

# PRIMECLUSTER™

Concepts Guide (Solaris, Linux)

## **Comments... Suggestions... Corrections...**

The User Documentation Department would like to know your opinion of this manual. Your feedback helps us optimize our documentation to suit your individual needs.

Fax forms for sending us your comments are included in the back of the manual.

There you will also find the addresses of the relevant User Documentation Department.

## **Certified documentation according DIN EN ISO 9001:2000**

To ensure a consistently high quality standard and user-friendliness, this documentation was created to meet the regulations of a quality management system which complies with the requirements of the standard DIN EN ISO 9001:2000.

cognitas. Gesellschaft für Technik-Dokumentation mbH  
[www.cognitas.de](http://www.cognitas.de)

## **Copyright and Trademarks**

Copyright © 2002, 2003 Fujitsu Siemens Computers Inc. and Fujitsu LIMITED.

All rights reserved.

Delivery subject to availability; right of technical modifications reserved.

All hardware and software names used are trademarks of their respective manufacturers.

This manual is printed on  
paper treated with  
chlorine-free bleach.

---

# Contents

|          |                                       |           |
|----------|---------------------------------------|-----------|
| <b>1</b> | <b>Preface</b>                        | <b>1</b>  |
| 1.1      | About this manual                     | 1         |
| 1.2      | Documentation                         | 2         |
| 1.3      | Conventions                           | 3         |
| 1.3.1    | Notation                              | 3         |
| 1.3.1.1  | Prompts                               | 4         |
| 1.3.1.2  | Manual page section numbers           | 4         |
| 1.3.1.3  | The keyboard                          | 4         |
| 1.3.1.4  | Typefaces                             | 4         |
| 1.3.1.5  | Example 1                             | 4         |
| 1.3.1.6  | Example 2                             | 5         |
| 1.3.2    | Command syntax                        | 5         |
| 1.4      | Important                             | 5         |
| <b>2</b> | <b>Clustering technology overview</b> | <b>7</b>  |
| 2.1      | Introduction                          | 7         |
| 2.2      | High availability                     | 8         |
| 2.2.1    | Cluster interconnect                  | 9         |
| 2.2.2    | HA manager (RMS)                      | 9         |
| 2.2.2.1  | Protecting data integrity             | 9         |
| 2.2.2.2  | Wizards                               | 15        |
| 2.3      | Scalability                           | 16        |
| <b>3</b> | <b>PRIMECLUSTER architecture</b>      | <b>17</b> |
| 3.1      | Architectural overview                | 17        |
| 3.2      | PRIMECLUSTER key design features      | 20        |
| 3.2.1    | Modularity                            | 20        |
| 3.2.2    | Platform independence                 | 20        |
| 3.2.3    | Scalability                           | 20        |
| 3.2.4    | Availability                          | 21        |
| 3.2.5    | Guaranteed data integrity             | 21        |
| 3.3      | PRIMECLUSTER modules                  | 22        |
| 3.3.1    | CF                                    | 22        |
| 3.3.1.1  | OSD                                   | 23        |
| 3.3.1.2  | ICF                                   | 23        |
| 3.3.1.3  | JOIN                                  | 24        |
| 3.3.1.4  | ENS                                   | 24        |
| 3.3.1.5  | Cluster Admin                         | 24        |
| 3.3.1.6  | PRIMECLUSTER SF                       | 25        |
| 3.3.1.7  | SCON                                  | 29        |
| 3.3.2    | Web-Based Admin View                  | 32        |

# Contents

---

|          |  |           |
|----------|--|-----------|
| 3.3.3    | RMS . . . . .  | 32        |
| 3.3.3.1  | RMS wizards . . . . .  | 34        |
| 3.3.4    | SIS . . . . .  | 34        |
| 3.3.5    | PAS . . . . .  | 35        |
| 3.3.6    | GDS . . . . .  | 36        |
| 3.3.7    | GFS . . . . .  | 39        |
| 3.3.7.1  | GFS Local file system (Solaris only) . . . . .               | 39        |
| 3.3.7.2  | GFS shared file system . . . . .                             | 39        |
| 3.3.7.3  | Benefits . . . . .   | 42        |
| 3.3.8    | GLS . . . . .  | 42        |
| 3.3.8.1  | NIC switching mode . . . . .                                 | 43        |
| 3.3.9    | SNMP . . . . .   | 44        |
| <b>4</b> | <b>Cluster interconnect details . . . . .</b>                | <b>45</b> |
| 4.1      | Overview . . . . .   | 45        |
| 4.1.1    | A cluster interconnect is different from a network . . . . . | 45        |
| 4.1.2    | Network availability . . . . .                               | 46        |
| 4.1.3    | Interconnect protocol . . . . .                              | 47        |
| 4.2      | Cluster interconnect requirements . . . . .                  | 47        |
| 4.2.1    | Redundancy . . . . .   | 48        |
| 4.2.2    | Routes . . . . .   | 49        |
| 4.2.2.1  | Heartbeats . . . . .   | 49        |
| 4.2.3    | Properties . . . . .   | 50        |
| 4.2.3.1  | Bandwidth . . . . .  | 50        |
| 4.2.3.2  | Latency . . . . .  | 52        |
| 4.2.3.3  | Reliability . . . . .  | 53        |
| 4.2.3.4  | Device interface . . . . .                                   | 53        |
| 4.2.3.5  | Security . . . . .   | 54        |
| <b>5</b> | <b>RMS . . . . .</b>   | <b>55</b> |
| 5.1      | RMS overview . . . . .                                       | 55        |
| 5.1.1    | Redundancy . . . . .   | 56        |
| 5.1.2    | Application switchover . . . . .                             | 57        |
| 5.1.2.1  | Automatic switchover . . . . .                               | 58        |
| 5.1.2.2  | Manual switchover . . . . .                                  | 58        |
| 5.1.2.3  | IP aliasing . . . . .  | 58        |
| 5.1.2.4  | Data integrity . . . . .                                     | 58        |
| 5.2      | RMS monitoring and switchover . . . . .                      | 59        |
| 5.2.1    | Base monitor . . . . .                                       | 59        |
| 5.2.2    | Configuration file . . . . .                                 | 60        |
| 5.2.2.1  | Interdependencies . . . . .                                  | 60        |
| 5.2.2.2  | Object types . . . . .                                       | 61        |
| 5.2.2.3  | Object definitions . . . . .                                 | 61        |
| 5.2.3    | Scripts . . . . .  | 62        |

|          |   |           |
|----------|---|-----------|
| 5.2.4    | Detectors . . . . .                     | 63        |
| 5.2.5    | RMS environment variables . . . . .     | 64        |
| 5.3      | RMS administration . . . . .            | 65        |
| 5.4      | Customization options . . . . .         | 65        |
| 5.4.1    | Generic types and detectors . . . . .   | 65        |
| <b>6</b> | <b>RMS wizards . . . . .</b>            | <b>67</b> |
| 6.1      | Wizard overview . . . . .               | 67        |
| 6.2      | Wizard architecture . . . . .           | 67        |
| 6.3      | RMS Wizard Tools . . . . .              | 68        |
| 6.3.1    | Shared-storage applications . . . . .   | 69        |
| 6.4      | RMS Application Wizards . . . . .       | 69        |
| <b>7</b> | <b>SIS . . . . .</b>                    | <b>71</b> |
| 7.1      | SIS overview . . . . .                  | 71        |
| 7.1.1    | Features . . . . .                      | 72        |
| 7.1.2    | Service nodes . . . . .                 | 72        |
| 7.1.3    | Gateway nodes . . . . .                 | 73        |
| 7.1.4    | Primary database node . . . . .         | 73        |
| 7.1.5    | Backup database node . . . . .          | 73        |
| 7.2      | SIS design . . . . .                    | 73        |
| 7.3      | VIP load-balancing algorithms . . . . . | 75        |
| 7.4      | Proxy addresses . . . . .               | 75        |
| 7.5      | Private addresses . . . . .             | 75        |
| 7.6      | Failover . . . . .                      | 76        |
|          | <b>Glossary . . . . .</b>               | <b>77</b> |
|          | <b>Abbreviations . . . . .</b>          | <b>91</b> |
|          | <b>Figures . . . . .</b>                | <b>95</b> |
|          | <b>Index . . . . .</b>                  | <b>97</b> |


# Contents

---

---

# 1 Preface

This manual is a conceptual overview of the PRIMECLUSTER suite of products. This suite of products is a fourth-generation clustering solution that provides high availability and scalability, which is independent of the operating system and hardware platform. Its modular software architecture consists of a basic set of modules deployed on all computers (nodes) in a cluster, plus optional modules that support specific types of applications. The modular architecture allows flexible clustering solutions for a wide range of customers—solutions that can be adapted to virtually any current or future platform.

 This guide describes all the components of PRIMECLUSTER. All of these components might not be available for all releases. The customer release notes on the CD should be checked to verify which features are available for a specific platform.

