Admin GUI. This method should be used whenever possible. However, each procedure in this section includes a CLI alternative.

7.2.1   Overriding automatic application startup

By default, the automatic startup of each application is controlled by its `AutoStartUp` attribute, which is typically set in the Wizard Tools configuration interface:

● If `AutoStartUp` is set to 1, the application starts automatically when RMS starts or when the application is switched to another node.

● If `AutoStartUp` is set to 0, the application must be started manually as described in "Starting RMS".

Automatic application startup can cause problems during some maintenance or troubleshooting procedures. If this is the case, you can suppress the `AutoStartUp` attribute for all applications.

> **i**   The following procedure changes the `HV_AUTOSTARTUP` global environment variable. Changes to `HV_AUTOSTARTUP` do not take effect until the next RMS startup.

► From the Cluster Admin *rms&pcs* (or *rms*) tab, select *Tools –> UserApplications AutoStartup* (Figure 133).
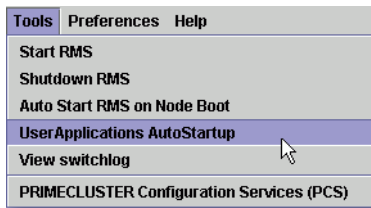
Figure 133: Controlling automatic application startup—step 1

You can then choose to override all `AutoStartUp` settings, or to cancel the override (Figure 134).
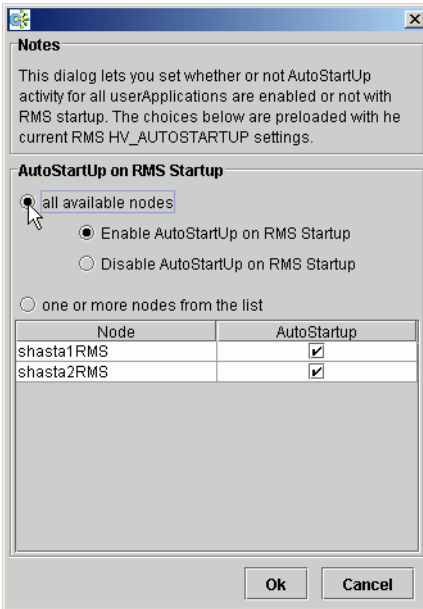
Figure 134: Controlling automatic application startup—step 2

CLI: hvsetenv

The action of each application's `AutoStartUp` attribute is controlled by the `HV_AUTOSTARTUP` environment variable (see the description in "Local environment variables"). You can set this variable with the `hvsetenv` command as follows:

```
hvsetenv HV_AUTOSTARTUP [0|1]
```

The allowable values are:

> 0 Prevent automatic application startup at next RMS startup

> 1 Allow automatic application startup at next RMS startup

If no value is specified, the command reports the current value of the `HV_AUTOSTARTUP` environment variable.

7.2.2   Switching an application

When you switch an application, RMS performs the following tasks:

● If the application is already running in the cluster, RMS shuts it down.

● After the application is completely shut down, RMS starts it on the node you specified.

Switch an application to any node as follows:

▶ Right-click on the application object on any node and select *Switch* from the context menu. A secondary menu appears, listing the available target nodes for switchover.

▶ Select the target from the secondary menu to switch the application to that node (Figure 135).
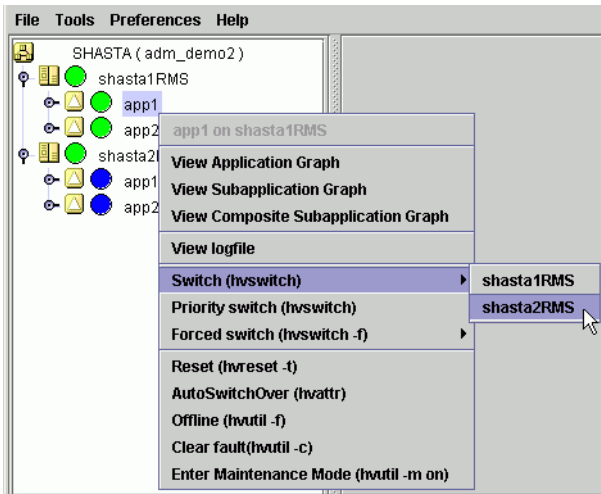
Figure 135: Switching an application

You will be prompted to confirm the action before RMS begins the operation.

A *Priority switch* switches the application to the target node according to the order of nodes in the application's PriorityList attribute.

If stopping the application fails, RMS does not start the application. This is because two competing instances of the same application could cause data corruption. In this event, you can start the application by using the *Forced switch* operation.

> ⚠ **Caution**
>
> Use the *Forced switch* mode only if an application cannot be switched normally. A forced application switch overrides all safety checks and could therefore result in data corruption or other inconsistencies. In PRIMECLUSTER for Solaris, RMS may kill the node on which RMS is not running before starting the application to reduce the risk of data corruption when the *Forced switch* request of an application is issued.

If the application is busy, the pop-up context menu will not offer the choices to switch the application. Instead, the menu will offer view-only operations, and the last menu item will indicate that the application is in a Wait state (Figure 136).
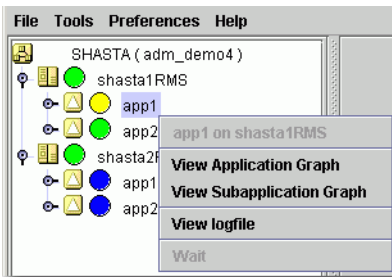
Figure 136: Switching a busy application

CLI: hvswitch

The syntax for the CLI is as follows:

    hvswitch [-f] *userApplication* [*SysNode*]

The hvswitch command manually switches control of a userApplication resource from one system node to another in the RMS configuration. The resource being switched must be of type userApplication. The system node must be of type SysNode. If no SysNode is specified, the application is switched to the local node. The -f option is a forced-switch option.

⚠ **Caution**

Use the 'hvswitch -f' operation carefully. A forced application switch overrides all safety checks and could therefore result in data corruption or other inconsistencies.
In PRIMECLUSTER for Solaris, RMS may kill the node on which RMS is not running before starting the application to reduce the risk of data corruption when the *Forced switch* request of an application is issued.

7.2.2.1   Notes on forcibly switching an application

When *Forced switch* request (forced startup) of an application is issued, RMS overrides all safety checks and starts the application. So if shared resources which require exclusive control between nodes become Online on the multiple nodes simultaneously, it could result in data corruption or other inconsistencies.

⚠ **Caution**

In PRIMECLUSTER for Solaris, RMS may kill the node on which RMS is not running before starting the application to reduce the risk of data corruption when the *Forced switch* request of an application is issued.

To perform forced startup of an application safely, check whether RMS is running on all nodes in the cluster before starting forced startup according to the following procedure, and if there are the nodes on which RMS is not running, then shut down the nodes.

▶ Check the node state by one of the following methods:

● Execute the cftool -n command on all nodes.

```
shasta1# cftool -n
Node      Number    State    Os       Cpu
shasta1   1         UP       Linux    Pentium
shasta2   2         UP       Linux    Pentium
```

● Check the CF tree of the Cluster Admin.
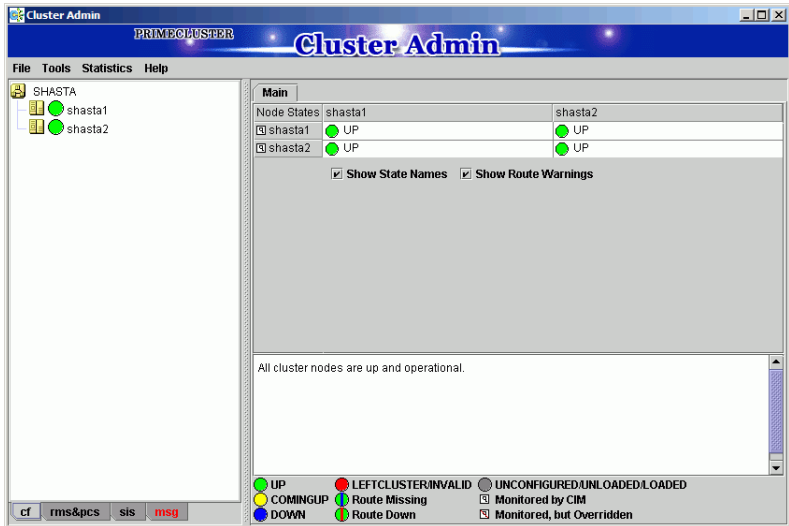
Figure 137: Main Cluster Admin window—CF view

▶ Check the following contents for the node states, and take corrective actions if necessary:

  ● Check the node states are all UP.

  ● If a LEFTCLUSTER node exists, recover CF from the LEFTCLUSTER state.

    For details, see "PRIMECLUSTER Cluster Foundation (CF) Configuration and Administration Guide."

  ● If a node with DOWN or UNKNOWN exists, or if a node for which the state is not displayed exists, check whether the operating system of the node has stopped. If the operating system is running, shut down the operating system or restart the system in single-user mode.

► Check whether there are no nodes on which RMS is not running among the nodes on which cluster applications will be forcibly started by one of the following methods:

● Execute the hvdisp −a command on nodes where cluster applications will be started and check that the state of objects whose Type is SysNode is Online.

```
shasta1# hvdisp −a
Local System:    shasta1RMS
Configuration:   /opt/SMAW/SMAWRrms/build/config.us
Resource              Type      HostName     State    StateDetails

─────────────────────────────────────────────────────────────
shasta2RMS            SysNode                Online
shasta1RMS            SysNode                Online
app2                 userApp                 Offline
Machine001_app2       andOp    shasta2RMS
Machine000_app2       andOp    shasta1RMS    Offline
ManageProgram000_Cmd_APP2 gRes               Offline
app1                 userApp                 Offline
Machine001_app1       andOp    shasta2RMS
Machine000_app1       andOp    shasta1RMS    Offline
ManageProgram000_Cmd_APP1 gRes               Offline
```

● Check that the state of all `SysNode` displayed in the RMS tree of the Cluster Admin is Online.
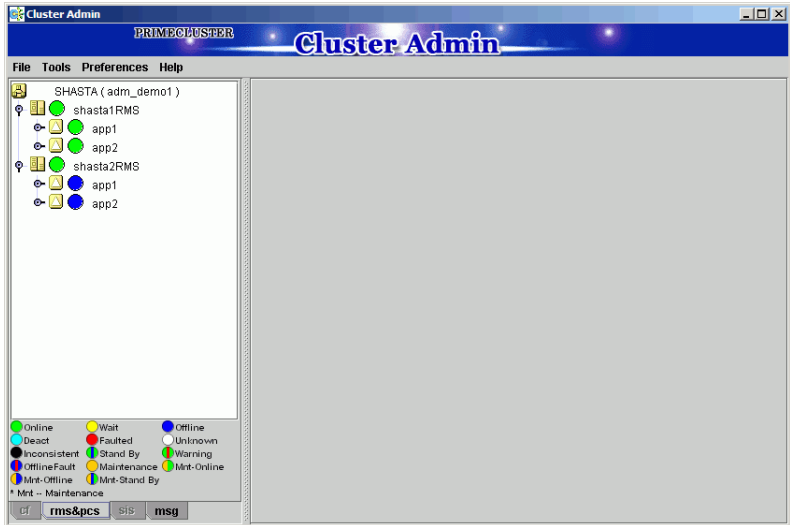
Figure 138: Main Cluster Admin window—RMS view

▶ If nodes which satisfy the following conditions exist, shut down the operating system of the nodes, or restart the system in single-user mode.

● The node state is UP, and

● The state of `SysNode` is not Online.

▶ Execute the *Forced switch* (`hvswitch -f`) to forcibly start the application.

7.2.3   Starting an application

If the application is already offline everywhere in the cluster, you can start it
(bring it online) on a single node as follows:

► Right-click on the application object and select *Online* from the pop-up
context menu (Figure 139).
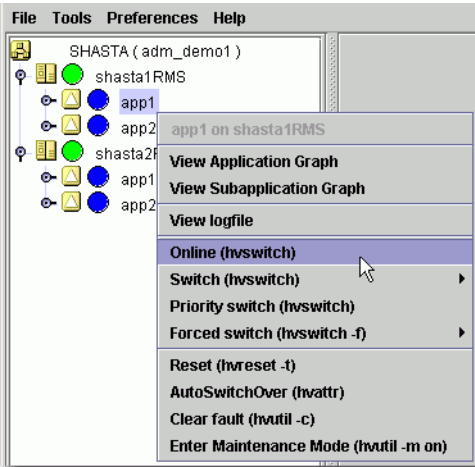
Figure 139: Starting an application

You will be prompted to confirm the action before RMS begins the operation.

CLI: hvswitch

Starting an application on the local node, like switching an application to another
node, employs the hvswitch command. Refer to "CLI: hvswitch" for the syntax.

7.2.4   **Stopping an application**

Stop an online application (take it offline) as follows:

▶ Right-click on the online application object and select *Offline* from the pop-
   up context menu (Figure 140).
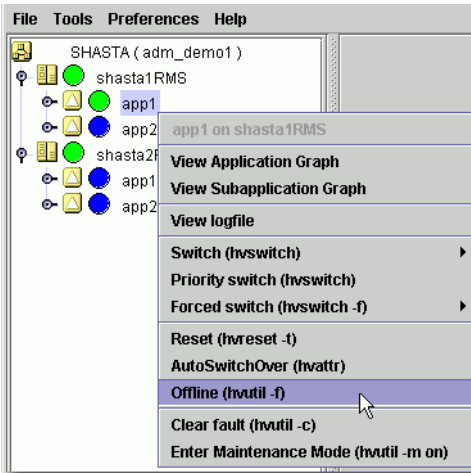
Figure 140: Shutting down an application

You will be prompted to confirm the action before RMS begins the operation.

CLI: hvutil -f

The syntax for the CLI is as follows:

```
hvutil -f userApplication
```

Note that this is a **normal offline request**. There is no "forced" offline request
for a userApplication.

> ℹ Use the command 'hvutil -s *userApplication*' to bring an offline
> userApplication to a Standby state.

7.2.5   Resetting an application

This operation completely re-initializes the target application and all objects in its resource tree (*i.e.*, the entire application graph) based on the actual detector reports. It will interrupt any ongoing RMS processing for the application, and any running scripts will be terminated.

⚠️   **Caution**

Resetting an application will cause information about previous failures or any other history to be lost. It will most likely result in an *Inconsistent* state for the application.

This operation is intended for use by an experienced administrator during a test phase. It should never been invoked in a production environment.

Reset (reinitialize) an online application as follows:

► Right-click on the online application object and select *Reset* from the pop-up context menu (Figure 141).
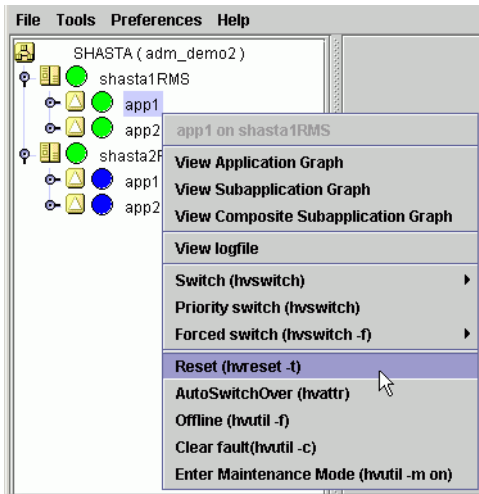
Figure 141: Resetting an application

You will be prompted to enter the reset timeout before RMS begins the operation (Figure 142).
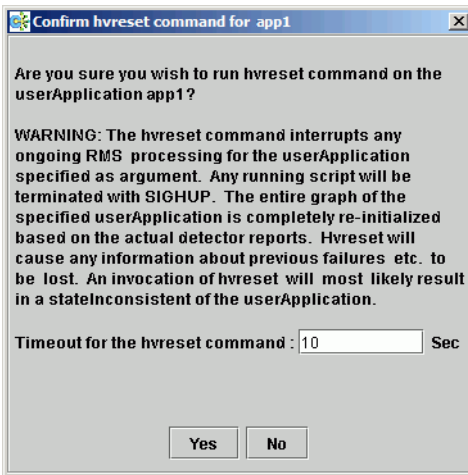
Figure 142: Choosing the reset timeout for the application

If RMS fails to re-initialize the entire application graph during this amount of time, it terminates with an error message. Default: 10 seconds

► Adjust the timeout value if desired, and then click *Yes* to reset the application.

CLI: hvreset -t

The syntax for the CLI is as follows:

    hvreset −t *timeout userApplication*

hvreset displays a message that warns of the consequences and then prompts you to confirm the action before proceeding.

When the userApplication is not performing online, offline, or fault processing, hvreset returns immediately and displays a message stating there is nothing to do. This case is considered to be successful command execution and returns exit code 0.

7.2.6   Changing AutoSwitchOver setting for an application

An application's *AutoSwitchOver* attribute specifies whether or not it should be automatically switched to another node in the event of a resource failure, a node failure, or a node shutdown. This attribute is set for each application at configuration time, but you can change it at runtime with the following procedure:

► Right-click on the application object on any node and select *AutoSwitchOver* in the context menu (Figure 143).
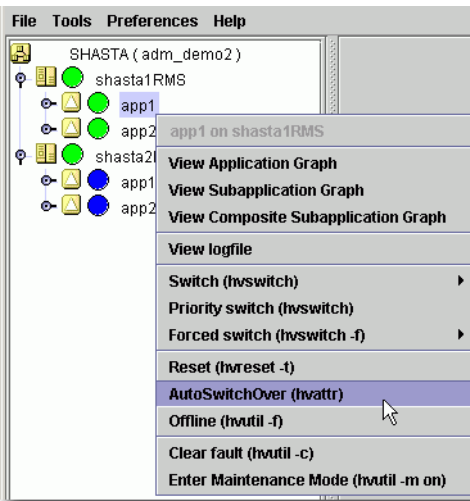
Figure 143: Changing an application's AutoSwitchOver settings—step 1

► Select the desired settings from the secondary pop-up window (Figure 144).
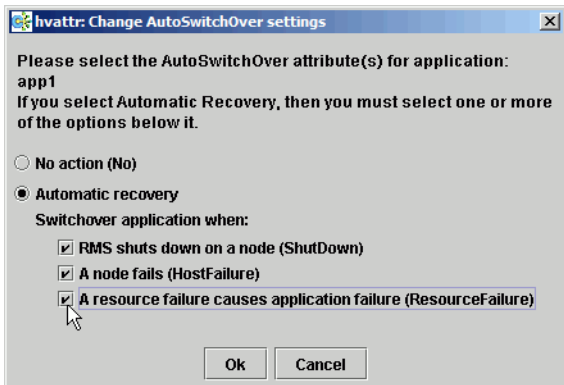
Figure 144: Changing an application's AutoSwitchOver settings—step 2

If you choose *Automatic recovery*, you can select or deselect one or more of the indicated conditions by clicking the appropriate checkboxes.

► Click *Ok* to confirm the *AutoSwitchOver* change on all nodes.

As soon as you click *Ok*, RMS starts updating the application's *AutoSwitchOver* setting throughout the cluster. The operation typically requires only a few seconds to update each node, but it may take longer in some unusual cases. There is no notification at either the beginning or the end of the process.

If some nodes are unreachable, RMS does not issue a warning. Instead, if one of the unreachable nodes becomes available at a later time, RMS automatically updates the application's *AutoSwitchOver* setting on that node as soon as the base monitor starts up. This guarantees that the *AutoSwitchOver* setting is consistent on every running node in the application's priority list.

CLI: hvattr

The syntax for the CLI is as follows:

    hvattr *<userApplication>*:AutoSwitchOver:*<value>* ...

The hvattr command is designed as a general-purpose command to perform clusterwide modifications of attributes at the application level. Each argument on the command line specifies a colon-separated triad of the application name, the attribute to modify, and the new value to be applied. You can supply multiple arguments on one command line, where each argument specifies a unique combination of application and attribute names.

Currently, only the *AutoSwitchOver* attribute is available to the user. For its allowable values, see "AutoSwitchOver".

i	When specifying an attribute value that contains the pipe character, the attribute value must be surrounded in quotes to prevent the shell from interpreting the pipe character itself.

You should avoid issuing multiple hvattr commands simultaneously. Normally, concurrent calls to hvattr will be gracefully rejected. In rare circum-stances, hvattr may not complete until timeout occurs, in which case you must wait until the running command exits before issuing another hvattr command.

7.2.7   Clearing a fault

For an application that is in the Faulted state, clear the fault as follows:

► Right-click on the application object and select *Clear Fault* from the pop-up context menu (Figure 145).
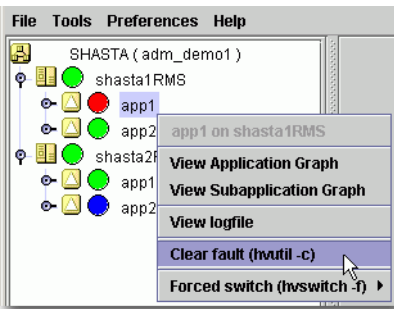
Figure 145: Clearing an application fault

Before the command proceeds, you will be informed of the action to be taken and prompted to confirm the operation (Figure 146).
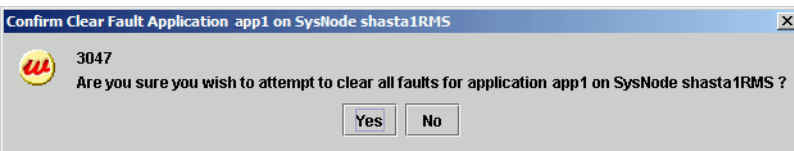
Figure 146: Clearing an application fault—confirmation dialog

CLI: hvutil -c

The syntax for the CLI is as follows:

    hvutil -c *userApplication*

> [i] If the `userApplication` is in the online state, then clearing the fault will
> cause RMS to attempt to bring the faulted resource to the online state. If
> the `userApplication` is in the offline or faulted state, then clearing the
> fault will attempt to bring the resource to the offline state. You will be
> informed of the action that is to be taken and prompted to confirm the
> operation before the command proceeds.

7.2.8   Activating an application

Activating an application takes it from the `Deact` state to the `Offline` state. It
does not bring it `Online`. Also, activating a `userApplication` with the Cluster
Admin GUI or the CLI has nothing to do with activating an RMS configuration in
the Wizard Tools—the two operations are completely independent. Activate a
deactivated application as follows:

▶ Right-click on the application object and select the *Activate* option from the
   pop-up menu.

CLI: hvutil -a

The syntax for the CLI is as follows:

    hvutil -a *userApplication*

> [i] You will not need to activate an application unless someone explicitly
> deactivated it with the command '`hvutil -d` *userApplication*'.

7.3   Using maintenance mode

Maintenance mode is a special mode of operation that allows an application to
be temporarily decoupled from its dependent resources. This allows, for
example, a file system to be taken offline for backup purposes without disrupting
the online state of its parent application.

7.3.1 Entering maintenance mode

You can enter maintenance mode for all applications on all nodes as follows:

▶ Right-click on the cluster at the top of the RMS tree and select *Enter Maintenance Mode* from the popup menu (Figure 147).
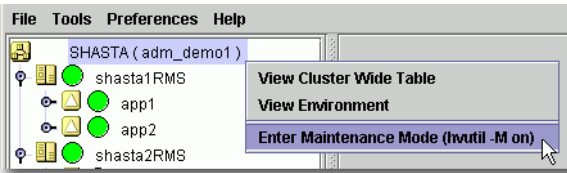
Figure 147: Starting maintenance mode for all applications

Enter maintenance mode for only one application as follows:

▶ Right-click on an application instance in the RMS tree and select *Enter Maintenance Mode* from the popup menu (Figure 148).
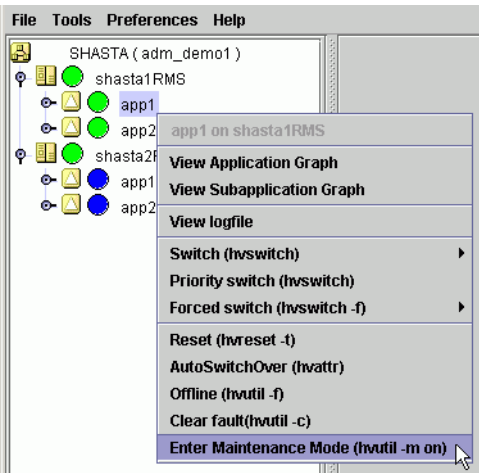
Figure 148: Starting maintenance mode for a single application

In either case, you will be prompted to confirm the operation (Figure 149 and Figure 150).
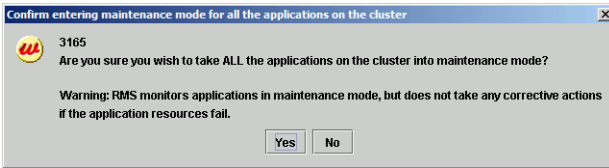
Figure 149: Maintenance mode confirmation for all applications
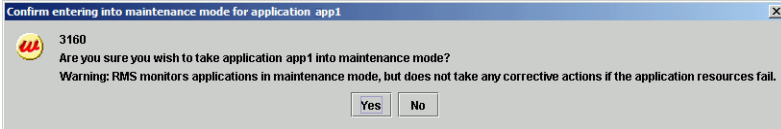
Figure 150: Maintenance mode confirmation for one application

> **i**   Maintenance mode is clusterwide: if an application is in maintenance mode on one node, it is also in maintenance mode on any other node where it can run.
> The application is allowed to start in the following states.
>
> – Online state
>
> – Standby state
>
> – Offline state
>
> – Warning state
>
> – Inconsistent state (If an application state just before it becomes inconsistent is Faulted state, it is excluded.)
>
> The maintenance mode is rejected if there is any request of dynamic reconfiguration of RMS graph.

Figure 151 shows the Cluster Admin window after one application is put into maintenance mode.
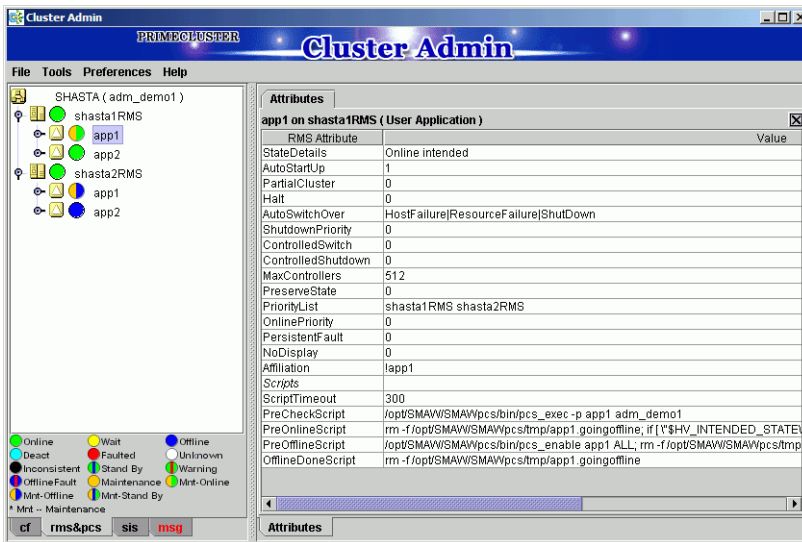
Figure 151: Typical cluster in maintenance mode

Note how the right half of the application status icons indicate the intended state (the state that would be attained if the application were taken out of maintenance mode). The intended state is also indicated by the application's *StateDetails* attribute, which is the first item in the *Attributes* table in the right pane.

7.3.2   Maintenance mode operating notes

When an application enters maintenance mode (MM), it affects all other applications that **share the same graph**. For example, if two applications are linked by a controller, then putting one in maintenance mode will cause the other to go into maintenance mode as well; which one is the parent and which one is the child does not matter in this case.

Conversely, if two applications do not share the same graph, *i.e.*, they are not linked by one or more controllers, then one can be put into MM while the other operates under normal RMS control.

For instance, in the example presented earlier, `app1` and `app2` are independent. While `app1` is in MM, `app2` continues to operate normally and can be switched from one node to the other, as illustrated in Figure 152 below.
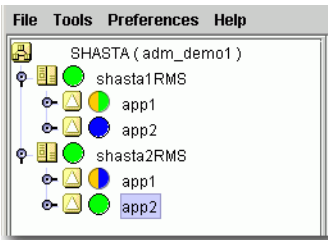
Figure 152: Normal operation of independent application

7.3.2.1 Overall cluster restrictions in maintenance mode

Even though some applications may continue to operate under normal RMS control, MM still places restrictions on the overall cluster operation. In particular, note the following:

● You must exit MM for an application before you can switch that application offline or to another node.

● You must exit MM everywhere in the cluster before you can shut down RMS.

7.3.3 Exiting maintenance mode

To exit maintenance mode, use the following procedure:

► Right-click on the cluster or an application and select *Exit Maintenance Mode* from the popup menu (Figure 153 and Figure 154).
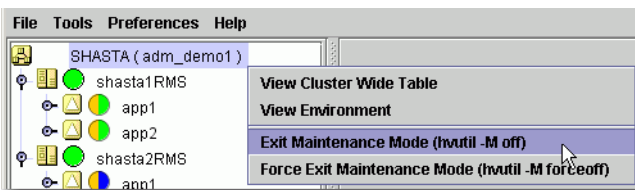
Figure 153: Normal maintenance mode exit for all applications
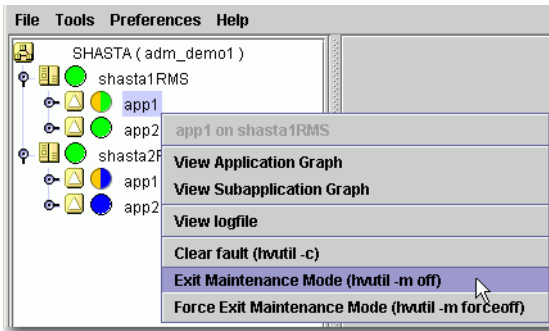
Figure 154: Normal maintenance mode exit for a single application

In either case, you will be prompted to confirm your action before the operation
proceeds.

| **i** | You can exit maintenance mode for a single application even if you entered maintenance mode for the entire cluster. |

To exit maintenance mode, the following conditions must be met:

– If you start maintenance mode when the application status is either
   Online, Offline, Standby, or Warning, statuses of the application and
   each resource must be the same as those before maintenance mode
   is started in order to exit maintenance mode. If the status is different,
   you cannot exit maintenance mode.

– If you start maintenance mode when the application status is Incon-
   sistent, remove causes which put the application into the Inconsistent
   state. You cannot exit maintenance mode until causes are removed.

For example, in the above example, there is a file system resource Res1 in
the application app1, and Res1 is in the online state on fuji2RMS when
starting the maintenance mode in app1. In this case, Res1 must be in the
online state on fuji2RMS to exit the maintenance mode of app1.

If you forcibly exit the maintenance mode by using [Force Exit Maintenance
Mode] in a different state from when starting the maintenance mode, the
application enters the Inconsistent state. Therefore, we do not
recommend using [Force Exit Maintenance Mode].