This manual is intended for end users, system administrators, and support personnel. The purpose of this manual is to provide conceptual information only. It is not designed as a guide for administration, configuration, or installation (for more information, refer to the Section “Documentation”).

## 1.1 About this manual

This manual is organized as follows:

- The Chapter “Clustering technology overview” describes the concepts and benefits of clustering, including the main components of PRIMECLUSTER.
- The Chapter “PRIMECLUSTER architecture” explains the PRIMECLUSTER architecture and discusses the key features that PRIMECLUSTER provides.
- The Chapter “Cluster interconnect details” discusses the concepts, requirements, and design considerations of the cluster interconnect.
- The Chapter “RMS” introduces the basic concepts, components, and benefits of RMS.
- The Chapter “RMS wizards” explains the concepts of the two RMS wizard products, RMS Wizard Tools and RMS Application Wizards.
- The Chapter “SIS” introduces the basic concepts and components of the Scalable Internet Services (SIS).

## 1.2 Documentation

The documentation listed in this section contains information relevant to PRIMECLUSTER and can be ordered through your sales representative.

In addition to this manual, the following manuals are also available for PRIMECLUSTER:

- *Installation Guide (Solaris)*—Provides instructions for installing PRIMECLUSTER.
- *Installation Guide (Linux)*—Provides instructions for installing PRIMECLUSTER.
- *Cluster Foundation Configuration (CF) and Administration Guide (Solaris)*—Provides instructions for configuring and administering the PRIMECLUSTER Cluster Foundation.
- *Cluster Foundation (CF) Configuration and Administration Guide (Linux)*—Provides instructions for configuring and administering the PRIMECLUSTER Cluster Foundation.
- *Reliant Monitor Services (RMS) Configuration and Administration Guide (Solaris)*—Provides instructions for configuring and administering RMS for PRIMECLUSTER on Solaris.
- *Reliant Monitor Services (RMS) Configuration and Administration Guide (Linux)*—Provides instructions for configuring and administering the RMS for PRIMECLUSTER on Linux.
- *Scalable Internet Services (SIS) Configuration and Administration Guide (Solaris, Linux)*—Provides information on configuring and administering SIS.
- *Global Disk Services Configuration and Administration Guide (Solaris)*—Provides information on configuring and administering Global Disk Services (GDS).
- *Global Disk Services Configuration and Administration Guide (Linux)*—Provides information on configuring and administering Global Disk Services (GDS).
- *Global File Services Configuration and Administration Guide (Solaris)*—Provides information on configuring and administering Global File Services (GFS).
- *Global File Services Configuration and Administration Guide (Linux)*—Provides information on configuring and administering Global File Services (GFS).



- *Global Link Services Configuration and Administration Guide: Redundant Line Control Function (Solaris)*—Provides information on configuring and administering the redundant line control function for Global Link Services (GLS).
- *Global Link Services Configuration and Administration Guide: Redundant Line Control Function (Linux)*—Provides information on configuring and administering the redundant line control function for Global Link Services (GLS).
- *Global Link Services Configuration and Administration Guide: Multipath Function (Solaris)*—Provides information on configuring and administering the multipath function for Global Link Services (GLS).
- *Web-Based Admin View Operation Guide (Solaris)*—Provides information on using the Web-Based Admin View management GUI.
- *Web-Based Admin View Operation Guide (Linux)*—Provides information on using the Web-Based Admin View management GUI.
- *SNMP Reference Manual (Solaris)*—Provides reference information on the Simple Network Management Protocol (SNMP) product for PRIMECLUSTER on Solaris.
- Release notices for all products—These documentation files are included as html files on the PRIMECLUSTER Framework CD. Release notices provide late-breaking information about installation, configuration, and operations for PRIMECLUSTER.
- *RMS Wizards documentation package*—Available on the PRIMECLUSTER CD. These documents deal with topics like the configuration of file systems and IP addresses, or the different kinds of wizards.

## 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) privileges to execute are preceded by the system administrator prompt, the hash sign (#). In some examples, the notation `node#` indicates a root prompt on the specified node. For example, a command preceded by `fuji2#` would mean that the command was run as user `root` on the node named `fuji2`. Entries that do not require system administrator rights are preceded by a dollar sign (\$).

### 1.3.1.2 Manual page section numbers

References to the operating system commands are followed by their manual page section numbers in parentheses — for example, `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, manual page names and other literal programming elements in the main body of text. |
| <i>Italic</i>  | Variables that you must replace with an actual value.  |
| <b>Bold</b>    | 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):/:/bin/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 syntax

The command 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

Material of particular interest is preceded by the following symbols in this manual:



Contains important information about the subject at hand.



**Caution**

Indicates a situation that can cause harm to data.

---

## 2 Clustering technology overview

This chapter introduces the basic concepts and benefits of clustering, including the main components of PRIMECLUSTER.

This chapter discusses the following:

- The Section “Introduction” introduces the concepts of clustering technology.
- The Section “High availability” describes the concept of high availability (HA) and specifies how PRIMECLUSTER makes applications highly available.
- The Section “Scalability” describes the scalable benefits of PRIMECLUSTER.

### 2.1 Introduction

Clustering is still imprecisely defined in distributed computing. In general, a cluster is a combination of computers or partitions of a computer (nodes) that act cooperatively to provide one or more of the following:

- High availability (HA)—Provided by redundant components
- Scalability—Supplied by replicating application resources

This document focuses on cluster servers that provide HA and scalability as provided by the PRIMECLUSTER software suite. It does not discuss other kinds of clustering such as administrative clusters or scientific computing clusters.

Figure 1 illustrates a typical two-node cluster.

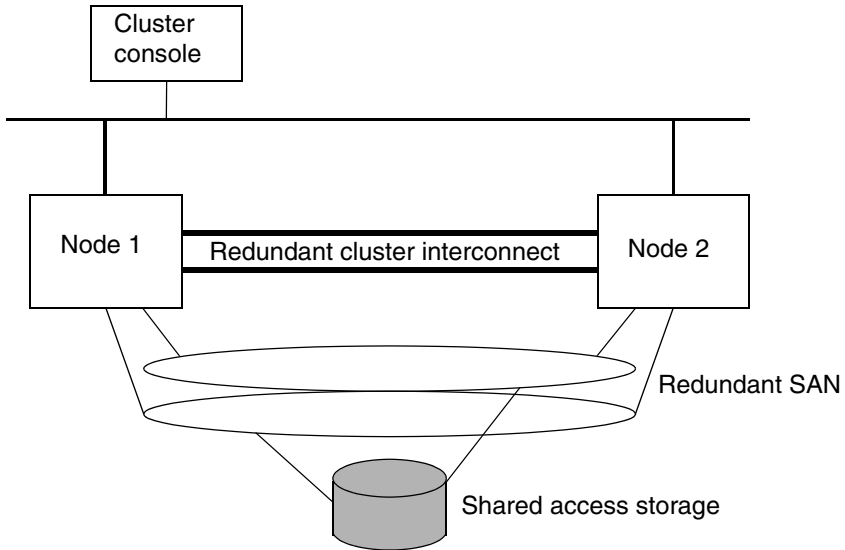


Figure 1: Typical two-node cluster

## 2.2 High availability

HA clusters use redundant components to compensate for a failure. Most HA clusters have a shared-storage environment; therefore, PRIMECLUSTER assumes, but does not require, that the cluster nodes connect to storage through a shared-access Storage Area Network (SAN). To provide redundancy, access to the SAN should include multiple host adaptors in each node. Additionally, each node in the cluster must know if the other nodes in the cluster are operational. They do this by sending heartbeats over the interconnect.

## 2.2.1 Cluster interconnect

The cluster interconnect is the basic building block of a cluster. The following rules apply to the cluster interconnect:

- A node must be connected to the cluster interconnect to participate in a cluster.
- It is strongly recommended that the cluster interconnect be redundant so that a failure in one component of the interconnect does not cause complete loss of contact within the cluster.

In addition to heartbeat requests, the cluster interconnect carries messages between nodes such as notification of events, communications between processes, and cluster file access. Additional details are discussed in the Chapter “Cluster interconnect details” later in this document.

## 2.2.2 HA manager (RMS)

Heartbeats between nodes help determine if a node is functional and if the node is able to communicate with other nodes in the cluster; however, heartbeats cannot determine if applications are running or if host adapters for the SAN are functional. PRIMECLUSTER provides Monitoring Agents (MAs), called detectors, that monitor the state of components for applications and the resources used by those applications. If a resource or application fails, the event is noted by the detector and reported to the HA manager, Reliant Monitor Services (RMS).

### 2.2.2.1 Protecting data integrity

RMS protects data integrity by performing the following tasks:

- Monitoring applications and their resources
- Ensuring that one application is only allowed to run on one node at a time (except parallel applications like Oracle RAC)
- Only automatically starting applications when all cluster nodes are in a known state

### Monitoring applications

RMS is configured with rules specific to the applications and the configuration of the cluster. When a detector reports a failure, RMS takes the appropriate actions to recover the resources that are needed to provide continued availability of the application. The recovery actions are defined for every application and resource.