Note that both the cluster and application popup menus shown above contain a
Force Exit Maintenance Mode item. If you choose this command, it will force RMS
to exit maintenance mode even if some resources are not in the appropriate
state. The prompt to confirm the operation for one application is shown in
Figure 155; the prompt for all applications is similar.
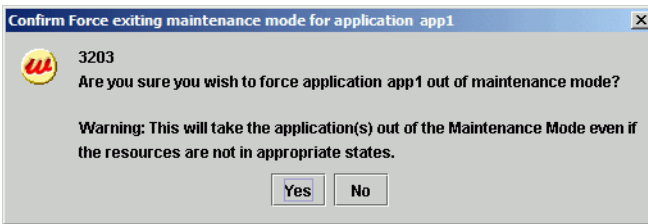
Figure 155: Forced maintenance mode confirmation for all applications

When you exit maintenance mode, the processing corresponding to the userApplication status is executed (the Online processing is for the Online state and the Offline processing is for the Offline state).

In these operations, if resource statuses under userApplication are the same as those before maintenance mode is started, the Online/Offline script is not executed and the resource statuses are not changed.

However, if the Online processing is executed for Cmdline resource to which the NULLDETECTOR attribute is set, the Online script is definitely executed despite the actual resource statuses.

7.3.4   Maintenance mode CLI: hvutil -m and -M

Control maintenance mode (MM) with the `hvutil` command:

```
hvutil -M { on | off | forceoff }

hvutil -m { on | off | forceoff } userApplication
```

Options:

-M	Applies the MM operation to all applications on all nodes
-m	Applies the MM operation to the specified application on all nodes

Operations:

on	Starts maintenance mode
off	Stops MM if all resources are in the appropriate state
forceoff	Forces MM to stop even if all resources are not in the appropriate state

The `hvutil` maintenance mode commands operate synchronously, so they do not return until the final state has been reached or until an error occurs. In the particular case where '`−m off`' returns a failure because one or more resources were in an inappropriate state, an error message is displayed that lists the problem resources.

8 Appendix—Site preparation

The *Software Release Guide PRIMECLUSTER* and the *PRIMECLUSTER Installation Guide* for your operating system describes how to prepare your cluster to operate RMS. Some of the procedures require you to modify system files so that RMS can identify the hosts, file systems, and network interfaces used in a configuration. You should have completed these procedures when RMS was installed.

In some cases, you will be creating or modifying your RMS configuration because changes have been made to your site. Certain site changes may require you to review and update your system files first. These changes include, but are not limited to, the following:

● IP addresses were changed.

● Redundant interconnects were added to the cluster.

● Hosts were added, removed, or renamed.

● Two or more clusters were merged into one.

● File systems or SANs were added or removed.

For convenience, the site preparation descriptions for hosts, file systems, and networks are duplicated here. If any of these specifications have changed since your initial RMS installation, you should review this material and make the necessary adjustments before proceeding with your RMS configuration.

The modifications generally involve adding RMS-specific entries to standard system files; pre-existing entries required for proper operation of your hosts and network are not affected. Resources for market-specific applications may require similar customization.

8.1 Network database files

8.1.1 /etc/hosts

The `/etc/hosts` file must contain the IP addresses and RMS names of all the host systems that are part of the cluster.

RMS uses its own internal set of host names to manage the machines in the cluster. When you configure the cluster, you will use the RMS host names and not the standard host names. These names must be entered in `/etc/hosts` on each system in the cluster to avoid problems should access to the DNS fail. If you used Cluster Admin to configure CIP for RMS, then `/etc/hosts` will already contain the correct RMS node names described below.

By default, the names follow the conventions in Table 5.

Entry type	RMS naming pattern	Examples
RMS host name	*<hostname>*RMS	`shasta1RMS` `shasta2RMS`

Table 5: RMS host name conventions in /etc/hosts

> **i** The RMS host name for a machine must match the contents of the `RELIANT_HOSTNAME` variable in that machine's `hvenv.local` configuration file, if that file exists.

RMS does not support IPV6 addresses.

Example

The following entries in `/etc/hosts` are for a cluster with nodes `shasta1` and `shasta2`. The interface names are assigned as follows:

– Standard host names on the public network 172.25.220

– RMS node names on the private network 192.168.10

```
172.25.220.83   shasta1
172.25.220.84   shasta2
# node names for RMS
192.168.10.83   shasta1RMS
192.168.10.84   shasta2RMS
```

8.1.1.1 Network interface names in /etc/hosts

If you plan to configure one or more network interfaces for switchover with the *Ip Address* subapplication, you must first enter the interface name(s) in the /etc/hosts file on every node where that interface can exist. Each entry consists of the interface IP address and its name in the normal format; no special comments are required.

Example

> If the interface shastavip with IP address 172.25.222.223 can be switched between nodes shasta1 and shasta2, then both nodes should contain the following line in /etc/hosts:

> `172.25.222.223    shastavip`

> | **i** | When you configure the *Ip Address* subapplication, you specify the interface name as it appears in /etc/hosts, and not the IP address. |
> |---|---|

8.1.2   /root/.rhosts (Linux) and /.rhosts (Solaris)

Contains entries to control trusted login from remote hosts.

The Wizard Tools require automatic login or authentication as root on every machine in the cluster. One method is to include the names of trusted hosts in the .rhosts file, which must be modified appropriately on each node. See the rhosts manual page for a complete description of the format.

Example

> If the cluster consists of hosts shasta1 and shasta2, then every machine's .rhosts file should contain the following lines:

> ```
> shasta1 root
> shasta2 root
> ```

> | **i** | The Cluster Foundation (CF) provides the equivalent of .rhosts functionality for all RMS configuration, administration, and operation tasks. |
> |---|---|

8.2   Configuration resource definitions

8.2.1   /opt/SMAW/SMAWRrms/etc/hvipalias

This file contains entries for all of the network interfaces that are to be used as resources in the configuration. Typically, each entry associates a logical interface name, or **IP alias**, with a physical interface on a specified node. The IP alias always presents the same IP address, even though it is switched from node to node, and even if the underlying physical interface has different characteristics on each node.

> **i**   Each IP alias with its IP address must be entered in the `/etc/hosts` file. The IP address does not appear in the `hvipalias` file.

Each entry in `hvipalias` must contain the following fields:

   <Uname>   <IfName>   <IfDevice>   <Netmask>

The fields are defined as follows:

● *Uname*—Name of the machine to host the logical interface. This is usually the value returned by the '`uname -n`' command.

   Alternatively, you can use the Cluster Foundation (CF) node name, which is returned by the '`cftool -ql`' command. This can be used to differentiate two machines which are configured with the same '`uname -n`' setting, provided they were assigned different names when CF was configured.

● *IfName*—Logical interface name, or IP alias. This name must appear with its associated IP address in the node's `/etc/hosts` file, and the associated IP address must be the same on every node.

● *IfDevice*—Physical device name to be associated with the logical interface when it is switched to the specified machine.

   If you specify two comma-separated device names, and the first device fails, the logical interface will failover to the second device.

● *Netmask*—The netmask to use with the IP address associated with the interface name, specified in the standard hexadecimal 8-digit format. The netmask is set with an `ifconfig` command after the physical device is brought online.

> **i**   To avoid network conflicts, a network hostname or address monitored by the *IpAddress* subapplication can be active on only one node in the cluster at any time.

When a configuration is activated, the local `hvipalias` file is copied to every other node in the configuration; the file with same name and path on the remote node will be overwritten with the new content. Therefore, the file must contain one line for every combination of interface and host where that interface can be switched.

For instance, if the interface named `dbhost` can be switched between two nodes, then `hvipalias` on each node should contain lines for both the local and remote interfaces:

Example (Linux)

```
#Uname    IfName  IfDevice   Netmask
shasta1   dbhost  eth1       0xffffff00
shasta2   dbhost  eth1       0xffffff00
```

Example (Solaris)

```
#Uname  IfName  IfDevice   Netmask
fuji2   dbhost  hme0       0xffffff00
fuji3   dbhost  hme0       0xffffff00
```

Example

```
#Uname    IfName  IfDevice   Netmask
shasta1   dbhost  eth1       0xffffff00
shasta2   dbhost  eth1       0xffffff00
```

If you specify the interface as *device1,device2* then the IP address will failover to the second device if the first device fails:

Example (Linux)

```
#Uname    IfName  IfDevice   Netmask
shasta1   dbhost  eth1,eth2  0xffffff00
shasta2   dbhost  eth1,eth2  0xffffff00
```

Example (Solaris)

```
#Uname  IfName  IfDevice   Netmask
fuji2   dbhost  hme0,hme1  0xffffff00
fuji3   dbhost  hme0,hme1  0xffffff00
```

Example

```
#Uname    IfName  IfDevice   Netmask
shasta1   dbhost  eth1,eth2  0xffffff00
shasta2   dbhost  eth1,eth2  0xffffff00
```

8.2.1.1   Optional fields

The following fields are optional. If specified, they must appear after the
required fields in the order presented here.

ifconfig parameters

You can specify a set of arguments to be sent to the `ifconfig` command. This
allows you to specify custom interface settings that may be required for a
physical device before the interface is switched there.

The field begins with the `IFCONFIG` keyword (all uppercase), followed by
whitespace, followed by the comma-delimited argument string that will be
passed to the `ifconfig` command. The keyword and arguments must appear
after the *Netmask* field and before any *Route* arguments.

For example, if an `mtu` value of 1200 is required for the local device associated
with the `dbhost` alias, the entry for the local node would be as follows:

Example (Linux)

```
#Uname    IfName  IfDevice   Netmask
shasta1   dbhost  eth1,eth2  0xffffff00  IFCONFIG mtu,1200
```

Example (Solaris)

```
#Uname   IfName  IfDevice    Netmask
fuji2    dbhost  hme0,hme1  0xffffff00  IFCONFIG mtu,1200
```

Example

```
#Uname    IfName  IfDevice   Netmask
shasta1   dbhost  eth1,eth2  0xffffff00  IFCONFIG mtu,1200
```

After an interface is successfully brought online, the `ifconfig` command with
the specified netmask and additional arguments will be invoked for the
associated device.

Route parameters

Remaining fields at the end of the line are passed to a `route` command for the
configured interface. If the string `$INTF` is encountered, it is replaced by the
interface name; otherwise, the fields are passed literally.

| i | Do not include the `add` or `delete` subcommands in the argument list. RMS generates these automatically when the interface is brought online or offline. |

Example (Linux)

```
shasta1  dbhost  eth1,eth2  0xffffff00  default dev $INTF
```

Example (Solaris)

```
fuji2  dbhost  hme0,hme1  0xffffff00  default dev $INTF
```

Example

```
shasta1  dbhost  eth1,eth2  0xffffff00  default dev $INTF
```

When the `dbhost` interface is brought online, RMS will issue the following command:

```
route add default dev dbhost
```

8.2.2 /opt/SMAW/SMAWRrms/etc/hvconsoles

Controls customized handling of fault messages, usually to remote consoles or special devices such as pagers. This does not affect the standard messages written to the RMS or system log files.

Each entry specifies a program to be executed when an RMS resource object encounters a fault.

If the file does not exist, you will receive no customized fault information. A complete description of the format is available in the comments in the `hvconsoles.template` file.

8.3   Linux file systems

To manage Linux file systems with RMS, you must create entries in the
/etc/fstab.pcl and /etc/exports.pcl configuration files as described in
the following sections. These configuration files share the following features:

- Leading whitespace and empty lines are ignored.

- A line beginning with the string '#RMS#' or '#RMS:*<appname>*#' is a file
  system specification for RMS. The entire specification must be on the same
  line—continuation to additional lines is not allowed.

- Any other line beginning with a pound sign (#), or any line beginning with an
  asterisk (*), is treated as a comment.

- All other lines are presently ignored. However, they may be processed by
  future versions of RMS.

8.3.1   /etc/fstab.pcl

This file contains entries for all of the local and remote file systems that are to
be used as resources in the configuration. RMS is responsible for mounting and
unmounting each of these file systems in order to bring them online or offline,
respectively, according to the requirements of the running configuration.

For each file system to be managed by RMS, create a line in /etc/fstab.pcl
with the standard fstab fields, and then insert the string #RMS# at the beginning
of the line. For more information, see the fstab manual page.

Note the following restrictions when you create /etc/fstab.pcl:

- Do not specify the same file system in both a standard /etc/fstab entry
  and an RMS /etc/fstab.pcl entry. The standard entry will mount the file
  system at system startup, and this will create a conflict when RMS starts up
  and attempts to mount the same file system.

- If a remote file system is specified in the form *<server_name>*:*<server_path>*,
  then *<server_name>* must be a host name that appears in the /etc/hosts
  file. It cannot be an IP address, and you should not rely on DNS to resolve
  the name.

Examples

```
#RMS#/dev/sdb2 /fs2     ext2     defaults 1 2
#RMS#/dev/sda1 /mnt/data1  auto     noauto,user  0 0
#RMS#/dev/sda2 /mnt/data2  auto     noauto,user  0 0
#RMS#boat:/opt/SUNWspro /opt/SUNWspro nfs \
          defaults,nfsvers=2,rsize=8192,wsize=8192
```

8.3.1.1　Configuring file systems for particular applications

If the RMS comment is of the form #RMS:*<appname>*#, the file system entry
applies only to the specified application. From an RMS perspective, file systems
assigned to a given application are independent of those assigned to other
applications. A file system can be assigned to two or more applications,
provided only one of the applications is online at any time.

Examples

```
#RMS:app1#/dev/sdb2  /data3  auto    noauto,user  0 0
#RMS:app2#/dev/sdb6  /data4  auto    noauto,user  0 0

#RMS:app1#boat:/var/tmp/test0 /var/tmp/nfs_app1  nfs \
          defaults,nfsvers=2,rsize=8192,wsize=8192
#RMS:app2#boat:/var/tmp/test1 /var/tmp/nfs_app2  nfs \
          defaults,nfsvers=2,rsize=8192,wsize=8192
```

8.3.1.2　Clusterwide configuration issues

In general, if you create an /etc/fstab.pcl control entry for a remote file
system or a shared filer on one node, then you should duplicate that entry on
every other node in the cluster, even if some nodes will not mount that file
system. This helps to ensure that the configuration behaves consistently
throughout the cluster.

Use a similar procedure for entries that specify local file systems and mount
points. If all nodes have the same architecture, you may be able to simply copy
the entire /etc/fstab.pcl control file. However, if the local physical disk
device differs from node to node, you must individually adjust the entries for the
same mount point. For example, the respective entries for /mnt1 on node1 and
node2 might be as follows:

node1:

```
#RMS#/dev/sda3   /mnt1  ...
```

node2:

```
#RMS#/dev/sdb5   /mnt1  ...
```

In all cases, for each mount point that appears in `/etc/fstab.pcl`, be sure to create the directory on every node in the cluster.

> **i** A shared NFS file system managed by RMS must have the same major device number and the same minor device number on every host that will mount that file system. This is necessary to ensure the file system is remounted transparently in the event of an application failover.

8.3.2   /etc/exports.pcl

This file contains entries for all file systems that may be made highly available for mounting on other hosts. RMS is responsible for sharing and unsharing each of these file systems according to the requirements of the running configuration.

For each file system to be managed by RMS, create a line in `/etc/exports.pcl` with the standard `exports` fields, and then insert the string #RMS# at the beginning of the line. For more information, see the `exports` manual page.

Example

```
#RMS#/usr   shasta*(rw)
```

> **i** RMS cannot export a **subdirectory** of a file system that is mounted from a remote server. It can only export the **root** of the remote file system.

8.4   Solaris file systems

◆ /etc/vfstab.pcl

Contains entries for all of the local file systems that are to be used as resources in the configuration. In other words, this file describes the file systems that should be mounted locally.

For each file system to be managed by RMS, create a line with the standard vfstab fields, and then insert the string #RMS# at the beginning of the line. RMS entries appear as comments and will be ignored by all processes other than PRIMECLUSTER components. For more information, see the vfstab manual page.

Example

    #RMS#/dev/dsk/c0t0d0s0 /dev/rdk/c0t0d0s0 /testfs1 ufs 1 yes -

◆ /etc/dfs/dfstab.pcl

Contains entries for all of the shared remote resources in the high-availability configuration. In other words, this file describes the file systems that can be mounted on a remote node.

For each file system to be managed by RMS, create a line with the standard dfstab fields, and then insert the string #RMS# at the beginning of the line. RMS entries appear as comments and will be ignored by all processes other than PRIMECLUSTER components. Therefore, to ensure that the NFS daemons start at boot time, there must be at least one non-comment, non-RMS entry in this file.

The non-RMS entry might be a dummy entry configured for a local file system and shared only to the local node. This would mean that no real sharing to a remote node is done, but it would still cause the NFS daemons to be started. For more information, see the dfstab manual page.

Example

The following contains both a non-RMS entry and an RMS entry:

share -F nfs -o ro=localhost /var/opt/example
#RMS# share -F nfs -o rw, root=
shasta1RMS:shasta1:045nfs045dia1:045msg:shasta1RMS: /sapmnt/045

8.4.1   NFS Lock Failover

The NFS Lock Failover feature is available for local file systems. If you configure NFS Lock Failover for a file system and the file system subsequently fails, then both the file system and its NFS locks will failover to the same node.

This feature requires the following site preparation steps:

● You must have a shared disk accessible to all nodes in the cluster.

● You must dedicate a directory to NFS Lock Failover on that shared disk. If you specify a directory that already exists, no other applications will be allowed to use it after it is configured.

● You must reserve one IP address for each application that uses NFS Lock Failover. You will specify this IP address when you configure the local file system for NFS Lock Failover. You will also configure this address in an Ip Address subapplication so it switches with the application that contains the file system.

> **i** Only one file system per userApplication object can be selected for NFS Lock Failover.

8.5 NFS servers

In a high availability environment such as RMS, an exported file system must be able to failover transparently when its server node is taken out of service: clients that mounted the file system before the failover should experience no access problems after the failover. NFS file systems require special preparation to achieve this result.

When a client mounts a remote NFS file system, it creates an internal **file handle** that it uses for future operations with the file system. To comply with NFS architecture, the client file handle includes the server's major and minor device numbers for the file system. This design can create access problems in the RMS environment. If the file system goes offline on the original server, and then comes back online on a second server that assigns different major and minor device numbers, the file handle will no longer be valid. This condition is called a **stale file handle**. The solution is to assign the same major and minor device numbers to the file system on every NFS server that may advertise that file system.

The above discussion refers to file systems in general, but in a high availability environment, the file system will actually be a shared disk volume that is accessible from any node that will export it. Preparing a shared disk volume with the same major and minor device number may require changes in the hardware or software configuration. If the shared disk volume is built on top of volume management software, additional steps may be necessary when the volume manager is installed.

This section provides some tips for preparing volume managers for use as NFS servers in the RMS environment.

8.6 Log files

8.6.1 /var/log/messages (Linux) or /var/adm/messages (Solaris)

By default, all RMS messages go to both the system log, `messages`, and the RMS `switchlog` file (located by default in `/var/opt/SMAWRrms/log`). If you do not want to send messages to the system log, then set `HV_SYSLOG_USE = 0` in the `hvenv.local` file. By default, `HV_SYSLOG_USE = 1`.

8.7   Other system services and databases

RMS requires the following system services or databases to be configured
according to the instructions in the *Software Release Guide PRIMECLUSTER* and
the *PRIMECLUSTER Installation Guide* for your operating system:

● PRIMECLUSTER Cluster Foundation (CF), including CIP

● `/etc/nsswitch.conf` system service lookup order database

● `echo` service—Linux only

9 Appendix—States

9.1 Basic states

Table 6 lists the states that detectors may report to the base monitor:

State	Description
Faulted	Error condition encountered. The error may have occurred in the resource, in one of its children, or during script processing.
Offline	Disabled, not ready for use. The scripts have successfully disabled the resource.
Online	Enabled, ready for use. All required children are online, and no errors were encountered while scripts were processed.
Standby	Ready to be quickly brought Online when needed.

Table 6: States reported by detectors for RMS objects

Table 7 lists additional resource states that may be displayed in the Cluster Admin GUI or by hvdisp:

State	Description
Deact	Applies to userApplication objects only. Operator intervention has deactivated the application throughout the cluster (such as for maintenance purposes).
Inconsistent	Applies to userApplication objects only. The object is Offline or Faulted, but one or more resource objects in its graph have their ClusterExclusive attribute set to 1 and are Online or Faulted.
OfflineFault	Fault that occurred in the past has not yet been cleared.
Unknown	No information is available. Reported before object initialization is completed.

Table 7: Additional states that may be displayed for RMS objects

State	Description
Wait	Temporarily in transition to a known state. An action has been initiated for the affected resource, and the system is waiting for the action to be completed before allocating one of the above states.
Warning	Some warning threshold has been exceeded. Note that this state is reported only for selected resources. –
Maintenance	Manual, temporary mode of operation in which the state of an application is decoupled from the states of its dependent resources. This allows, for example, a file system to be taken offline for backup without disturbing the state of its parent application. An application in maintenance mode is usually marked with its **intended state**, which is the state that would be attained if the application were immediately taken out of maintenance mode. The maintenance mode intended states are Maintenance-Online, Maintenance-Offline, and Maintenance-Standby.

Table 7: Additional states that may be displayed for RMS objects

The interpretation of Offline and Faulted may depend on the resource type. For instance, a mount point resource can be either Online (mounted) or Offline (not mounted); in this case, the detector would never report the Faulted state. On the other hand, a detector for a physical disk can report either Online (normal operation) or Faulted (input or output error); it would never report Offline.

9.2 State details

Besides the basic states listed above, RMS may report additional state details in the following locations:

● In the Cluster Admin GUI, the properties view of an object includes the *State Details* item at the top of the list. Unlike most other attributes, which are determined at configuration time by the Wizard Tools, this information-only field is dynamically set by RMS at runtime.

● In the output of the `hvdisp` utility, the `StateDetails` column appears at the end of each line.

In most cases, the *StateDetails* field is empty. RMS typically provides this extra information when an application is in maintenance mode, or when an object is in a transitional, inconsistent, or standby state. Table 8 lists all possible *StateDetails* values for RMS objects.

Value	Description
Failed Over	`Offline` processing successful and failover initiated
Faulted	Received `Faulted` report
Inconsistent on remote	`userApplication` is `Online` on multiple hosts, but is not `Online` on the local host
Initial Fault	`userApplication` already faulted when RMS started
Joined	`SysNode` is in `Offline` state because it has joined the cluster
Killed	`SysNode` is in `Faulted` state because of a successful kill
Not Joined	`SysNode` is in `Offline` state because it has not yet joined the cluster
Offline	Received `Offline` report
Offline Failed	`Offline` processing failed
Offline Success	`Offline` processing successful
Offline intended	Intended state is `Offline`
Online	Received `Online` report
Online	`userApplication` is `Online` on multiple hosts

Table 8: StateDetails values for RMS objects

Value	Description
Online !!	Intended state is `Online`, but some resources have conflicting states
Online intended	Intended state is `Online`
PreCheckScriptFailed	`PreCheckScript` failed
Preserved	`PreserveState` set, no `Offline` processing initiated
Shutdown	`SysNode` is in `Faulted` state because it has been shutdown
Standby	Received `Standby` report
Standby !!	Intended state is `Standby`, but some resources have conflicting states
Standby intended	Intended state is `Standby`

Table 8: StateDetails values for RMS objects

For example, if an application was online on a particular node before it was put into maintenance mode, it will generally return to the online state on the same node when it leaves maintenance mode. RMS indicates this by reporting *Online intended* in the state details field on that node. On other nodes where the application was previously offline, RMS will report *Offline intended* in the state details field.

10 Appendix—Object types

The following alphabetical list describes all object types that are supplied with RMS and configured by the Wizard Tools.

andOp

> *Required attributes:*
> HostName (for direct children of a userApplication object)

Object associated with its children by a logical *AND* operator. This object type is online if all children are online, and offline if all children are offline.

controller

> *Required attributes:*
> Resource

Object that allows a parent userApplication to control one or more child userApplication objects.

ENV

> *Required attributes:*
> (none required)

Object containing clusterwide (global) environment variables.

ENVL

> *Required attributes:*
> (none required)

Object containing node-specific (local) environment variables.

gResource

> *Required attributes:*
> rKind
> rName

Custom (generic) object. Usually represents system resources such as file systems, network interfaces, or system processes.

orOp

> *Required attributes:*
> (none required)

Object associated with its children by a logical *OR* operator. This object type is online if at least one child is online.

SysNode

Required attributes:
(none required)

Represents nodes in the cluster; at least one required. Only userApplication objects are allowed as its children.

userApplication

Required attributes:
(none required)

Represents an application to be monitored; at least one required. Must have one or more SysNode objects as its parents. For each SysNode parent, it must have one child andOp with its HostName attribute set to the name of the corresponding SysNode.

11 Appendix—Attributes

Some object types require specific attributes for RMS to monitor that object type. Some attributes can be modified through the user interface, while others are managed internally by the Wizard Tools. The following sections describe the possible values of all attributes. The default value varies depending on each resource.

11.1 Attributes available to the user

Attributes in this section can be changed using the Wizard Tools user interface.

The following default value is for Cmdline resource.Other resources have a default value individually.

AlternateIp

Possible Values: Any interconnect name

Valid for `SysNode` objects. Space-separated list that RMS uses as additional cluster interconnects if the interconnect assigned to the `SysNode` name becomes unavailable. All these interconnects must be found in the `/etc/hosts` database. By default, the configuration wizards assume the alternate interconnects to node *<nodename>* have names of the form *<nodename>*rmsAI*<nn>*, where *<nn>* is a two-digit, zero-filled number. This setting is restricted to very specific configurations and must never be used in a cluster with CF as interconnect.

AutoRecover

Possible Values: 0, 1

Valid for resource objects. If set to 1, executes the online script for an object if the object becomes faulted while in an `Online` state. If the object is able to return to the `Online` state, the fault is recovered.

This attribute must be 0 for `controller` objects: RMS handles switchover of child applications automatically.

AutoRecoverCleanup

Possible Values: 0, 1

Valid for `controller` objects. If set to 1, and `AutoRecover` is 1, then a faulted child application is requested to go `Offline` before recovering. If set to 0 and `AutoRecover` is 1, then a faulted child application recovers without going `Offline`.

AutoStartUp

Possible Values: 0, 1

Valid for `userApplication` objects. If set to 1, automatically brings the application `Online` on its highest priority `SysNode` (the first node in its `PriorityList` attribute) when RMS is started. Note that the application will not start automatically if the highest priority `SysNode` is unavailable, regardless of the state of the other nodes.

You can override the `AutoStartUp` attribute for all `userApplication` objects by setting the `HV_AUTOSTARTUP` variable. See the description of `HV_AUTOSTARTUP` in the section "Local environment variables".

AutoSwitchOver

Possible Values: Valid string containing one or more of the following: `No`, `HostFailure`, `ResourceFailure`, `ShutDown`

Valid for `userApplication` objects. Configures an application for automatic switchover if it becomes faulted. The values can be combined using the vertical bar ("|") character. The `No` value inhibits automatic switchover and cannot be combined with any other value.

For backward compatibility, the numeric values 0 and 1 are accepted: 0 is equivalent to `No`, and 1 is equivalent to `HostFailure | ResourceFailure | ShutDown`.

ClusterExclusive

Possible Values: 0, 1

Valid for resource objects. If set to 1, guarantees that the resource is online on only one node in the cluster at any time. If set to 0, allows a resource to be online on more than one node at a time. Note that "online" in this context refers to any phase of online processing. For instance, if a resource is in the `Online` state on one node while its `PreOnlineScript` is executing on another node, then both resource objects would be considered as online for the purposes of this test.

The user can modify this attribute for `cmdline` subapplications only. The configuration tools control this attribute for all other subapplications.

ControlledSwitch

Possible Values: 0, 1

Valid for controlled `userApplication` objects. If set to 0, RMS allows a manual switch request from the CLI or the GUI. If set to 1, only the parent controller can issue switch requests to this `userApplication`.

FaultScript
> *Possible Values:* Valid script (character)

> Valid for all object types. Specifies a script to be run if the associated resource enters the `Faulted` state.

Halt
> *Possible Values:* 0, 1

> Valid for `userApplication` objects. Controls local node elimination in the event of a double fault. A double fault occurs when a second fault is generated during the initial fault processing of an application.

> If `Halt` is set to 1, and another node is available to run the application, a double fault will trigger the following sequence of events:

> 1. First, RMS on the local node will exit immediately.

> 2. Next, RMS on another node will invoke the Shutdown Facility to eliminate the local node.

> 3. Finally, all applications that were online on the local node, and that have their `AutoSwitchOver` parameter set to include `HostFailure`, will be switched over to the available node.

> Note: Even if all the conditions for the `Halt` attribute are met for an application (`AutoSwitchOver` setting, additional hosts available), other applications running on the same host may block the `Halt` operation. For instance, another application may have no other available hosts, or it may not have the appropriate `AutoSwitchOver` setting. In either case, RMS will continue to run on the local node. To prevent this, allocate additional hosts for the other applications and adjust their priority lists to minimize node conflicts with the application that has its `Halt` attribute set.

I_List
> *Possible Values:* Space-separated list of `SysNode` names

> Valid for all `SysNode` objects. List of additional cluster interconnects that should be monitored by RMS. These interconnects are used only by customer applications and not by any PRIMECLUSTER products. All monitored interconnects must be found in the `/etc/hosts` database. In addition, all `SysNode` objects must have the same number of additional interconnects.

LieOffline
> *Possible Values:* 0, 1

Valid for all resource objects. If set to 1, allows the resource to remain `Online` during `Offline` processing.

MonitorOnly
> *Possible Values:* 0, 1

Valid for resource objects. If set to 1, a faulted state of the object is ignored by the parent when calculating the parent's state. A parent must have at least one child for which `MonitorOnly` is not set.

> –

OfflineScript
> *Possible Values:* Valid script (character)

Valid for all object types except `SysNode` objects. Specifies the script to be run to bring the associated resource to the `Offline` state.

OnlinePriority
> *Possible Values:* 0, 1

Valid for `userApplication` objects. Allows RMS to start the application on the node where it was last online when the entire cluster was brought down and then restarted. If set to 0 or not set (the default), the application comes online on the node with the highest priority in the attribute `PriorityList`. If set to 1, the application comes online on the node where it was last online. In case of `AutoStartUp` or a priority switch, this last-online node has the highest priority, regardless of its position in the priority list.

RMS keeps track of where the application was last online by means of timestamps. The node which has the latest timestamp for an application is the node on which the application will go online. Different cluster nodes should be in time-synchronization with each other, but this is not always the case. Since RMS does not provide a mechanism for ensuring time-synchronization between the nodes in the cluster, this responsibility is left to the system administrator. If RMS detects a severe time-discrepancy between the nodes in the cluster, an ERROR message is printed to the switchlog.

The `ntp` time service should be used to establish consistent time across the nodes in the cluster. Refer to the manual page for `ntpd` or `xntpd` for more information.