RMS recovery actions are as follows:

- Local recovery—The application is not switched to another host and is brought back `Online` on the current cluster host.
- Remote recovery—The application is switched to another cluster host.

### Cluster partition

A cluster partition is the result of multiple failures in the cluster interconnects such that some or all of the nodes of the cluster continue to operate, but subsets of cluster nodes cannot communicate with each other (this is sometimes called *split-brain syndrome*). The redundant nature of the cluster interconnect prevents partitions from occurring when only a single component has failed.

Figure 2 shows an example of two breaks in the cluster interconnect in which Node 1 and Node 2 can no longer communicate with each other. However, both nodes still have access to the SAN. Therefore, if recovery actions were taken independently on each node, two instances of an application could be running unaware on the two nodes of the cluster. If these instances were to make uncoordinated updates to their data, then data corruption would occur. Clearly, this condition cannot be allowed.



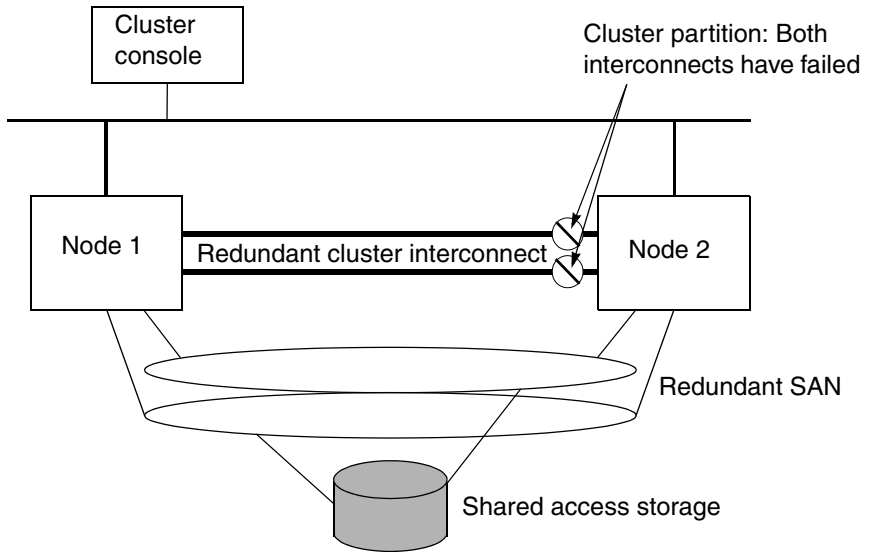


Figure 2: Cluster partition in two-node cluster

When a heartbeat failure occurs, each node sets the nodes from which they do not receive a heartbeat into a state called `LEFTCLUSTER`. The `LEFTCLUSTER` state is a state between `UP` and `DOWN` and indicates that the node is no longer part of the cluster, but its actual status is unknown. Before the nodes can take any recovery action, they must ensure that the cluster has a consistent state. The consistent state is sometimes called a quorum.

The terms consistent state and quorum are used interchangeably in `PRIMECLUSTER` documents. The cluster is in a consistent state when every node of the cluster is in a known state (`UP` or `DOWN`) and each node that is `UP` can communicate with every other node that is `UP`. The applications in the cluster should ensure that the cluster is in a consistent state before starting any operations that will alter shared data. For example, `RMS` ensures that the cluster is in a consistent state before it will start any applications in the cluster.

`PRIMECLUSTER` performs an elimination through a variety of methods, depending on the architecture of the nodes. When `PRIMECLUSTER` determines that a node is entering the `LEFTCLUSTER` state, it eliminates that node; thereby, recovering the application and guaranteeing data integrity.



The term *quorum* has been used in various ways in the literature to describe the handling of cluster partitions. Usually, this implies that when  $(n + 1)/2$  nodes can see each other, they have a quorum and the other nodes will not perform any I/O operations. The PRIMECLUSTER methods are different than the conventional meaning of quorum, so the term *cluster integrity* was adopted. There are several commands that retain the quorum heritage, so keep in mind that quorum means Cluster Integrity in PRIMECLUSTER.

### Cluster Integrity Monitor

The purpose of the Cluster Integrity Monitor (CIM) is to allow applications to determine when it is safe to perform operations on shared resources. It is safe to perform operations on shared resources when a node is a member of a cluster that is in a consistent state.

A consistent state is when all the nodes of a cluster that are members of the CIM set are in a known and safe state. The nodes that are members of the CIM set are specified in the CIM configuration. Only these nodes are considered when the CIM determines the state of the cluster.

When a node first joins or forms a cluster, the CIM indicates that the cluster is consistent only if it can determine the following:

- The status of the other nodes that make up the CIM set
- Whether the nodes of the CIM set are in a safe state

The method used to determine the state of the members of the cluster is sometimes called the CIM method. The CIM can use several different CIM methods; however, the following are available by default and are discussed here:

- NSM—The Node State Monitor (NSM) monitors the node states, and it tracks the state of the nodes that are currently, or have been, members of the cluster. This is also known as the NULL or default CIM method. NSM is an integrated part of the PRIMECLUSTER CF.
- RCI—The RCI (Remote Cabinet Interface) is a special PRIMEPOWER environmental control and state network that can both report on the state of the systems and control the systems on Solaris systems. (For more information, refer to the PRIMECLUSTER *Cluster Foundation (CF) Configuration and Administration Guide (Solaris)*.)

Multiple CIM methods can be registered and used in a certain order. The lower priority method determines the state of a node only when higher priority methods do not determine a node's state; that is, if the CIM method is unable to determine the node state. For example, the node has registered both RCI and NSM as CIM methods and RCI has higher priority. When a node that CIM checks is a PRIMEPOWER partition that the RCI CIM method can recognize, the RCI CIM method will return either `UP` or `DOWN` and this is final. However, if a node that the RCI method is checking is not connected to the RCI, then the RCI method will fail. CIM must then go to the next method; that is, the NSM CIM method to determine the state of node.

The CIM reports on whether a node state in a cluster is consistent (`true`), or a node state is not consistent (`false`) for the cluster. `True` and `false` are defined as follows:

- `TRUE`—A known state for all CIM nodes
- `FALSE`—An unknown state for any cluster CIM node

### Shutdown Facility

The CIM allows applications to determine when a cluster is in a consistent state, but it does not take action to resolve inconsistent clusters. Many different methods to ensure consistent clusters have been used in the high-availability field, but there is only one method that has proven completely effective and does not require cooperation between the nodes. PRIMECLUSTER uses this method known as the Shutdown Facility (SF) to return to a consistent cluster state when something occurs to disrupt that state. In the cluster partition example shown in Figure 2, both nodes will report the other node as having the state `LEFTCLUSTER`. The CIM will return a `FALSE` status. To get the cluster into a consistent state, SF forces one of the nodes into a safe state by either forcing a panic or shutting off the power.

The SF can be configured to eliminate nodes through a variety of methods. When the SF receives a request to eliminate a node, it tries to shut down the node by using the methods in the order that were specified. Once a method has successfully eliminated the node, the node's state is changed to `DOWN` by the SF.

The transition from `LEFTCLUSTER` to `DOWN` is the signal used by the various cluster services to start recovery actions. In the two-node cluster configuration in Figure 2, the first elimination method could be to panic the node through the cluster console. If this were to fail, the next method might be to use a method to

turn off the power to the node. Note that different systems will support different shutdown methods. For example, the cluster console is available for Solaris, but is not available for Linux.

If all of the configured SF methods fail to return a positive acknowledgement that the requested node has been eliminated, then no further action is taken. This leaves the cluster in an inconsistent state and requires operator intervention to proceed.

This fail-safe approach ensures that damage to user data could not occur by inadvertently allowing an application to run in two parts of a partitioned cluster. This also protects from the situation where a node fails to respond to heartbeats (for example, an extreme system load) and then comes back to life later. In this case, the application on the node that returns to life may continue to run even though the other node has taken action to start that application.

## Monitoring Agents

Many hardware platforms provide an environmental monitoring and control network that can report on the state of a system. For example, the PRIME-POWER hardware provides the Remote Cabinet Interface (RCI) that was discussed previously in the Shutdown Facility as a means to shut down a system. The RCI can also be used to monitor the state of the system and quickly detect when a node has panicked or been shutdown. Other hardware platforms use different interfaces, but the functionality is similar.

PRIMECLUSTER provides a mechanism where these hardware monitors can be used to quickly detect a system state change and inform the cluster membership functions. Without this monitoring capability, only the cluster heartbeat timeout will detect that a node has panicked; this will take up to 10 seconds with the default heartbeat interval. When a Monitoring Agent (MA) is used, it can detect a node panic very quickly. For example, with PRIMEPOWER hardware and the RCI, the MA takes less than 1 second to detect a system panic. MAs are implemented as plug-ins that interfaces with the Shutdown Facility.

The MA technology allows PRIMECLUSTER to recover from monitored node failures very quickly. For non-cluster aware applications the time from when a node panic occurs to the time that the application recovery begins can be as short as 2.5 seconds under optimal conditions. The time the application takes to start up and become ready to operate varies from one application to another. For cluster-aware applications, such as Oracle RAC, the time from a system panic to the time Oracle has started recovery and is processing queries on the

surviving nodes can be as short as 6.5 seconds. At this point, Oracle may still be performing some recovery actions that might impact performance, but it is able to respond to user queries.

If a node fails, PRIMECLUSTER does the following:

- Detects a node failure
- Notifies of the failure
- Confirms the node state
- Eliminates the node

The MA notifies SF of a node failure on detecting it. SF seeks a redundant confirmation regarding the node state to assess the reliability of the failure notification. This verification procedure is required to prevent the node that is normally running from being shut down.

SF confirms the node state as follows:

- Collects the node state information from all registered MAs again.
- Checks if the response to the CF heartbeat request is returned.

SF prompts the MA to eliminate the failed node when all the MAs notify SF of the node failure, and CF notifies SF of the failure in responding to the heartbeat request. When the node elimination is done, this brings the other node DOWN.

### 2.2.2.2 Wizards

To properly recover an application, RMS must know about the resources an application requires for proper operation. The configuration of the resources and the relationship between the resources can be very complex. To assist in obtaining this information for RMS, the Wizard Tools and Application Wizards simplify the configuration process. The Wizard Tools capture generic information about the cluster and common application services. The Application Wizards are expert systems for configuring specific, complex applications such as SAP R/3 or Oracle Parallel Server (OPS) when they are to be monitored by RMS.



For information on the availability of the RMS Application Wizards, contact your local customer support service.

## 2.3 Scalability

Scalability is another benefit of PRIMECLUSTER. Scalability is provided by the cluster's ability to grow in computing capacity. There are two basic types of applications relative to scalability. These types of applications can be divided as follows:

- Applications that interact closely with the cluster software and are designed for a distributed environment
- Applications that are not aware of the cluster

### **Applications that interact with cluster software**

An example of a scalable application that interacts with cluster software is OPS 9iRAC. OPS 9iRAC starts an instance of the database server on some or all the nodes of the cluster. In addition, OPS interacts with the cluster software to send messages between the instances on different nodes and must know about the state of the nodes in the cluster.

### **Applications that are not aware of the cluster**

Applications that do not have special provisions for distributing their work in the cluster can be replicated on different nodes. If the application is designed so that different users can run separate copies of the application on a single computer, then this effect can be extended by using Scalable Internet Services (SIS) (refer to the Chapter "SIS") to distribute the workload to the nodes of the cluster and Global File Services (GFS) to allow an application to run anywhere in the cluster (refer to the Section "PRIMECLUSTER modules" and the manuals for PRIMECLUSTER GFS for more details).

---

## 3 PRIMECLUSTER architecture

This chapter explains the PRIMECLUSTER architecture and discusses the key features that PRIMECLUSTER provides.

This chapter discusses the following:

- The Section “Architectural overview” introduces the basic PRIMECLUSTER components and how the components function within the clustering environment.
- The Section “PRIMECLUSTER key design features” details the key design features of PRIMECLUSTER, including operating system and platform independence, scalability, availability, and guaranteed data integrity.
- The Section “PRIMECLUSTER modules” describes the Cluster Foundation (CF), Reliant Monitor Services (RMS), Web-Based Admin View graphical user interface (GUI), and optional PRIMECLUSTER services.

### 3.1 Architectural overview

The PRIMECLUSTER design is based on a long history of clustering and high availability (HA) software and hardware experience. Figure 3 is a conceptual model of a typical cluster design that illustrates PRIMECLUSTER's position as a middleware solution. PRIMECLUSTER supplies this solution as follows:

- Easily ported to new hardware platforms, operating systems, and cluster interconnects
- Only provides services that are directly related to the management or use of the cluster
- Supplies interfaces so that other applications, such as enterprise management software, can interact with or call on services provided by PRIMECLUSTER

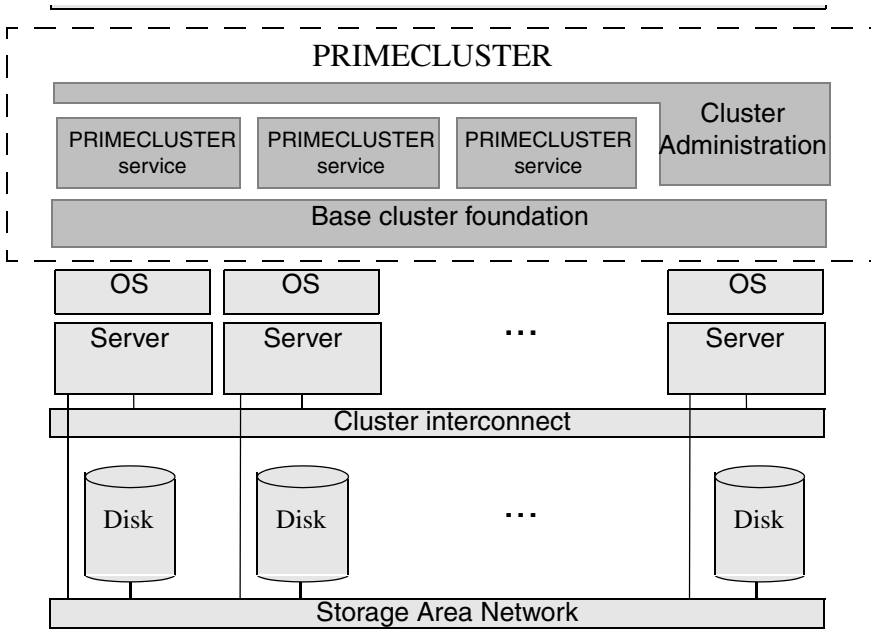


Figure 3: Diagram of a typical PRIMECLUSTER setup

Figure 4 shows a conceptual overview of the PRIMECLUSTER software architecture and how it interfaces with a server's native operating system. All of the PRIMECLUSTER software modules use operating-system-independent interfaces with an operating-system-dependant (OSD) layer to communicate between themselves and to access the base operating system services. Some examples of the operations that the OSD provides are as follows:

- Memory allocations
- Synchronizations
- Device and network access



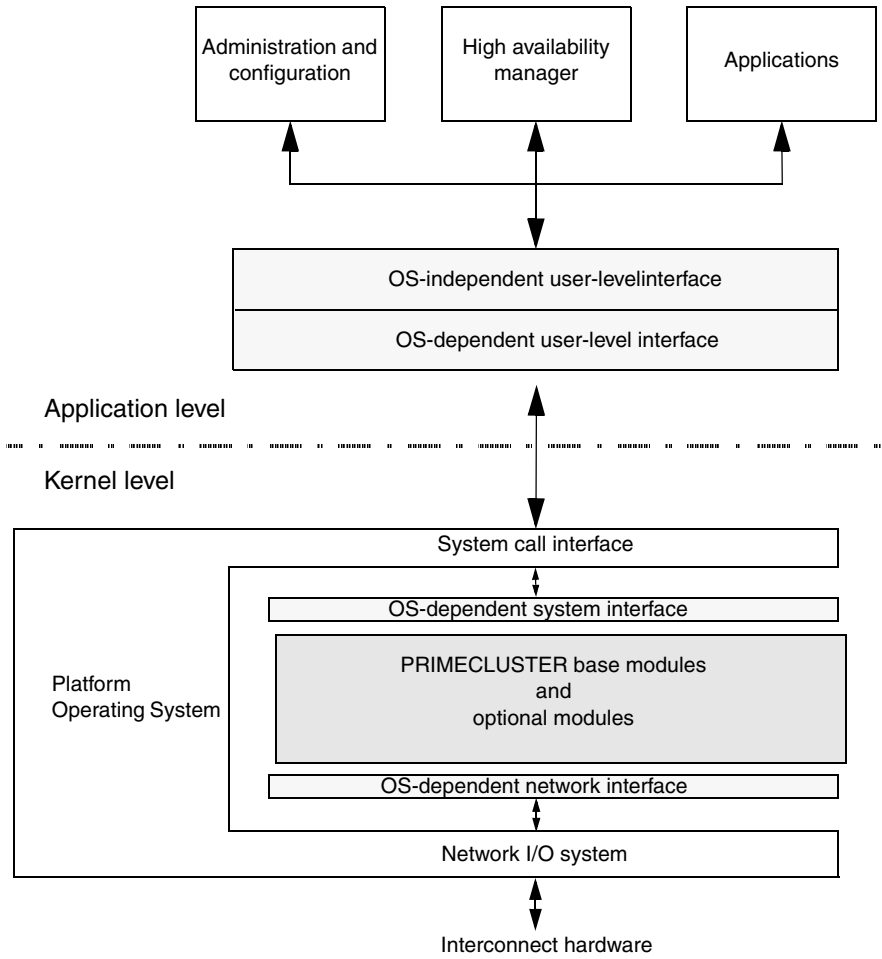


Figure 4: PRIMECLUSTER framework overview

## 3.2 PRIMECLUSTER key design features

The PRIMECLUSTER clustering software has been designed to satisfy the following goals:

- Modularity
- Platform independence
- Scalability
- Availability
- Guaranteed data integrity

### 3.2.1 Modularity

PRIMECLUSTER is composed of a core set of modules called the Cluster Foundation (CF) that provide the basic clustering functions. In addition, PRIMECLUSTER has optional modules such as the Parallel Application Services (PAS) module, the Scalable Internet Services (SIS) module, and the Reliant Monitor Services (RMS) module.