The `OnlinePriority` persistent state information will be cleared if RMS is restarted with the last online node removed from the configuration.

OnlineScript

Possible Values: Valid script (character)

Valid for all objects except `SysNode` objects. Specifies the script to bring the associated resource to the `Online` or `Standby` state.

PartialCluster

Possible Values: 0, 1

Valid for `userApplication` objects. Specifies whether an application can negotiate online requests.

If set to 0, then the application can negotiate its online request only when all nodes where it can possibly run are online.

If set to 1, then the application can negotiate its online request within the current set of online nodes, even if some other nodes (including the application's primary node) are offline or faulted.

Note that a `userApplication` that has its `PartialCluster` attribute set will not be affected by startup timeouts from remote nodes: the application can still go online on the local node. See the description of `HV_AUTOSTART_WAIT` in the section "Global environment variables".

PersistentFault

Possible Values: 0, 1

Valid for `userApplication` objects. If set to 1, the application maintains a `Faulted` state across an RMS shutdown and restart. The application returns to the `Faulted` state if it was `Faulted` before, unless the fault is explicitly cleared by either 'hvutil -c' or 'hvswitch -f', or if RMS is restarted with the `Faulted SysNode` removed from the configuration.

PostOfflineScript

Possible Values: Valid script (character)

Valid for all objects except `SysNode` objects. Specifies the script to be run after the state of the associated resource changes to `Offline`.

PostOnlineScript

Possible Values: Valid script (character)

Valid for all objects except `SysNode` objects. Specifies the script to be run after the state of the associated resource changes to `Online` or `Standby`.

PreOfflineScript

> *Possible Values:* Valid script (character)

> Valid for all objects except `SysNode` objects. Specifies the script to run before the object is taken to the `Offline` state.

PreOnlineScript

> *Possible Values:* Valid script (character)

> Valid for all objects except `SysNode` objects. Specifies the script to be run before the associated resource is taken to the `Online` or `Standby` state.

PreserveState

> *Possible Values:* 0, 1

> Valid for `userApplication` objects. If set to 1, the resources are not to be taken `Offline` after a fault. Ignored if `ResourceFailure` is set to `AutoSwitchOver`.

PriorityList

> *Possible Values:* Valid list of `SysNode` names (character)

> Valid for `userApplication` objects. Contains a list of `SysNode` objects where the application can come `Online`. The order in the list determines the next node to which the application is switched during a priority switchover, ordering a switchover after a `Fault`. The list is processed circularly.

> The user specifies this attribute indirectly when selecting the nodes for an application. RMS uses the order in which the nodes were selected and creates `PriorityList` automatically. The user can change the `PriorityList` by adding individual nodes from the list in the desired order, rather than automatically selecting the entire list.

> For applications controlled by or a `controller` object, the order of nodes in `PriorityList` is ignored. However, each child application must be able to run on the nodes specified for the parent application.

Resource

> *Possible Values:* Valid name (character)

> Valid for `controller` objects. Contains the name of the child (controlled) `userApplication`.

ScriptTimeout

> *Possible Values:* 0–*MAXINT* (in seconds) or valid string of the form "*timeout_value*[:[*offline_value*][:*online_value*]]"

> Valid for all object types. Specifies the timeout value for all scripts associated with that object in the configuration file. RMS sends a kill signal to the script if the timeout expires.

> Use the string format to specify individual timeout values of *offline_value* for OfflineScript and *online_value* for OnlineScript.

ShutdownPriority

> *Possible Values:* 0–*MAXINT*

> Valid for userApplication objects. ShutdownPriority assigns a weight factor to the application for use by the Shutdown Facility.

> When interconnect failures and the resulting concurrent node elimination requests occur, SF calculates the shutdown priority of each subcluster as the sum of the subcluster's SF node weights plus the RMS ShutdownPriority of all online application objects in the subcluster. The optimal subcluster is defined as the fully connected subcluster with the highest weight.

StandbyCapable

> *Possible Values:* 0, 1

> Valid for resource objects. If set to 1, the object performs standby processing on all nodes where the parent application is supposed to be Offline.

> The user can modify this attribute for a cmdline subapplication only. The configuration tools control this attribute for all other subapplications.

StandbyTransitions

> *Possible Values:* StartUp, SwitchRequest, ClearFaultRequest or any combination joined by vertical bars (|)

> Valid for userApplication objects. The value specifies when standby processing is initiated for the application object:

> – StartUp—at startup. This setting is ignored if the real-world application is already online, or if the application object is forced to go online because the AutoStartUp attribute is set.

> – SwitchRequest—after application switchover, if the application was online before the switchover.

  – `ClearFaultRequest`—after a faulted state is cleared with
   '`hvutil -c`'.

WarningScript

  Possible Values: Valid script (character)

  Valid for GDS resource objects. Specifies the script to be run after the
  posted state of the associated resource changes to `Warning`.

11.2 Attributes managed by configuration wizards

Attributes in this section are managed internally by the configuration wizards or
by RMS at runtime.

Affiliation

  Possible Values: Any string

  Valid for resource objects. Used for display purposes in the user
  interface—no functional meaning within RMS.

Class

  Possible Values: any string

  Valid for all objects except `SysNode`. Describes the class of the resource
  object. Used by other programs for various purposes (for example,
  SNMP agents). This value is supplied by the configuration wizards.

Comment

  Possible Values: any string

  Valid for all objects. Used for documentation in the configuration file—no
  functional meaning within RMS.

DetectorStartScript

  Possible Values: Any valid detector start script

  Valid for resource object with detector. Specify the detector start
  command directly in the *<configname>*.`us` file.

  Note that a `controller` object has no detector because RMS deter-
  mines its state internally.

HostName
> *Possible Values:* Any `SysNode` name

> Must be set only in the first-level `andOp` children of a `userApplication` object. Each of these `andOp` objects associates its parent application with the `SysNode` specified in its `HostName` attribute; the child `andOp` objects also determine the priority of the application's nodes.

LastDetectorReport
> *Possible Values:* `Online`, `Offline`, `Faulted`, `Standby`

> Valid for resource objects with detector. This attribute contains the most recent detector report for the object. The value may be displayed in the Cluster Admin GUI; the possible values depend on the type of resource the object represents.

MaxControllers
> *Possible Values:* 0–512

> Valid for `userApplication` objects. Upper limit of parent `userApplication` objects for the specified child application.

NoDisplay
> *Possible Values:* 0, 1

> Valid for all object types. If set to 1, specifies that the resource should not be displayed when `hvdisp` is active. Can be overridden with '`hvdisp` −S *<resource_name>*'.

NullDetector
> *Possible Values:* on, off

> Valid for resource objects with detector. Used to disable a detector at runtime by setting `NullDetector` to on. This attribute is for use with dynamic reconfiguration only. `NullDetector` must never be set hard-coded to on in the RMS configuration file.

OfflineDoneScript
> *Possible Values:* Valid script (character)

> Valid for `userApplication` objects. The last script run after the application has completed offline processing.

PreCheckScript
> *Possible Values:* Valid script (character)

Valid for `userApplication` objects. Specifies the script to be forked as the first action during `Online` or `Standby` processing. If the script returns with a zero exit code, processing proceeds. If the script returns with an exit code other than zero, processing is not performed and an appropriate warning is logged to the `switchlog` file.

rKind

Possible Values: 0–2047

Valid for `gResource` objects. Specifies the kind of detector for the object.

rName

Possible Values: Valid string (character)

Valid for `gResource` objects. Specifies a string to be forwarded to the generic detector.

SplitRequest

Possible Values: 0, 1

Valid for `controller` objects. If set to 1, then `PreOffline` and `PreOnline` requests will be propagated to child applications separately from the `Offline` and `Online` requests. If 0, then separate `PreOffline` or `PreOnline` requests will not be issued for the child applications.

StateDetails

Possible Values: Any string

Valid for all objects. Displays additional state details in the Cluster Admin GUI or the `hvdisp` CLI user interface. In most cases, the state details field is empty. RMS typically provides this extra information when an application is in maintenance mode, or when an object is in a transitional, inconsistent, or standby state.

12 Appendix—Environment variables

This appendix provides a complete list of the environment variables used by RMS, grouped into the following types:

● "Global environment variables"

● "Local environment variables"

● "Script execution environment variables"

The discussion in the chapter "Environment variables" describes how RMS manages environment variables.

12.1 Setting environment variables

⚠️ **Caution**

Do not explicitly set RMS environment variables in the user environment. Doing so can cause RMS to lose environment variables settings.

ℹ️ Do not change the `hvenv` configuration file. Changes to your configuration's environment variables should be confined to the *<RELIANT_PATH>*`hvenv.local` file.

The values of environment variables are specified as export directives in the `hvenv.local` file. To adjust a variable's setting, you would open `hvenv.local` with a text editor of your choice and modify (or add) the appropriate line.

A typical export directive would appear as follows:

```
export SCRIPTS_TIME_OUT=200
```

When RMS starts, it reads the values of environment variables from `hvenv` and `hvenv.local` and initializes the `ENV` and `ENVL` objects respectively. No further reference is made to these two configuration files while RMS is running. Therefore, any changes you make to `hvenv.local` will not take effect until the next time RMS starts up.

Values in the `ENVL` (local) object override values in the `ENV` (global) object. If a global variable setting appears in the `hvenv.local` file, it will override the corresponding setting in the `hvenv` file. However, if you adjust a global variable

in the `hvenv.local` file on one node, you must make the same adjustment to `hvenv.local` on every other node in the cluster. Global variable settings must agree clusterwide.

While RMS is running, you can display the environment variables with the `hvdisp` command, which does not require root privilege:

- `hvdisp ENV`

- `hvdisp ENVL`

12.2 Global environment variables

| i | Global variable settings (`ENV`) are included in the configurations checksum that is common to the cluster. The checksum is verified on each node during startup of the base monitor. RMS will fail to start if it detects a checksum difference between the values on any two nodes. |

| i | The default values of the environment variables are found in *<RELIANT_PATH>*`/bin/hvenv`. They can be redefined in the `hvenv.local` configuration file. |

The following list describes the global environment variables for RMS:

`HV_AUTOSTARTUP_IGNORE`
> *Possible values:* List of RMS cluster nodes. The list of RMS cluster nodes must be the names of the `SysNode`s as found in the RMS configuration file. The list of nodes cannot include the CF name.
> *Default:* "" (empty)

> List of cluster nodes that RMS ignores when it starts. This environment variable is not set by default. A user application will begin its automatic startup processing if the `AutoStartUp` attribute is set and when all cluster nodes defined in the user application have reported `Online`. If a cluster node appears in this list, automatic startup processing will begin even if this node has not yet reported the `Online` state.

> Use this environment variable if one or more cluster nodes need to be taken out of the cluster for an extended period and RMS will continue to use the configuration file that specifies the removed cluster nodes. In this case, specifying the unavailable cluster nodes in this environment variable ensures that all user applications are automatically brought online even if the unavailable cluster nodes do not report `Online`.

⚠ **Caution**

If the `HV_AUTOSTARTUP_IGNORE` environment variable is used, ensure that it is correctly defined on all cluster nodes and that it is always kept up-to-date. When a node is brought back into the cluster, remove it from this environment variable. If this does not occur, data loss could occur because RMS will ignore this node during the startup procedure and will not check whether the application is already running on the nodes specified in this list. It is the system administrator's responsibility to keep this list up-to-date if it is used.

`HV_AUTOSTART_WAIT`

Possible values: 0–*MAXINT*
Default: 60 (seconds)

Defines the period (in seconds) that RMS waits for cluster nodes to report `Online` when RMS is started. If this period expires and not all cluster nodes are online, a switchlog message indicates the cluster nodes that have not reported `Online` and why the user application(s) cannot be started automatically.

Note that `HV_AUTOSTART_WAIT` timeouts from remote nodes will not affect a local `userApplication` that has its `PartialCluster` attribute set: the application can still go online on the local node. See the description of the `PartialCluster` attribute in the section "Attributes available to the user".

ℹ This attribute generates a warning message only. `AutoStartUp` will proceed even if the specified period has expired.

`HV_CHECKSUM_INTERVAL`

Possible values: 0–*MAXINT*
Default: 120 (seconds)

Interval in seconds for which the RMS base monitor waits for each `Online` node to verify that its checksum is the same as the local checksum.

If checksums are confirmed within this interval, then RMS on the local node continues its operations as usual. However, if a checksum from a remote node is not confirmed, or if it is confirmed to be different, then the local monitor shuts down if it has been started less than `HV_CHECKSUM_INTERVAL` seconds before.

Also, if a checksum from a remote node is not confirmed, or if the checksum is confirmed to be different, then the local monitor considers the remote node as `Offline` if that local monitor has been started more than `HV_CHECKSUM_INTERVAL` seconds before.

HV_COM_PORT

Possible values: 0–65535
Default: 8000

The communication port used by the RMS base monitor on all nodes in the cluster.

HV_LOG_ACTION_THRESHOLD

Possible values: 0–100
Default: 98

Determines conditions under which `hvlogcontrol` cleans up RMS log files. If the percentage of used space on the file system containing *RELIANT_LOG_PATH* is greater than or equal to this threshold, all subdirectories below *RELIANT_LOG_PATH* will be removed. Furthermore, if `HV_LOG_ACTION` is set to `on` and all subdirectories have already been removed, the current log files will be removed too. See "`HV_LOG_ACTION`" for more information.

HV_LOG_WARN_THRESHOLD

Possible values: 0–100
Default: 95

Defines when `hvlogcontrol` warns the user about the volume of RMS log files. If the percentage of used space on the file system containing *RELIANT_LOG_PATH* is greater than or equal to this threshold value, `hvlogcontrol` issues a warning to the user. See also `HV_LOG_ACTION_THRESHOLD` above.

HV_LOH_INTERVAL

Possible values: 0–*MAXINT*
Default: 30

Specifies the minimum difference in seconds when comparing timestamps to determine the last online host (LOH) for `userApplication`. It is determined if the `OnlinePriority` attribute is set to 1.

If the difference between the LOH timestamp entries logged in the `userApplication` on two cluster nodes is less than the time specified by this attribute, RMS does not perform `AutoStartUp` and does not allow priority switches. Instead, it sends a message to the console and waits for operator intervention.

When adjusting this variable, the quality of the time synchronization in the cluster must be taken into account. The value must be larger than any possible random time difference between the cluster hosts.

HV_USE_ELM

Possible values: 0, 1
Default: 1

Specifies the heartbeat monitoring mode used by the RMS base monitor:

0—remote node and base monitor states are detected by periodically sending UDP heartbeat packets across the network. If no heartbeats are received from a remote node during an interval defined by `HV_CONNECT_TIMEOUT`, RMS marks the node as down and waits for a recovery period before taking further action.

1—combines the Enhanced Lock Manager (ELM) method and the UDP heartbeat method. This setting is valid only when CF is installed and configured. The ELM lock is taken and held by the local node until ELM reports a remote node down or remote base monitor down. In either of these cases, the remote node is immediately killed. Until ELM reports a change in a remote node's state, RMS also monitors the UDP heartbeat of each remote node as described above, but with a much longer recovery timeout.

Whether or not ELM is enabled, a remote node is killed if its UDP heartbeat is not received before its heartbeat recovery timeout expires. When CF is not present, ELM is disabled automatically, and the heartbeat recovery timeout defaults to 45 seconds. When CF is present, ELM is enabled by default, and the heartbeat recovery timeout defaults to 600 seconds; this avoids premature node kills when the remote node is slow to respond.

Only experts should disable ELM manually. When CF is present but ELM is disabled, the default 600 second heartbeat recovery timeout is too long for efficient detection of remote RMS or node outages. In this case, the recovery timeout on the local node must also be adjusted manually by starting RMS with the '`hvcm −h` *<timeout>* `−c` *<config_file>*' command.