### 3.2.2 Platform independence

PRIMECLUSTER is independent of the operating system or hardware platform. Its modules are designed and coded based on an abstraction of an operating system's kernel functions. This is done with an OSD adaptation layer specific to different operating systems and network interconnects. This approach allows PRIMECLUSTER to plug seamlessly into the supported operating system without any modification to that operating system.

### 3.2.3 Scalability

PRIMECLUSTER provides not only high availability, but scalability as well. PRIMECLUSTER allows multiple nodes to act together to provide a common service. For example, with SIS, you can distribute web hits across all nodes in a cluster. With PAS, you can run a parallel database across multiple nodes. GFS provides you with a clusterwide file system so cooperating processes on many nodes can access the same data.

PRIMECLUSTER's scalability is important for customers who find that an application's demand for resources (especially CPU resources) has exceeded the capacity of a single machine. By clustering the nodes, more capacity is available for these types of applications.

### **3.2.4 Availability**

PRIMECLUSTER implements a symmetric software architecture: All cluster information is fully replicated across nodes, thus avoiding single software points of failure. PRIMECLUSTER also supports multiple redundant interconnects, avoiding single hardware points of failure. RMS, the PRIMECLUSTER high availability manager, ensures the availability of applications by restarting or recovering applications on other nodes if there is a node failure.

PRIMECLUSTER also includes an optional network load-balancing module (SIS, GLS) that can improve network availability.

### **3.2.5 Guaranteed data integrity**

PRIMECLUSTER's algorithms guarantee that network partitions (or split-brain scenarios)— even during multiple hardware interconnect failures—do not result in data inconsistency problems. The quorum algorithms ensure that only one partial cluster can operate during a network partition.

## 3.3 PRIMECLUSTER modules

The required PRIMECLUSTER modules are as follows:

- Cluster Foundation (CF)—Provides the basic cluster services on which all other modules are built. Included within this module is Cluster Admin, which supplies the interface for cluster administration, configuration, monitoring, and diagnostics services.
- Web-Based Admin View—Provides a framework under which all the PRIMECLUSTER GUIs, including Cluster Admin, run.

The optional modules are as follows:

- Reliant Monitor Services (RMS)—Manages high-availability switchover (failover) of application processes and their resources. RMS also contains the RMS wizards, which make RMS configuration easier.
- Parallel Application Services (PAS)—Provides a high-performance, low-latency communication facility that can be used by applications.
- Scalable Internet Services (SIS)—Performs network load balancing and provides a highly available interface to the cluster.
- Global Disk Services (GDS)—Provides the volume management component that improves the availability and manageability of disk-stored data.
- Global File Services (GFS)—Provides a file system that can be accessed from two or more nodes to which a shared disk unit is connected.
- Global Link Services (GLS)—Enables high reliability communications through the use of multiple network interface cards to create multiple redundant paths to a local system.
- PRIMECLUSTER Simple Network Management Protocol (SNMP)—Collects and reports information and statistics on the status and configuration of PRIMECLUSTER modules.

### 3.3.1 CF

The Cluster Foundation is the base on which all the other modules are built. It provides the fundamental services, such as the OSD, that all other PRIMECLUSTER components use as well as the base cluster services.

CF has the following features:

- Contains a loadable pseudo device driver that automatically loads when the system starts
- Supplies the CF driver that contains the CF kernel-level OSD and generic modules

Some of the functions that CF provides are detailed in the sections that follow.

### **3.3.1.1 OSD**

The CF operating-system-dependant (OSD) module provides an interface between the native OS and the abstract, OS-independent interface upon which all PRIMECLUSTER modules depend. This allows PRIMECLUSTER to use the same source files for all supported operating systems and architectures. The two main advantages of this design are as follows:

- Only need to maintain one version of the source
- Simplifies the porting of CF to a new operating system or architecture

### **3.3.1.2 ICF**

The Internode Communication Facility (ICF) module is the network transport layer for all PRIMECLUSTER inter-node communications. It provides the following functions:

- Ordered, guaranteed, node-to-node datagram communication services
- Interfaces via OS-dependent code to the Network I/O sub-system
- Guarantees to deliver messages queued for transmission to the destination node in the same sequential order, unless the destination node fails

To avoid a single point of hardware failure, ICF supports multiple interconnects. When multiple interconnects are available, ICF spreads messages across all available interconnects to improve performance. ICF automatically switches between interconnects when a failure occurs. ICF has a route recovery mechanism for transient interconnect failures such as a network switch being powered off and on again.

ICF is only usable by the CF internal components and is not available to user-level resources. To provide applications with an access to the cluster interconnect, Cluster Interconnect Protocol (CIP) is used. CIP provides a standard TCP/IP protocol suite over ICF.

### 3.3.1.3 JOIN

The cluster join services module (JOIN) dynamically joins a node to a cluster. If the cluster does not exist, then CF creates it. It is possible for several nodes to simultaneously try to form a cluster. The mastering algorithm solves this problem by using a distributed-election algorithm to determine which node should become master and form the cluster. Each node has equal rank, and each node can form the initial one-node cluster.

After the initial cluster is formed, all other nodes can join it. JOIN has built-in support for rolling upgrades by providing versioning information during the initial mastering phase. A new node joining the cluster automatically selects the protocol version in use by the current cluster.

### 3.3.1.4 ENS

The Event Notification Services (ENS) module provides an atomic-broadcast facility for events. Messages queued to ENS are guaranteed to either be delivered to all of the nodes or to none of the nodes. PRIMECLUSTER modules and application programs can both use ENS. Applications can register with ENS to receive notification of cluster events such as nodes joining or leaving the cluster. Applications can also define and broadcast application-specific events to other applications that register for it.

### 3.3.1.5 Cluster Admin

The Cluster Admin manager is an administrative interface for the following cluster features:

- Configuration
- Administration
- Operations and diagnostics services

Administration can be done from any node in the cluster, remotely from the Internet, or from both. A Java-enabled Web browser serves as the administrative interface; a conventional command-line interface is also available on a node. Diverse, clear-reporting metrics and event logs provide concise and timely information on the state of the cluster.



Not all PRIMECLUSTER products are represented in Cluster Admin.

### 3.3.1.6 PRIMECLUSTER SF

The PRIMECLUSTER Shutdown Facility (SF) provides an interface to guarantee machine shutdown during error processing within the cluster. PRIMECLUSTER SF is made up of the following major components:

- Shutdown Daemon (SD)—The SD monitors the state of cluster machines and provides an interface for gathering status and requesting manual machine shutdown.
- One or more Shutdown Agents (SA)—The SA's role is to guarantee the shutdown of a remote cluster host.

The advantages of PRIMECLUSTER Shutdown Facility are as follows:

- Ability to shut down a cluster host with or without running RMS
- Ability to shut down a cluster host from any PRIMECLUSTER service-layer product
- SCON is optional on all Solaris clusters regardless of the number of nodes

Redundant shutdown methods are available

#### Monitoring Agent

The Monitoring Agent (MA) has the capability to monitor the state of a system and promptly detect a failure such as system panic and shutdown. This function is provided by taking advantage of the hardware features that detect the state transition and inform the upper-level modules.

Without the MA, the cluster heartbeat timeout only detects a node panic during periodic intervals. The MA allows the PRIMECLUSTER system to quickly detect a node failure.

The MA provides the following functions:

- Monitors the remote node state—The MA monitors the state of the remote node that uses the hardware features. It also notifies the Shutdown Facility (SF) of a failure in the event of an unexpected system panic and shutoff. Even when a request of responding to heartbeat is temporarily disconnected between cluster nodes because of an overloaded system, the MA recognizes the correct node state.
- Eliminates the remote node—The MA provides a function to forcibly shut down the node as Shutdown Agent (SA).

The hardware products supported by the MA are as follows:

- RCI—The MA monitors the node state and detects a node failure by using the SCF/RCI mounted on a PRIMEPOWER platform. The System Control Facility (SCF), which is implemented on a hardware platform, monitors the hardware state and notifies the upper-level modules. The MA assures node elimination and prevents access to the shared disk.
- RCCU—The console monitoring agent monitors message output to the console of each node. If an error message of a node failure is output to one node, the other node detects the message and notifies SF of a node failure. Normally, the console monitoring agent creates a loop, monitoring another node, for example, A controls B, B controls C, and C controls A. If one node goes down because of a failure, another node takes over the monitoring role instead of this failed node.