Note that the recovery timeout should be set to the same value on every node in the cluster. When ELM is disabled, the recommended global value is 45 seconds.

RELIANT_LOG_LIFE

Possible values: Any number of days
Default: 7 (days)

Specifies the number of days that RMS logging information is retained. Every time RMS starts, the system creates a directory that is named on the basis of when RMS was last started, and which contains all constituent log files. All RMS log files are preserved in this manner. All sub directories under RELIANT_LOG_PATH whose update time (ctime) is older than the number of days specified in this variable are deleted by a cron job.

RELIANT_LOG_PATH

Possible values: Any valid path
Default: /var/opt/SMAWRrms/log

Specifies the directory where all RMS, PCS, and Wizard Tools log files are stored. The location to store the internal log will not be changed even if the value of this variable is changed.

RELIANT_PATH

Possible values: Any valid path
Default: /opt/SMAW/SMAWRrms

Specifies the root directory of the RMS directory hierarchy. Users do not normally need to change the default setting.

RELIANT_SHUT_MIN_WAIT

Possible values: 0–*MAXINT*
Default
- Solaris: *MAXINT* (seconds)
- Linux: 900 (seconds)

Defines the period (in seconds) until the hvshut command times out.

If the hvshut command is executed with any of the −l, −s, and −a options, RMS performs shutdown processing after it performs offline processing of the active applications.

Set the total time of the following items for RELIANT_SHUT_MIN_WAIT:

1. Maximum time required to complete offline processing of applications

2. Maximum time required to shut down RMS (30 seconds)

Use the total value of the script timeout of all resources included in an application for the value of 1.

To check the script timeout value of each resource, execute the `hvdisp` command by using the resource name as the argument and see the setting value (in seconds) of the `ScriptTimeout` attribute of the resource with which the `OfflineScript` attribute is not blank.

If there are two or more applications, use the largest value of the total values of the script timeout for each application.

> **i** By increasing the value of `RELIANT_SHUT_MIN_WAIT`, the following effects occur if delay or hang-up in offline processing of applications occurs.

– It may take longer time than the setting value of `RELIANT_SHUT_MIN_WAIT` to shut down RMS or the operating system.

– It may take longer time than the setting value of `RELIANT_SHUT_MIN_WAIT` to automatically switch applications by shutting down RMS or the operating system.

If the value of `RELIANT_SHUT_MIN_WAIT` is too large, use the processing time in the case that it is expected to take the longest time until timeout of the assumed cases where offline processing of an application times out for the value of the item 1 above.

Note that if the value of `RELIANT_SHUT_MIN_WAIT` is too small, timeout of the `hvshut` command may frequently occur before offline processing of applications completes. Tune `RELIANT_SHUT_MIN_WAIT` carefully.

⚠ **Caution**

Resources may remain active without stopping because RMS terminates abnormally when the `hvshut` command times out. Under this situation, if you start RMS on a different node and forcibly activate the application, the resources may become active on several nodes at the same time, and data may be corrupted when a shared disk is controlled by the resources. For this reason, if the `hvshut` command times out, shut down the operating system of the node where RMS has terminated abnormally, or forcibly shut down the node to ensure that resources stop. Then, start RMS and applications.

12.3   Local environment variables

Local environment variable settings can vary from node to node. The following list describes the local environment variables for RMS:

HV_AUTOSTARTUP

  Possible values: 0, 1
  Default: 1 (normal processing of AutoStartUp attribute)

  Controls the action of the AutoStartUp attribute for all userApplication objects on the local node. If set to 1 (the default value) the automatic startup of each userApplication is determined by its AutoStartUp attribute (see the section "Attributes available to the user"). If set to 0, the AutoStartUp attribute is ignored and no automatic startup occurs. HV_AUTOSTARTUP can be set in the Cluster Admin *Tools* menu or by using the hvsetenv command; in either case, the change does not take effect until the next RMS startup.

HV_CONNECT_TIMEOUT

  Possible values: 5–*MAXINT*
  Default: 30 (seconds). Users do not normally need to change the default setting.

  The maximum time (in seconds) that the heartbeat from a node is not received before the base monitor assumes the connection to that node has been lost and starts the UDP heartbeat recovery timer.

  Input values less than 5 are converted internally to 5.

HV_LOG_ACTION

  Possible values: on, off
  Default: off

  Determines whether the current log files in the *RELIANT_LOG_PATH* directory will be deleted when the percentage of used space on the file system containing *RELIANT_LOG_PATH* is greater than or equal to HV_LOG_ACTION_THRESHOLD. See "HV_LOG_ACTION_THRESHOLD" for more information.

HV_MAX_HVDISP_FILE_SIZE

  Possible values: 0–*MAXINT*
  Default: 20,000,000 (bytes)

Prevents the unlimited growth of the temporary file that RMS uses to supply `hvdisp` with configuration data and subsequent configuration and state changes. The value of this variable is the maximum size in bytes of the temporary file *<RELIANT_PATH>*/`locks/`.`rms.`*<process id of the hvdisp process>*.

HV_MAXPROC

Possible values: 0–99
Default: 30

Defines the maximum number of scripts RMS can have forked at any time. The default (30) is sufficient in most cases.

HV_MLOCKALL

Possible values: 0, 1
Default: 0

If set to 1, the base monitor process and any memory it allocates will be locked in memory. If set to 0 (the default), the base monitor may be swapped out.

HV_RCSTART

Possible values: 0, 1
Default: 1 (start RMS in the `rc` script)

Determines if RMS is started in the `rc` script. If set to 1 (the default value), RMS is started automatically at system boot time. If set to 0, RMS must be started manually. HV_RCSTART can be set in the Cluster Admin *Tools* menu or by using the `hvsetenv` command. (Prerequisite for `rc` start: `CONFIG.rms` exists and contains a valid entry.)

HV_REALTIME_PRIORITY

Possible values: 1–99
Default: 50

Defines the real time priority for the RMS base monitor and its detectors. Caution should be used when adjusting this variable. High settings can prevent other OS real-time processes from getting their processor time slice. Low settings can prevent the RMS base monitor from reacting to detector reports and from performing requests from command line utilities.

This variable is processed only on Solaris platforms. It has no effect on Linux platforms.

HV_SCRIPTS_DEBUG

Possible values: 0, 1
Default: 0

Controls debugging output from RMS scripts. If this variable is set to 1, it overrides the setting for scripts that are generated and managed by the Wizard Tools, causing them to write detailed runtime information about the commands that are executed to the RMS `switchlog` file. The type of information logged may vary according to the script. This setting applies only to those scripts provided with PRIMECLUSTER products. To disable script debug message logging, delete the `HV_SCRIPTS_DEBUG` entry or set `HV_SCRIPTS_DEBUG=0` in `hvenv.local`.

Note: when this variable appears in `hvenv.local`, RMS adds it to the script environment but otherwise makes no attempt to process it. Therefore, it is not reported in the Cluster Admin GUI or in '`hvdisp ENVL`' output.

HV_SYSLOG_USE

Possible values: 0, 1
Default: 1 (in `hvenv`)

Controls output to the system log from the RMS base monitor. RMS always records RMS ERROR, FATAL ERROR, WARNING, and NOTICE messages in the RMS `switchlog` file. By default, these messages are duplicated in the system log file `/var/adm/messages` (Solaris) or `/var/log/messages` (Linux). To disable RMS messages in the system log, set `HV_SYSLOG_USE=0` in `hvenv.local`.

Note

It is recommended for Linux to change this variable to 0. If this variable is set to 1 on Linux, under a high load on the system, the system log output will hang due to the performance limit of the system log processing by OS. In this case, RMS may not respond to any heartbeat from other node.

RELIANT_HOSTNAME

Possible values: valid name
Default: <*nodename*>RMS

The name of the local node in the RMS cluster. The default value of this variable is the node name with an RMS suffix (for example: shasta1RMS), as generated by the following command:
```
export RELIANT_HOSTNAME=`cftool -l 2>/dev/null | \
tail -1 | cut -f1 -d" "`RMS
```

If this preset value is not suitable, it must be modified accordingly on all nodes in the cluster.

The specified cluster node name must correspond to the SysNode name in the *<configname>*.us configuration file. The node name determines the IP address that RMS uses for establishing contact with this node.

RELIANT_INITSCRIPT
Possible values: any executable
Default: <RELIANT_PATH>/bin/InitScript

Specifies an initialization script to be run by RMS when the system is started. This script is run before any other processes are activated. It is a global script that is run once on every cluster node on which it is defined.

RELIANT_STARTUP_PATH
Possible values: any valid path
Default: <RELIANT_PATH>/build

Defines where RMS searches at start time for the configuration files.

SCRIPTS_TIME_OUT
Possible values: 0–*MAXINT*
Default: 300 (seconds)

Specifies the global period (in seconds) within which all RMS scripts must be terminated. If a specific script cannot be terminated within the defined period, it is assumed to have failed and RMS begins appropriate processing for a script failure.

If this value is too low, error conditions will be produced unnecessarily, and it may not be possible for the applications to go online or offline. An excessively high value is unsuitable because RMS will wait for this period to expire before assuming that the script has failed.

In case the global setting is not appropriate for all objects monitored by RMS, this global value can be overridden by an object-specific setting of the ScriptTimeout attribute.

12.4   Script execution environment variables

The variables in this section are set by the RMS base monitor when it executes an object's script. These exist only in the script's environment and only for the duration of the script execution. Since these variables are explicitly set, they have no default values.

HV_APPLICATION
>   *Possible values:* any `userApplication` name
>
>   Name of the `userApplication` object at the top of the sub-tree that contains the current object.

HV_AUTORECOVER
>   *Possible values:* 0, 1
>
>   If set to 1, the script was initiated due to an `AutoRecover` attempt.

HV_FORCED_REQUEST
>   *Possible values:* 0, 1
>
>   If set to 1, the script is currently processing a forced request.

HV_LAST_DET_REPORT
>   *Possible values:* one of `Online`, `Offline`, `Faulted`, `NoReport`
>
>   Last detector report for the current object.

HV_OFFLINE_REASON
>   *Possible values:* one of `DEACT`, `SWITCH`, `FAULT`, `STOP`, `SHUT`
>
>   Reason for a resource to become `Offline` is set:
>   `SWITCH`: switch request from `userApplication` (`hvswitch`)
>   `STOP`: stop request from `userApplication` (`hvutil -f`, `hvutil -c`)
>   `FAULT`: resource failure
>   `DEACT`: deact request from `userApplication` (`hvutil -d`)
>   `SHUT`: stop request from RMS (`hvshut`)

HV_NODENAME
>   *Possible values:* any object name
>
>   Name of current object.

HV_REQUESTING_CONTROLLER
>   *Possible values:* controller name and node name
>   If non-empty, contains the name of the controller and node that initiated the request for the current script execution.

`HV_SCALABLE_CONTROLLER`
> *Possible values:* scalable controller name
> The name of the scalable controller that controls this application.

`HV_SCALABLE_INFO`
> *Possible values:* scalable applications list
> Contains the list of all scalable applications everywhere in the cluster.

`HV_SCRIPT_TYPE`
> *Possible values:* one of `PreCheckScript, PreOnlineScript, OnlineScript, PostOnlineScript, PreOfflineScript, OfflineScript, PostOfflineScript, OfflineDoneScript, FaultScript`
>
> Script type.

`NODE_SCRIPTS_TIME_OUT`
> *Possible values:* 0–*MAXINT*
>
> Timeout value for the current object and script type.

> **i** | `HV_REQUESTING_CONTROLLER`, `HV_SCALABLE_CONTROLLER` and `HV_SCALABLE_INFO` can be used only by the Online/Offline processing of the application controlled with the controller object.

13 Appendix—RMS command line interface

The primary interface for configuring RMS is the RMS Wizard Tools, and the primary interface for administering RMS is the Cluster Admin GUI. These user interfaces call the RMS command line interface (CLI), and, under certain conditions, you may find it useful to invoke the CLI directly.

The following section lists the RMS CLI commands available to administrators. Specific procedures using some of these commands are described in the chapter "Controlling RMS operation". For a complete description of any command's usage, see its online `man` page. For a list of all PRIMECLUSTER commands related to RMS, see the chapter "Appendix—List of manual pages".

> **i** With few exceptions, RMS CLI commands require root privilege. The exceptions are noted in the following list.

> **i** RMS CLI commands accept case-insensitive application names.

13.1 Available RMS CLI commands

hvassert
Tests an RMS resource for a specified resource state. It can be used in scripts when a resource must achieve a specified state before the script can issue the next command. Does not require root privilege.

hvattr
Provides an interface for changing the `AutoSwitchOver` attribute at runtime. The change can be made from a single node in the cluster and will be applied clusterwide for one or more `userApplication` objects in the currently running configuration. The values `HostFailure`, `ResourceFailure`, `ShutDown`, or `No` may be specified. For more information, see the description of the `AutoSwitchOver` attribute in "Appendix—Attributes".

hvcm
Starts the base monitor and the detectors for all monitored resources. In most cases, it is not necessary to specify options to the `hvcm` command.

The base monitor is the decision-making module of RMS. It controls the configuration and access to all RMS resources. If a resource fails, the base monitor analyzes the failure and initiates the appropriate action according to the specifications for the resource in the configuration file.

`hvconfig`

Either displays the current RMS configuration or sends the current configuration to an output file.

The output of the `hvconfig` command is equivalent to the running RMS configuration file, but does not include any comments that are in the original file. Also, the order in which the resources are listed in the output might vary from the actual configuration file.

`hvdisp`

Displays information about the current configuration for RMS resources. Does not require root privilege.

`hvdump`

Gets debugging information about RMS on the local node.

`hvlogclean`

Either saves old log files into a subdirectory whose name is the time RMS was last started, or, if invoked with the −d option, deletes old log files. In either case, `hvlogclean` creates a clean set of log files even while RMS is running.

`hvreset`

Reinitializes the graph of an RMS user application on one or more nodes in the configuration. Running scripts will be terminated, ongoing requests and contracts will be cleaned up, and information about previous failures will be purged. If the process is successful, the entire graph will be brought back into a consistent initial state, but an inconsistent state is also a possible result. Therefore, use this command for test purposes only, and never invoke it on a production cluster.

⚠ **Caution**

This command is intended for use by experts only.

`hvsetenv`

Provides an interface for changing the following RMS environment variables on the local node:

– `HV_RCSTART` controls the automatic startup of RMS.

– `HV_AUTOSTARTUP` controls the automatic startup of all applications.

–

For more information about these environment variables, see "Appendix—Environment variables".

hvshut

Shuts down RMS on one or more nodes in the configuration. The base monitor on the local node sends a message to other online nodes indicating which node or nodes will be shut down.

hvswitch

Manually switches control of a user application resource from one system node to another in the RMS configuration. The resource being switched must be of type userApplication. The system node must be of type SysNode.

hvutil

Provides general administration interface to RMS. It performs various resource administration tasks, such as dynamically setting logging levels, sending a resource Offline, clearing faulted resources or hung cluster nodes in the Wait state, setting detector time periods, setting Maintenance Mode, and so forth.

⚠ **Caution**

Setting high logging levels in hvutil can cause disk overflow if enabled for too long. See the online man pages for more information.

14 Appendix—List of manual pages

This appendix lists the online manual pages for CCBR, CF, CIP, PAS, Resource Database, RMS, SF, Monitoring Agent, SIS, Web-Based Admin View, RMS Wizards, and miscellaneous utilities. To display a manual page, enter the following command:

$ `man` *man_page_name*

i	This is not an exhaustive list. Some PRIMECLUSTER utilities are intended only for internal use, so their manual pages are not listed here.

14.1 CCBR

System administration

`cfbackup`
> save the cluster configuration information for a PRIMECLUSTER node

`cfrestore`
> restore saved cluster configuration formation on a PRIMECLUSTER node

14.2 CF

System administration

`cfconfig`
> configure or unconfigure a node for a PRIMECLUSTER cluster

`cfset`
> apply or modify `/etc/default/cluster.config` entries into the CF module

`cfrecon`
> dynamically reconfigure the cluster interconnects used by a node

`cftool`
> print node communications status for a node or the cluster

`rcqconfig`
    configures or reports cluster quorum settings

`rcqquery`
    acquires the state of consistency (quorum) of the cluster

14.3   CIP

System administration

`cipconfig`
    start or stop CIP 2.0

`ciptool`
    retrieve CIP information about local and remote nodes in the cluster

File format

`cip.cf`

14.4   PAS

System administration

`mipcstat`
    MIPC statistics

`clmstat`
    CLM statistics

14.5   Resource Database

i	To display a Resource Database manual page, add `/etc/opt/FJSVcluster/man` to the `MANPATH` environment variable.

System administration

`clautoconfig`
    execute of the automatic resource registration

`clbackuprdb`
      save the resource database

`clexec`
      execute the remote command

`cldeldevice`
      delete resource registered by automatic resource registration

`clinitreset`
      reset the resource database

`clinitscript`
      report the connection confirmation results for shared disk units

`clrestorerdb`
      restore the resource database

`clsetacfparam`
      checks the connections of shared disk units and sets up the operation for
      automatic resource registration

`clsetparam`
      display and change the resource database operational environment

`clsetup`
      set up the resource database

`clspconfig`
      set up the operation of patrol diagnosis

`clsptl`
      execute patrol diagnosis

`clstartrsc`
      resource activation

`clstoprsc`
      resource deactivation

`clsyncfile`
      distribute a file between cluster nodes

User command

`clgettree`
      display the tree information of the resource database

14.6   RMS

System administration

`hvassert`
    assert (test for) an RMS resource state

`hvattr`
    make clusterwide attribute changes at runtime from a single node
    (installed with the Wizard Tools)

`hvcm`
    start the RMS configuration monitor

`hvconfig`
    display or save the RMS configuration file

`hvdisp`
    display RMS resource information

`hvdump`
    collect debugging information about RMS

`hvlogclean`
    clean RMS log files

`hvlogcontrol`
    control the volume of log disks

`hvreset`
    reinitialize the graph of an RMS user application (for use by experts in
    test conditions only—not for use on production clusters)

`hvsetenv`
    controls automatic startup of RMS or all user applications on the local
    host

`hvshut`
    shut down RMS

`hvswitch`
    switch control of an RMS user application resource to another node

`hvutil`
    manipulate availability of an RMS resource

File formats

`config.us`
>   RMS configuration file

`hvenv.local`
>   RMS local environment variable file

`hvgdstartup`
>   RMS generic detector startup file

14.7   SF

System administration

`rcsd`
>   shutdown daemon for the Shutdown Facility

`sdtool`
>   interface tool for the shutdown daemon

`sfcipher`
>   tool for encrypting passwords

File formats

`rcsd.cfg`
>   configuration file for the shutdown daemon

`SA_blade.cfg`
>   configuration file for FTS server blade shutdown agent

`SA_ipmi.cfg`
>   configuration file for the BMC (Board Management Controller) shutdown agent on IPMI (Intelligent Platform Management Interface) compliant platforms

`SA_pprci.cfg`
>   configuration file for RCI shutdown agent

`SA_rccu.cfg`
>   configuration file for RCCU shutdown agent

`SA_SATxscf.cfg`
>   configuration file for Solaris midrange server XSCF board shutdown agent

`SA_SATxscf2.cfg`
>   configuration file for Solaris APL server XSCF board shutdown agent

`SA_scon.cfg`
>   configuration file for SCON shutdown agent

`SA_sspint.cfg`
>   configuration file for Sun E10000 shutdown agent

`SA_sunF.cfg`
>   configuration file for sunF system controller shutdown agent

`SA_wtinps.cfg`
>   configuration file for WTI NPS shutdown agent

`SA_icmp.cfg`
>   configuration file for ICMP shutdown agent

`SA_vmSPgp.cfg`
>   configuration file for shutdown agent (for panic) of Virtual Machine
>   Service Provider (vmSP)

`SA_vmSPgr.cfg`
>   configuration file for shutdown agent (for reset) of Virtual Machine
>   Service Provider (vmSP)

`SA_vmchkhost.cfg`
>   configuration file for shutdown agent of vmchkhost (Host OS check)

| **i** | To see the system administration man pages for any of the above shutdown agents, omit the '`.cfg`' suffix. CIP configuration file format |

14.8   Monitoring Agent

System administration

`cldevparam`
>   Changes and displays the tunable operation environment for
>   asynchronous monitoring

`clmmbmonctl`
>   Starts, stops, restarts, and displays the operating system of the MMB
>   asynchronous monitoring daemon

`clmmbsetup`
>   Registers, changes, deletes, and displays MMB information

`clrcimonctl`
>   Start, stop or restart of the RCI monitoring agent daemon, and display of
>   daemon presence

`clrccumonctl`
>   Start, stop or restart of the console monitoring agent daemon, and
>   display of daemon presence

`clrccusetup`
>   Registers, changes, deletes, or displays console information

14.9   SIS

System administration

`dtcpadmin`
>   start the SIS administration utility

`dtcpd`
>   start the SIS daemon for configuring VIPs

`dtcpstat`
>   status information about SIS

14.10 Web-Based Admin View

System administration

`fjsvwvbs`
>   stop Web-Based Admin View

`fjsvwvcnf`
>   start, stop, or restart the web server for Web-Based Admin View

`wvconf`
>   configure Web-Based Admin View for a cluster

`wvCntl`
> start, stop, or get debugging information for Web-Based Admin View

`wvGetparam`
> display Web-Based Admin View's environment variable

`wvSetparam`
> set Web-Based Admin View environment variable

`wvstat`
> display the operating status of Web-Based Admin View

14.11 RMS Wizards

RMS Wizard Tools and RMS Wizard Kit
> RMS Wizards are documented as HTML pages in the `SMAWRhv-do` package on the CD-ROM. After installing this package, the documentation is available in the following directory:
>
> *<RELIANT_PATH>*`/htdocs.solaris/wizards.en` (Solaris)
>
> *<RELIANT_PATH>*`/htdocs.linux/wizards.en` (Linux)
>
> The default value of *<RELIANT_PATH>* is `/opt/SMAW/SMAWRrms/`.

System administration

`clrwzconfig`
> set up the linking function between the PRIMECLUSTER resource manager and the middleware products after the RMS configration definitions are activated

14.12 Miscellaneous utilities

`cluster_uninstall`
> uninstall PRIMECLUSTER software (man page available only for Solaris)

`sshconf`
> configure `ssh` access between two or more nodes

15   Appendix—Release information

This appendix provides information about the changed sections in this manual.

No	VL	Edition	Section	Description
1	4.3A10	June 2011	The "Global environment variables"	Changed the explanations of the following global environment variables: - HV_LOG_ACTION_THRESHOLD - HV_LOH_INTERVAL
2	4.3A10	June 2011	The "SF"	Changed file formats.
3	4.3A10	February 2012	The "Abbreviations"	Deleted Solaris 9 and added Solaris 11 for the supported OSes.
4	4.3A10	February 2012	The "Using the Wizard Tools interface (hvw)"	Deleted "3.6 Further reading."
5	4.3A10	February 2012	The "Stopping RMS"	Added an item to Caution when stopping RMS on a node selected in the list.
6	4.3A10	February 2012	The "Stopping RMS"	Added the explanation to Caution that RMS should not be stopped while RMS is running.
7	4.3A10	February 2012	The "Switching an application"	Changed the explanation of Priority switch and Caution.
8	4.3A10	February 2012	The "Entering maintenance mode"	Added application statues for which maintenance mode can be started.
9	4.3A10	February 2012	The "Exiting mainte-nance mode"	Added explanations for exiting maintenance mode.
10	4.3A10	February 2012	The "Optional fields"	Deleted the description of MAC address.

No	VL	Edition	Section	Description
11	4.3A10	February 2012	The "Global environment variables"	Changed the description of RELIANT_SHUT_MIN_WAIT.
12	4.3A10	February 2012	The "Local environment variables"	Changed the default value of HV_CONNECT_TIMEOUT.
13	4.3A10	February 2012	The "SF"	Deleted "PRIMEPOWER" from the explanation of SA_pprci.cfg.
14	4.3A10	February 2012	The "Glossary" - automatic power control - mixed model cluster - switching mode - synchronized power control	Added explanations.

Glossary

Items in this glossary that apply to specific PRIMECLUSTER components are indicated with the following notation:

● (CF)—Cluster Foundation

● (PCS)—PRIMECLUSTER Configuration Services

● (RMS)—Reliant Monitor Services

● (SIS)—Scalable Internet Services

Some of these products may not be installed on your cluster. See your PRIME-CLUSTER sales representative for more information.

AC
    See *Access Client*.

Access Client
    GFS kernel module on each node that communicates with the Meta Data Server and provides simultaneous access to a shared file system.

activating a configuration (RMS)
    Preparing an RMS configuration to be run on a cluster. This involves two major actions: first, the configuration is **generated** on the host where the configuration was created or edited; second, the configuration is **distributed** to all nodes affected by the configuration. The user can activate a configuration using PCS, the Wizard Tools, or the CLI.

    See also *generating a configuration (RMS),* distributing a configuration (RMS).

administrative LAN
    An optional private local area network (LAN) used for administrative commands to the nodes in the cluster. To provide an extra level of security, normal users do not have access to the administrative LAN. In PRIMECLUSTER configurations, the System Console and Cluster Console reside on the administrative LAN if one is present.

    See also public LAN.

API
    See Application Program Interface.

application (RMS)

In the RMS context, an application object is a special resource used to group other resources into a logical collection. Typically, it is used to represent a real-world application or application suite in a high-availability configuration.

Application Program Interface

A shared boundary between a service provider and the application that uses that service.

application template (RMS)

A predefined group of object definition value choices used by PCS, the Wizard Tools, or the PCS Wizard Kit to create object definitions for a specific type of application.

attribute (RMS)

The part of an object definition that specifies how the base monitor acts and reacts for a particular object type during normal operations.

automatic power control

This function is provided by the Enhanced Support Facility (ESF), and it automatically switches the server power on and off.

automatic switchover (RMS)

The procedure by which RMS automatically switches control of a `userApplication` over to another node after specified conditions are detected.

See also directed switchover (RMS), failover (RMS, SIS), switchover (RMS), symmetrical switchover (RMS).

availability

Availability describes the need of most enterprises to operate applications via the Internet 24 hours a day, 7 days a week. The relationship of the actual to the planned usage time determines the availability of a system.

base cluster foundation (CF)

This PRIMECLUSTER module resides on top of the basic OS and provides internal interfaces for the CF (Cluster Foundation) functions that the PRIMECLUSTER services use in the layer above.

See also Cluster Foundation (CF).

base monitor (RMS)

The RMS module that maintains the availability of resources. The base monitor is supported by daemons and detectors. Each node being monitored has its own copy of the base monitor.

Cache Fusion

The improved interprocess communication interface in Oracle 9i that allows logical disk blocks (buffers) to be cached in the local memory of each node. Thus, instead of having to flush a block to disk when an update is required, the block can be copied to another node by passing a message on the interconnect, thereby removing the physical I/O overhead.

CCBR

See Cluster Configuration Backup and Restore.

CF

See Cluster Foundation (CF).

CF node name (CF)

The CF cluster node name, which is configured when a CF cluster is created.

child (RMS)

A resource defined in the configuration file that has at least one parent. A child can have multiple parents, and can either have children itself (making it also a parent) or no children (making it a leaf object).

See also resource (RMS), object (RMS), parent (RMS).

cluster

A set of computers that work together as a single computing source. Specifically, a cluster performs a distributed form of parallel computing.

See also RMS configuration (RMS).

Cluster Admin

A Java-based, OS-independent management tool for PRIMECLUSTER products such as CF, SIS, RMS and PCS. Cluster Admin is available from the Web-Based Admin View interface.

See also Cluster Foundation (CF), Scalable Internet Services (SIS), Reliant Monitor Services (RMS), PRIMECLUSTER Configuration Services (PCS), Web-Based Admin View.

Cluster Configuration Backup and Restore
> CCBR provides a simple method to save the current PRIMECLUSTER configuration information of a cluster node. It also provides a method to restore the configuration information.

Cluster Foundation (CF)
> The set of PRIMECLUSTER modules that provides basic clustering communication services.
>
> See also base cluster foundation (CF).

cluster interconnect (CF)
> The set of private network connections used exclusively for PRIME-CLUSTER communications.

Cluster Join Services (CF)
> This PRIMECLUSTER module handles the forming of a new cluster and the addition of nodes.

Configuration Definition Language (PCS)
> The syntax for PCS configuration templates.
>
> See also PRIMECLUSTER Configuration Services (PCS).

configuration file (RMS)
> In the RMS context, the single file that defines the monitored resources and establishes the interdependencies between them. The default name of this file is `config.us.`

console
> See single console.

custom detector (RMS)
> See detector (RMS).

custom type (RMS)
> See generic type (RMS).

daemon
> A continuous process that performs a specific function repeatedly.

database node (SIS)
> Nodes that maintain the configuration, dynamic data, and statistics in a SIS configuration.
>
> See also gateway node (SIS), service node (SIS), Scalable Internet Services (SIS).

detector (RMS)

A process that monitors the state of a specific object type and reports a change in the resource state to the RMS base monitor.

DHCP

Dynamic Host Control Protocol. A standard method of delivering information to a host at boot time. This is most often used to dynamically assign the host's IP address and netmask, but many other parameters are possible, including domain names, DNS servers, and time servers.

directed switchover (RMS)

The RMS procedure by which an administrator switches control of a `userApplication` over to another node.

See also automatic switchover (RMS), failover (RMS, SIS), switchover (RMS), symmetrical switchover (RMS).

distributing a configuration (RMS)

The process of copying a configuration file and all of its associated scripts and detectors to all nodes affected by the configuration. This is normally done automatically when the configuration is **activated** using PCS, the Wizard Tools, or the CLI.

See also activating a configuration (RMS), generating a configuration (RMS).

DOWN (CF)

A node state that indicates that the node is unavailable (marked as down). A `LEFTCLUSTER` node must be marked as `DOWN` before it can rejoin a cluster.

See also UP (CF), LEFTCLUSTER (CF), node state (CF).

Enhanced Lock Manager (ELM) (CF)

A light weight, high performance, highly responsive lock manger, specifically designed for providing a high reliability heartbeat messaging mechanism for PRIMECLUSTER modules.

ENS (CF)

See Event Notification Services (CF).

environment variables

Variables or parameters that are defined globally.

error detection (RMS)

The process of detecting an error. For RMS, this includes initiating a log entry, sending a message to a log file, or making an appropriate recovery response.

Event Notification Services (CF)

This PRIMECLUSTER module provides an atomic-broadcast facility for events.

failover (RMS, SIS)

With SIS, this process switches a failed node to a backup node. With RMS, this process is known as switchover.

See also automatic switchover (RMS), directed switchover (RMS), switchover (RMS), symmetrical switchover (RMS).

gateway node (SIS)

Gateway nodes have an external network interface. All incoming packets are received by this node and forwarded to the selected service node, depending on the scheduling algorithm for the service.