The console monitoring agent also ensures node elimination by sending a break signal to the failed node.

Figure 5 shows how the monitoring feature is taken over in a cluster system with three nodes if one node goes down. The dotted line indicates that a node monitors another node.

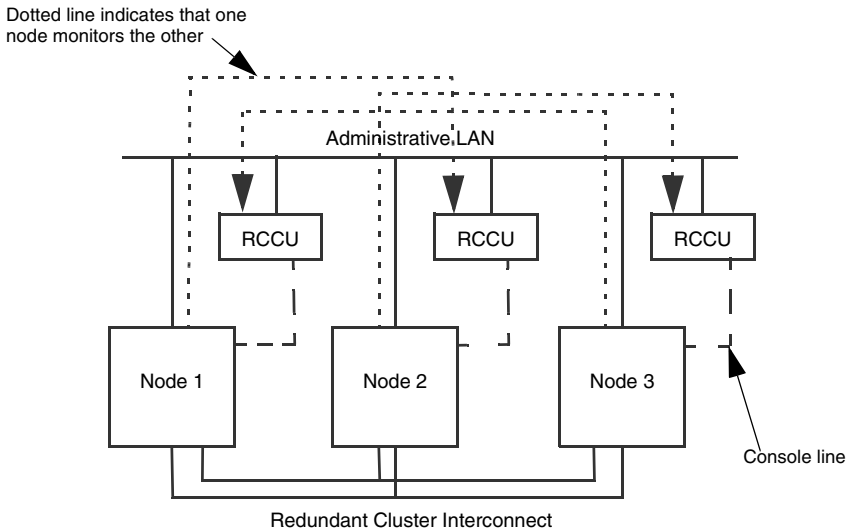


Figure 5: MA normal operation



When a failure occurs, and Node 2 is DOWN, the following actions occur:

- Node 1 begins to monitor Node 3.
- The following message is output to the `/var/adm/messages` file of Node 1:

```
FJSVcluster:Information:DEV:3044: The console monitoring agent took over monitoring (node: targetnode)
```

Figure 6 shows how Node 1 added Node 3 as the monitored node when Node 2 went down.

Node 1 added Node 3 as the monitored node.

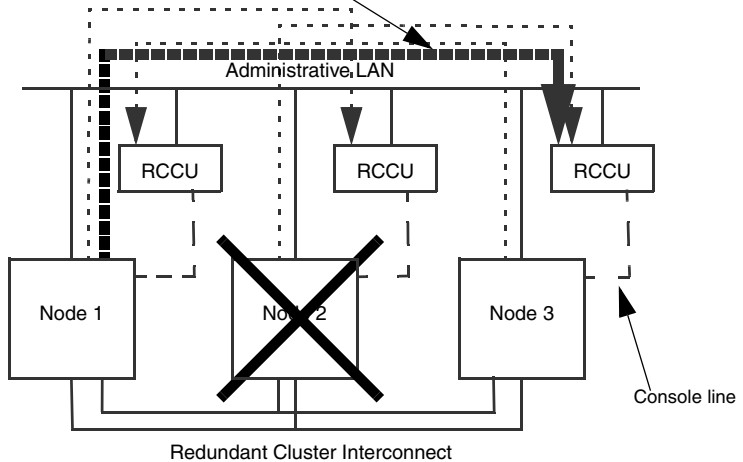


Figure 6: MA operation in the event of node failure

When Node 2 recovers from the failure and starts, the following actions occur:

- The original monitoring mode is restored.
- The following message is output to the `/var/adm/messages` file of Node 1:

```
FJSVcluster:Information:DEV:3045: The console monitoring agent cancelled to monitor (node: targetnode)
```

Figure 7 shows how Node 2 returns to monitoring Node 3 once it has been restored to the cluster.

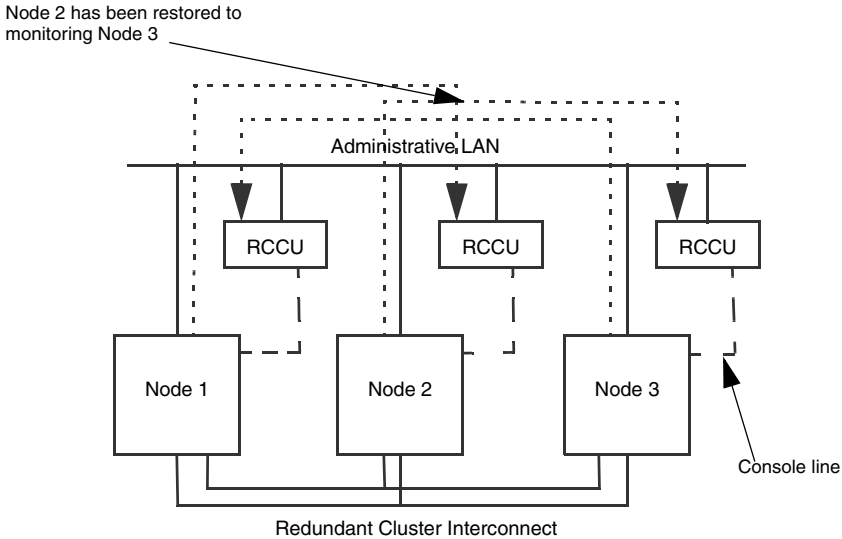


Figure 7: Node recovery

The following are possible messages that might be found in the `/var/adm/messages` file:

- `FJSVcluster:Information:DEV:3042: The RCI monitoring agent has been started`

Indicates that the RCI monitoring agent is enabled.

- `FJSVcluster:Information:DEV:3043: The RCI monitoring agent has been stopped`

Indicates that the monitoring feature is disabled.

- `FJSVcluster:Information:DEV:3040: The console monitoring agent has been started (node:monitored node name)`

Indicates that the monitoring feature of the console monitoring agent is enabled.

- `FJSVcluster:Information:DEV:3041: The console monitoring agent has been stopped (node:monitored node name)`

Indicates that the monitoring feature of the console monitoring agent is disabled. When the monitoring feature is not enabled, the other feature that forcibly brings the node `DOWN` might not work.



The console monitoring agent monitors the console message of the remote node. So it cannot recognize the node state in the event of an unexpected shutdown. In such a case, the node goes into the `LEFTCLUSTER` state, and you need to mark the remote node `DOWN`.

### 3.3.1.7 SCON



This section is for Solaris only.

In `PRIMECLUSTER`, a cluster console can be used to replace the consoles for standalone systems. This cluster console provides a single point of control for all cluster nodes. In addition to providing administrative access, a cluster console runs the `SMAWRscon` software which performs host elimination tasks when required.

In most installations of `PRIMECLUSTER`, a single cluster console can be used, but in some instances, you must configure multiple cluster consoles to provide adequate administrative access to cluster nodes.

The instances where multiple cluster consoles are needed are as follows:

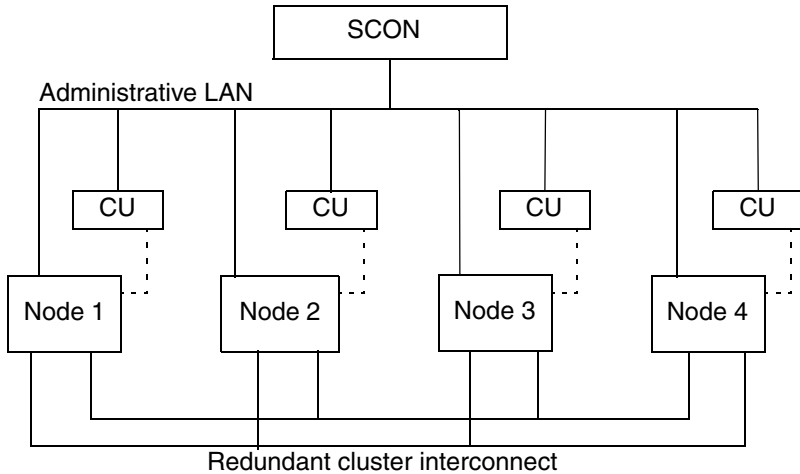
- When using two or more `PRIMEPOWER` 800, 1000, or 2000 systems that do not share a common system management console.
- Where the nodes are so far apart that it would be unreasonable for them to share a common cluster console. This may be the case when the cluster nodes are used to provide disaster recovery capabilities.

When two or more cluster consoles are used in a cluster, it is called a distributed cluster console configuration.

The cluster console attaches to the console serial ports for systems such as the `PRIMEPOWER` models 100 through 850 that do not support hardware partitioning by means of a serial-line-to-network converter unit (CU). The CU can be one of several types supported in `PRIMEPOWER` clusters such as the Remote Console Access (RCA) or Remote Console Connection Unit (RCCU). For systems that support hardware partitioning, such as the `PRIMEPOWER` models 800, 1000 – 2500, the cluster console interfaces to the System Management Console to selectively control each partition.