See also service node (SIS), database node (SIS), Scalable Internet Services (SIS).

GDS

See Global Disk Services.

generating a configuration (RMS)

The process of creating a single configuration file that can be distributed to all nodes in the configuration and activated at a later time. This is normally done automatically when the configuration is **activated** using PCS, the RMS Wizards, or the CLI.

See also activating a configuration (RMS), distributing a configuration (RMS).

generic type (RMS)

An object type which has generic properties. A generic type is used to customize RMS for monitoring resources that cannot be assigned to one of the supplied object types.

See also object type (RMS).

GFS

See Global File Services.

Global Disk Services

This optional product provides volume management that improves the availability and manageability of information stored on the disk unit of the Storage Area Network (SAN).

Global File Services

This optional product provides direct, simultaneous accessing of the file system on the shared storage unit from two or more nodes within a cluster.

Global Link Services

This PRIMECLUSTER optional module provides network high availability solutions by multiplying a network route.

GLS

See Global Link Services.

generic type (RMS)

An object type which has generic properties. A generic type is used to customize RMS for monitoring resources that cannot be assigned to one of the supplied object types.

See also object type (RMS).

graph (RMS)
See system graph (RMS).

graphical user interface

A computer interface with windows, icons, toolbars, and pull-down menus that is designed to be simpler to use than the command-line interface.

GUI

See graphical user interface.

high availability

A system design philosophy in which redundant resources are employed to avoid single points of failure.

See also Reliant Monitor Services (RMS).

Intelligent Platform Management Interface

A firmware and hardware specification that provides common interfaces for monitoring and managing computers. IPMI operates through an onboard Baseboard Management Controller (BMC) on the target machine to provide OS-independent remote management functions, whether or not the target machine is powered on.

interconnect (CF)

See cluster interconnect (CF).

Internet Protocol address

A numeric address that can be assigned to computers or applications.

See also *IP aliasing*.

Internode Communications facility

This module is the network transport layer for all PRIMECLUSTER internode communications. It interfaces by means of OS-dependent code to the network I/O subsystem and guarantees delivery of messages queued for transmission to the destination node in the same sequential order unless the destination node fails.

IP address

See *Internet Protocol address*.

IP aliasing

This enables several IP addresses (aliases) to be allocated to one physical network interface. With IP aliasing, the user can continue communicating with the same IP address, even though the application is now running on another node.

See also *Internet Protocol address*.

IPMI

See *Intelligent Platform Management Interface*.

JOIN (CF)

See *Cluster Join Services (CF)*.

keyword

A word that has special meaning in a programming language. For example, in an RMS configuration file, the keyword object identifies the kind of definition that follows.

leaf object (RMS)

A bottom object in a system graph. In the configuration file, this object definition is at the beginning of the file. A leaf object does not have children.

LEFTCLUSTER (CF)

A node state that indicates that the node cannot communicate with other nodes in the cluster. That is, the node has left the cluster. The reason for the intermediate LEFTCLUSTER state is to avoid the network partition problem.

See also *UP (CF)*, *DOWN (CF)*, *network partition (CF)*, *node state (CF)*.

link (RMS)

Designates a child or parent relationship between specific resources.

local area network

See *public LAN*.

local node

The node from which a command or process is initiated.

See also *remote node, node*.

log file

The file that contains a record of significant system events or messages. The ASCC control and satellite daemons maintain log files on every node on which they run. The Wizard Tools, PCS, the RMS base monitor, and RMS detectors each maintain their own log files as well.

Management Information Base

A hierarchical database of information about the local network device. The database is maintained by network management software such as an SNMP agent.

See also *Simple Network Management Protocol*.

MDS

See *Meta Data Server*.

message

A set of data transmitted from one software process to another process, device, or file.

message queue

A designated memory area which acts as a holding place for messages so they can be processed in the same order they were received.

Meta Data Server

GFS daemon that centrally manages the control information, or meta-data, of a file system.

MIB

See *Management Information Base*.

MMB

Abbreviation for Management Board, which is one of the hardware units installed in PRIMEQUEST.

mixed model cluster
> A cluster system that is built from different SPARC Enterprise models. For example, one node is SPARC Enterprise M3000, and another node is SPARC Enterprise M4000.
> The models are divided into the following groups: SPARC T3-1/T3-2/T3-4, SPARC Enterprise T1000/T2000, SPARC Enterprise T5120/T5220/T5140/T5240/T5440, and SPARC Enterprise M3000/M4000/M5000/M8000/M9000.

mount point
> The point in the directory tree where a file system is attached.

multihosting
> Multiple controllers simultaneously accessing a set of disk drives.

native operating system
> The part of an operating system that is always active and translates system calls into activities.

network partition (CF)
> This condition exists when two or more nodes in a cluster cannot communicate over the interconnect; however, with applications still running, the nodes can continue to read and write to a shared device, compromising data integrity.

node
> A host that is a member of a cluster.

node state (CF)
> Every node in a cluster maintains a local state for every other node in that cluster. The node state of every node in the cluster must be either UP, DOWN, or LEFTCLUSTER.
>
> See also *UP (CF)*, *DOWN (CF)*, *LEFTCLUSTER (CF)*.

object (RMS)
> A representation of a physical or virtual resource in the RMS configuration file or in a system graph.
>
> See also *leaf object (RMS)*, *object definition (RMS)*, *object type (RMS)*.

object definition (RMS)
> An entry in the configuration file that identifies a resource to be monitored by RMS. Attributes included in the definition specify properties of the corresponding resource.
>
> See also *attribute (RMS)*, *object type (RMS)*.

object type (RMS)

A category of similar resources monitored as a group, such as disk drives. Each object type has specific properties, or attributes, which limit or define what monitoring or action can occur. When a resource is associated with a particular object type, attributes associated with that object type are applied to the resource.

See also *generic type (RMS)*.

online maintenance

The capability of adding, removing, replacing, or recovering devices without shutting or powering off the node.

operating system dependent (CF)

This module provides an interface between the native operating system and the abstract, OS-independent interface that all PRIMECLUSTER modules depend upon.

Oracle Real Application Clusters (RAC)

Oracle RAC allows access to all data in a database to users and applications in a clustered or MPP (massively parallel processing) platform. Formerly known as Oracle Parallel Server (OPS).

OSD (CF)

See *operating system dependent (CF)*.

parent (RMS)

An object in the RMS configuration file or system graph that has at least one child.

See also *child (RMS), configuration file (RMS)*, leaf object (RMS), *system graph (RMS)*.

PCS

See *PRIMECLUSTER Configuration Services (PCS)*.

PCS Wizard Kit (PCS)

RMS configuration products that have been designed for specific applications. Each component of the PCS Wizard Kit includes customized default settings, subapplications, detectors, and scripts. These application wizards also tailor the PCS interface to provide controls for the additional features.

See also *PCS*, *Reliant Monitor Services (RMS)*.

primary node (RMS)

> The default node on which a user application comes online when RMS is started. This is always the node name of the first child listed in the `userApplication` object definition.

PRIMECLUSTER Configuration Services (PCS)

> The graphical configuration interface for PRIMECLUSTER products. PCS uses standard templates written in Configuration Definition Language (CDL) to provide a user-friendly configuration environment for products such as RMS. The standard templates can be modified or replaced to provide a customized interface for specific applications or installations.

PRIMECLUSTER services (CF)

> Service modules that provide services and internal interfaces for clustered applications.

private network addresses

> Private network addresses are a reserved range of IP addresses specified by the Internet Corporation for Assigned Names and Numbers (ICANN). Modern switches and routers prevent these addresses from being routed to the Internet, allowing two or more organizations to assign the same private addresses for internal use without causing conflicts or security risks.

private resource (RMS)

> A resource accessible only by a single node and not accessible to other RMS nodes.
>
> See also *resource (RMS)*, *shared resource*.

public LAN

> The local area network (LAN) by which normal users access a machine.
>
> See also *administrative LAN*.

queue

> See *message queue*.

redundancy

> The capability of one component to assume the resource load of another physically similar component in case the original component fails or is shut down. Common examples include RAID hardware and/or RAID software to replicate data stored on secondary storage devices, multiple

network connections to provide alternate data paths, and multiple nodes that can be dynamically reprovisioned to maintain critical services in a cluster.

Reliant Monitor Services (RMS)

The package that maintains high availability of user-specified resources by providing monitoring and switchover capabilities on Linux and Solaris platforms.

remote node

A node that is accessed through a LAN or telecommunications line.

See also *local node, node*.

reporting message (RMS)

A message that a detector uses to report the state of a particular resource to the base monitor.

resource (RMS)

A hardware or software element (private or shared) that provides a function such as a mirrored disk, mirrored disk pieces, or a database server. A local resource is monitored only by the local node.

See also *private resource (RMS)*, *shared resource*.

resource definition (RMS)

See *object definition (RMS)*.

resource label (RMS)

The name of the resource as displayed in a system graph.

resource state (RMS)

Current state of a resource.

RMS

See *Reliant Monitor Services (RMS)*.

RMS commands (RMS)

Commands that enable RMS resources to be administered from the command line.

RMS configuration (RMS)

A configuration made up of two or more nodes connected to shared resources. Each node has its own copy of operating system and RMS software, as well as its own applications.

RMS Wizard Kit (RMS)

    RMS configuration products that have been designed for specific applications. Each component of the Wizard Kit includes customized default settings, subapplications, detectors, and scripts. These application wizards also tailor the RMS Wizard Tools interface to provide controls for the additional features.

    See also *RMS Wizard Tools (RMS)*, *Reliant Monitor Services (RMS)*.

RMS Wizard Tools (RMS)

    A software package composed of various configuration and administration tools used to create and manage applications in an RMS configuration.

    See also *RMS Wizard Kit (RMS)*, *Reliant Monitor Services (RMS)*.

SAN

    See *Storage Area Network*.

scalability

    The ability of a computing system to efficiently handle any dynamic change in work load. Scalability is especially important for Internet-based applications where growth caused by Internet usage presents a scalable challenge.

Scalable Internet Services (SIS)

    The package that dynamically balances network traffic loads across cluster nodes while maintaining normal client/server sessions for each connection.

SCON

    See *single console*.

script (RMS)

    A shell program executed by the base monitor in response to a state transition in a resource. The script may cause the state of a resource to change.

service node (SIS)

    Service nodes provide one or more TCP services (such as FTP, Telnet, and HTTP) and receive client requests forwarded by the gateway nodes.

    See also *database node (SIS)*, *gateway node (SIS)*, *Scalable Internet Services (SIS)*.

SF

    See *Shutdown Facility*.

shared resource

A resource, such as a disk drive, that is accessible to more than one node.

See also *private resource (RMS)*, *resource (RMS)*.

Shutdown Facility

The PRIMECLUSTER interface that manages the shutdown of cluster nodes. The SF is automatically invoked during failover operations. It also notifies other PRIMECLUSTER products of the successful completion of node shutdown so that recovery operations can begin.

Simple Network Management Protocol

A set of protocols that facilitates the exchange of information between managed network devices. The protocols are implemented by software agents residing in the devices. Each agent can read and write data in the local Management Information Base (MIB) in response to SNMP requests from other devices on the network.

See also *Management Information Base*.

single console

The workstation that acts as the single point of administration for nodes being monitored by RMS. The single console software, SCON, is run from the single console.

SIS

See *Scalable Internet Services (SIS)*.

SNMP

See *Simple Network Management Protocol*.

state

See *resource state (RMS)*.

Storage Area Network

The high-speed network that connects multiple, external storage units and storage units with multiple computers. The connections are generally fiber channels.

subapplication (RMS)

A part of the configuration template that is designed to configure one resource type for high availability. The RMS configuration may include multiple instances of each resource type. The Generic template contains subapplications for commands, application controllers, IP addresses, virtual network interfaces, local and remote file systems, volume managers, and storage managers.

switching mode

LAN duplexing mode presented by GLS.

There is a total of five switching mode types: fast switching mode, NIC switching mode, GS/SURE linkage mode, multipath mode, and multilink Ethernet mode.

switchover (RMS)

The process by which RMS switches control of a userApplication over from one monitored node to another.

See also *automatic switchover (RMS)*, *directed switchover (RMS)*, *failover (RMS, SIS)*, *symmetrical switchover (RMS)*.

symmetrical switchover (RMS)

This means that every RMS node is able to take on resources from any other RMS node.

See also *automatic switchover (RMS)*, *directed switchover (RMS)*, *failover (RMS, SIS)*, *switchover (RMS)*.

synchronized power control

When the power of one node is turned in the cluster system, this function turns on all other powered-off nodes and disk array unit that are connected to nodes through RCI cables.

system disk (GDS)

Disk on which the active operating system is installed. System disk refers to the entire disk that contains the slices that are currently operating as one of the following file systems (or the swap area):

For Solaris: /, /usr, /var, or swap area
For Linux: /, /usr, /var, /boot, /boot/efi, or swap area

system graph (RMS)

A visual representation (a map) of monitored resources used to develop or interpret the RMS configuration file.

See also *configuration file (RMS)*.

template

See *application template (RMS)*.

type

See *object type (RMS)*.

UP (CF)

A node state that indicates that the node can communicate with other nodes in the cluster.

See also *DOWN (CF)*, *LEFTCLUSTER (CF)*, *node state (CF)*.

virtual disk

A pseudo-device that allows a portion or a combination of physical disks to be treated as a single logical disk. The virtual disk driver is inserted between the highest level of the OS logical input/output (I/O) system and the physical device driver(s), allowing all logical I/O requests to be mapped to the appropriate area on the physical disk(s).

See also *subapplication (RMS)*.

Web-Based Admin View

A Java-based, OS-independent interface to PRIMECLUSTER management components.

See also *Cluster Admin*.

wizard (RMS)

An interactive software tool that creates a specific type of application using pretested object definitions.

Wizard Kit (RMS)

See *PCS Wizard Kit (PCS)*, *RMS Wizard Kit (RMS)*.

Wizard Tools (RMS)

See *RMS Wizard Tools (RMS)*.

Glossary

252   J2UZ-5295-04ENZ0(03)

Abbreviations

AC
Access Client

API
application program interface

AS
Adaptive Services

bm
base monitor

CCBR
Cluster Configuration Backup/Restore

CDL
Configuration Definition Language

CF
Cluster Foundation or Cluster Framework

CIM
Cluster Integrity Monitor

CIP
Cluster Interconnect Protocol

CLI
command line interface

CLM
Cluster Manager

CRM
Cluster Resource Management

DHCP
Dynamic Host Control Protocol

DLPI
Data Link Provider Interface

ELM
Enhanced Lock Manager

ENS
>Event Notification Services

GDS
>Global Disk Services

GFS
>Global File Services

GLS
>Global Link Services

GUI
>graphical user interface

HA
>high availability

ICF
>Internode Communication Facility

I/O
>input/output

IPMI
>Intelligent Platform Management Interface

JOIN
>cluster join services module

LAN
>local area network

MDS
>Meta Data Server

MIB
>Management Information Base

MIPC
>Mesh Interprocessor Communication

MMB
>Management Board (on PRIMEQUEST systems)

NIC
>network interface card

NSM

Node State Monitor

OSD

operating system dependent

PAS

Parallel Application Services

PCS

PRIMECLUSTER Configuration Services

RCCU

Remote Console Control Unit

RCI

Remote Cabinet Interface

RMS

Reliant Monitor Services

SA

Shutdown Agent

SAN

Storage Area Network

SCON

single console software

SD

Shutdown Daemon

SF

Shutdown Facility

SIS

Scalable Internet Services

SNMP

Simple Network Management Protocol

VIP

Virtual Interface Provider

XSCF

eXtended System Control Facility

Figures

Figures

Tables

Index