### Single cluster console

A single cluster console configuration is one in which the console lines for all cluster nodes are accessible from one central cluster console (see Figure 8). Note that the CU in Figure 8 represents a generic conversion unit which is responsible for converting serial-line to network access and represents either the RCA or RCCU units.



..... Console lines

Figure 8: Single cluster console configuration

The single cluster console runs the SMAWRscon software and is responsible for node elimination for all nodes in the cluster. In this configuration, all cluster nodes are known to the single cluster console, and at runtime, all cluster nodes forward shutdown requests to this console.

### Distributed cluster console

A distributed cluster console configuration is a cluster with more than one cluster console. Each of these cluster consoles has access to a selected subset of the console lines for the cluster nodes (see Figure 9). The console line for each cluster node can only be accessed by one cluster console. Note that the

CU in Figure 9 represents a generic conversion unit, which is responsible for converting serial-line to network access and represents either the RCA or RCCU units.

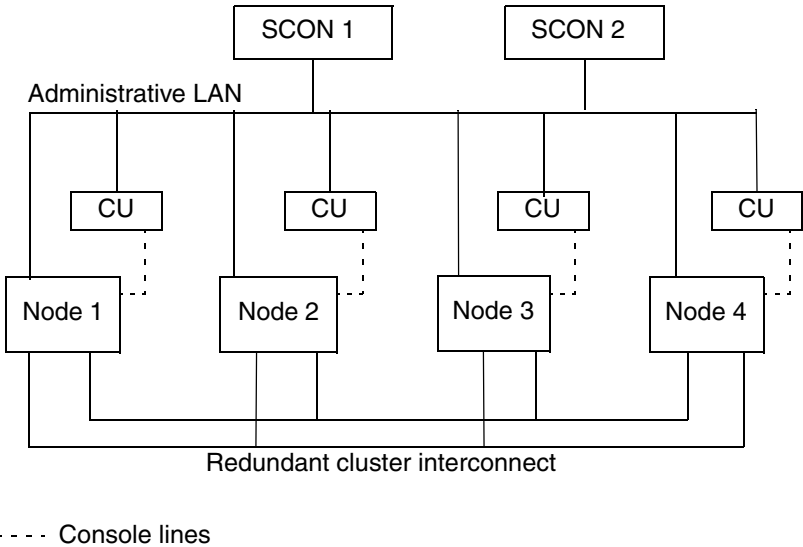


Figure 9: Distributed cluster console configuration

In Figure 9, SCON 1 controls access to Node 1 and Node 2, and SCON 2 controls access to Node 3 and Node 4. When configuring the SMAWRscon product on SCON 1, only Node 1 and Node 2 will be known by it. Similarly on SCON 2, the SMAWRscon product will only know of Node 3 and Node 4.

At runtime, all shutdown requests are sent to all cluster consoles and the cluster console responsible for the node being shutdown responds to the request.

### Network considerations

You must consider the following while configuring the network for both single cluster console and distributed cluster console configurations:

- Do not connect cluster consoles on the cluster interconnect
- Connect all CUs, cluster consoles, and cluster nodes to the administrative LAN

- Keep the administrative LAN physically separate from all public networks

### 3.3.2 Web-Based Admin View

Web-Based Admin View is a GUI framework used by the PRIMECLUSTER products. The features of Web-Based Admin View are as follows:

- A common framework for multiple GUIs. In addition to the Cluster Admin GUI, which controls CF, SIS, and RMS, PRIMECLUSTER contains GUIs for other services such as GDS and GFS. Web-Based Admin View accommodates all of these GUIs.
- A single login for multiple GUIs.
- Password encryption. Passwords sent from the client browser to the management server are encrypted.
- Logging of all GUI commands dealing with configuration or administration.
- The ability to off load the management overhead onto the management servers outside the cluster.

### 3.3.3 RMS

RMS is an application availability manager that ensures the availability of both hardware and software resources in a cluster. This is accomplished through redundancy and through the ability to fail over monitored resources to surviving nodes.

Monitored resources can be almost any system component, such as the following:

- File system
- Volume (disk)
- Application
- Network interface
- Entire node

For redundancy, RMS uses multiple nodes in the cluster. Each node is configured to assume the resource load from any other node. In addition, RAID hardware and/or RAID software replicate data stored on secondary storage devices.

For application availability, RMS monitors resources with detector programs. When a resource fails, RMS triggers a user-defined response. Normally, the response is to make the resource available on other nodes.

Resources that are mutually dependent can be combined into logical groups such that the failure of any single resource in the group triggers a response for the entire group. During switchover, RMS ensures that all of a group's resources on the original node (before the failure) are brought offline prior to any resources being brought online on the new node. This prevents any possibility of data corruption by two or more nodes attempting to access a resource simultaneously.

Figure 10 shows how RMS uses detectors to monitor resources. A detector reports any changes in the state of a resource to the RMS base monitor, which then determines if any action is required.

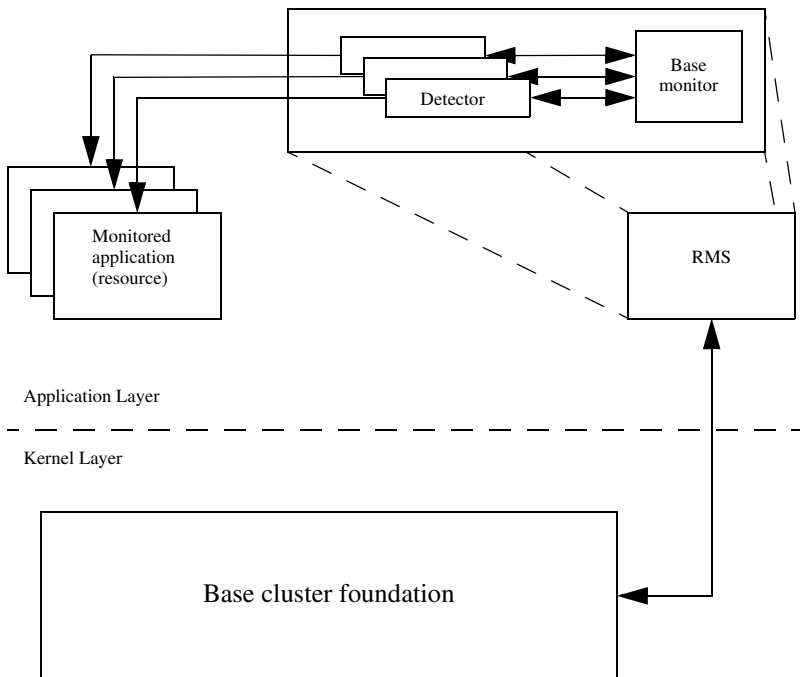


Figure 10: RMS resource monitoring

### 3.3.3.1 RMS wizards

The RMS wizards can be used to create the RMS configuration and are divided into the following two main product areas:

- **RMS Wizard Tools**—Contains the wizard framework and interfaces with the RMS base. This product simplifies the configuration setup and works with RMS and RMS Application Wizards as part of the HA (high availability) cluster environment.
- **RMS Application Wizards**—Enables an RMS HA configuration to be built for specific, customer-oriented applications. RMS Application Wizards are separately distributed software options for creating optimized HA environments for individual enterprise user applications. (Using RMS Application Wizards requires that RMS Wizard Tools be purchased and installed.)



RMS, RMS Wizard Tools, and RMS Application Wizards are separate products.

### 3.3.4 SIS

SIS enables PRIMECLUSTER to act as a scalable, reliable, and easily managed network system by providing the following:

- Access through one or more Virtual Interface Provider (VIP) addresses, which appear to clients as a single network server
- Makes the cluster appear to clients as a standard network server
- Provides transparent communication with the cluster by disguising the fact that the network request is serviced by one of the nodes in the cluster

*Example:*

Figure 11 is an example of a four-node SIS cluster. In this case, the SIS cluster is serving Web pages. The SIS cluster has one virtual address, `www.mycluster.com`, and when a client connects to this address, the Web pages are served by any of the available nodes.



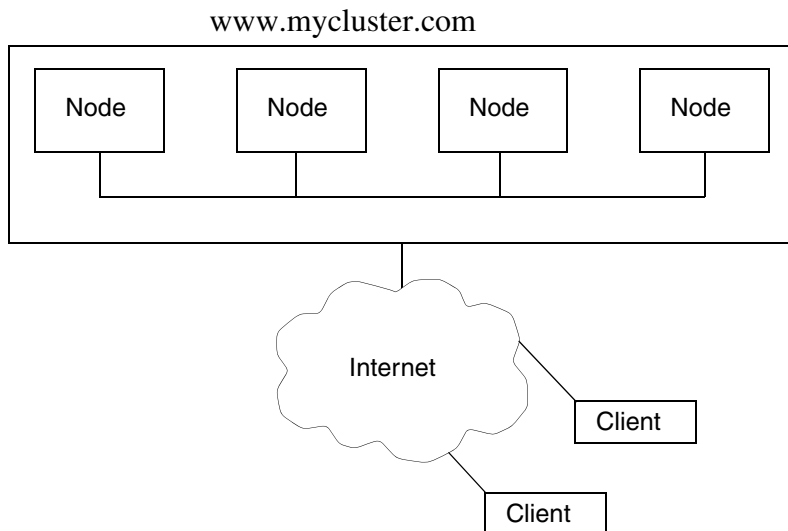


Figure 11: Example of a SIS cluster

### 3.3.5 PAS

Parallel database applications were one of the first commercial applications to use cluster technology. By partitioning the workload and data across multiple nodes in a cluster, parallel database applications can achieve performance beyond the limits of a single, large multiprocessor server.

The OPS (Oracle Parallel Server) has been the leader in parallel database applications since it was first deployed on commercial UNIX platforms in the early 90s. The Oracle 9iRAC *Cache Fusion* technology, which takes advantage of the high-speed, interprocess communication API that PRIMECLUSTER provides, now overcomes the performance bottleneck of previous OPS implementations (that is, where database instances communicated through a secondary storage system).

### 3.3.6 GDS

Global Disk Services (GDS) is a volume management software that improves the availability and manageability of disk-stored data. GDS protects data from hardware failures and operational mistakes, and supports the management of disk units.

GDS has following functions, which are closely related:

- To improve availability of disk data
- To improve manageability of disk data

GDS's mirroring function protects data from hardware failures by maintaining replicas of disk data on multiple disks. This allows users to continue to access the disk data without stopping the application in the event of an unexpected trouble.

The GDS management functions reduce the system administrator's workloads of disk management. The user-friendly functions simplify management, and at the same time, prevent data corruption by operational mistakes.

In a SAN (Storage Area Network) environment, multiple Solaris servers can be connected to multiple disk units. Disk-stored data can be accessed from those servers. This allows simultaneous access to a file system or database and improves the efficiency of data duplication between the servers and backup procedures. On the other hand, it also carries the risk of data damage, as multiple servers will compete to access the shared disk. Therefore, volume management functions suitable for the SAN environment are essential.

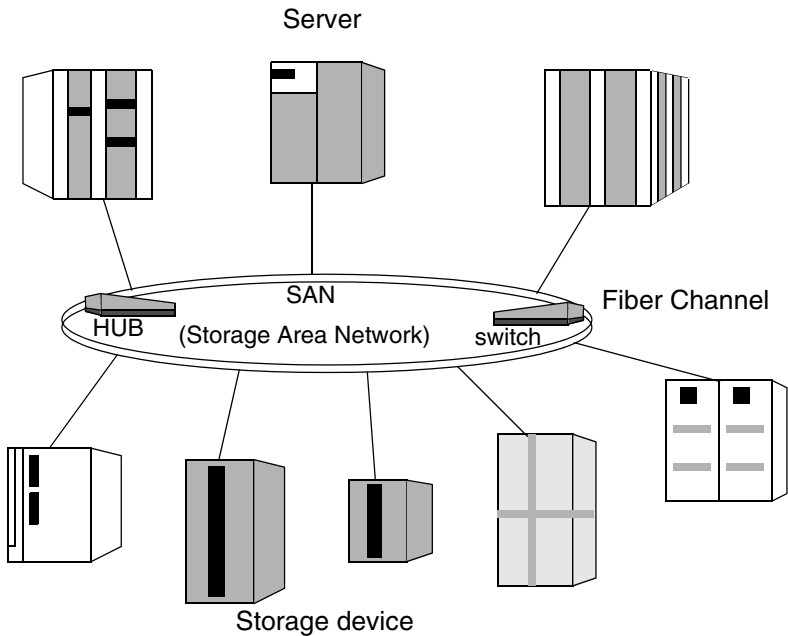


Figure 12: SAN (Storage Area Network)

GDS provides volume management functions suitable for SAN environments. GDS allows users to integrate management of all disk units connected to all servers, including local disk units that are connected to specific servers as well as disk units that are shared by multiple servers in a SAN environment.



It is only the Solaris version that manages local disks.

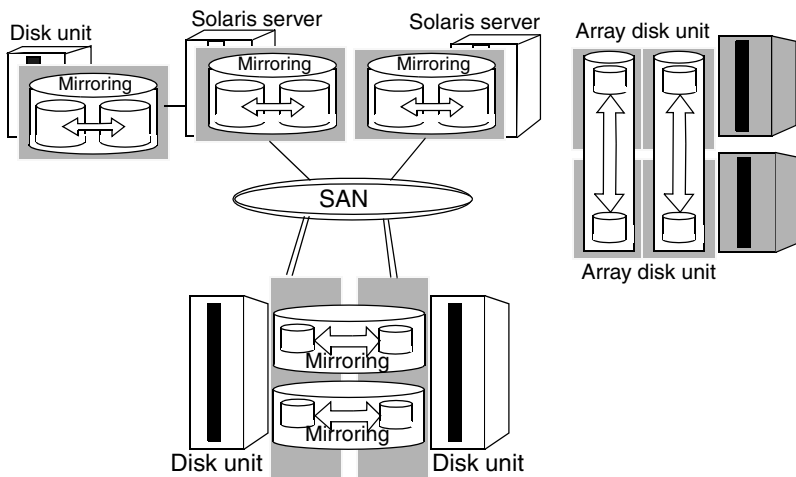


Figure 13: Mirroring between disk units (Solaris only)

GDS's main functions are as follows:

- Mirroring of system disks (Solaris only)
- Mirroring of shared disks
- Mirroring of disk array units
- Hot spare that automatically recovers mirroring in the event of a disk failure
- Hot swap that replaces a failed disk without stopping applications
- JRM (Just Resynchronization Mechanism) that pursues high-speed recovery of mirroring in the event of an unexpected system down or cluster failover
- Integrated management and access control of disks in a SAN environment
- Automatic configuration that recognizes physical connections to servers and disk units, registers the configuration information, and checks the connections (Solaris only)
- Concatenation that enables creation of large volumes of data
- Striping that distributes load of access to disks
- Logical partitioning that enables flexible use of disks
- Snap-shot that supports backup minimizing the effect on core services

See PRIMECLUSTER *Global Disk Services Configuration and Administration Guide* for further details.

### 3.3.7 GFS

The two types of GFS file system are local and shared. The GFS local file system (Solaris only) is used within one node. The GFS shared file system allows users to select optimal file system in accordance with the intended use.

#### 3.3.7.1 GFS Local file system (Solaris only)

The GFS local file system expands UFS, the representative file system of UNIX, and improves availability and performance. The GFS local file system provides the following features:

- Logical partitioning allows a file system to be partitioned into multiple logical slices.
- High-speed recovery of a file system (equivalent to UFS logging)
- Serial block allocation on an extent basis.
- Easy expansion of file system area
- Area reallocation in the event of a disk block failure
- Setup of extent attributes to a file
- Defragment

The GFS local file system provides high-speed failover of a file system in the event of a node down in standby operation, using the recovery functions of data integrity. In addition, data area and defragment functions are enabled without stopping operation services. The GFS local file system is independently used, or it is used as a failover file system by RMS.

#### 3.3.7.2 GFS shared file system

Global File Services (GFS) is a highly reliable file system, assuring simultaneous access from multiple nodes to which a shared disk unit is connected (up to 2 nodes for Linux).

GFS provides the following features:

- Simultaneous shared access to a file or file system from two or more nodes

- Data consistency and integrity when file data is referenced or updated by two or more nodes.
- When a node is down, file operation can be continued on the other node while maintaining the consistency of the file system
- File system high-speed recovery function
- High-speed input/output (I/O) processing with sequential block assignment of file area
- File access using file cache of each node
- Multi-volume supports the distribution of input/output processing load and the use of large-scale file systems (Solaris only)
- Addition of online volume (area extension) is possible without disrupting or reconfiguring the file system (Solaris only)
- Creation, deletion, expansion, and operation of the file system can be performed using a GUI based Web browser interface

### **Simultaneous shared access while maintaining consistency**

The GFS shared file system assures integrity of data when data is updated from multiple nodes. In the Solaris version, the file-lock function is enabled on multiple nodes using a conventional UFS API. When a distributed application is executed on multiple nodes, the conventional APIs are used ensuring application data transfer.

### **High availability**

When the GFS file system is used from multiple nodes, file access can be continued from the other node even if a node is down. The file system information retained by the downed node automatically restores the consistency within the GFS on the other nodes. That is, the application program running on the other nodes can continue processing without causing an error in the file system operation.

The operations required to change file system structure (such as file creation or deletion) are recorded in the area called the update log with the GFS file system. By using the information stored in this area, a system failure can be recovered in seconds without having to make a full check of the file system structure.

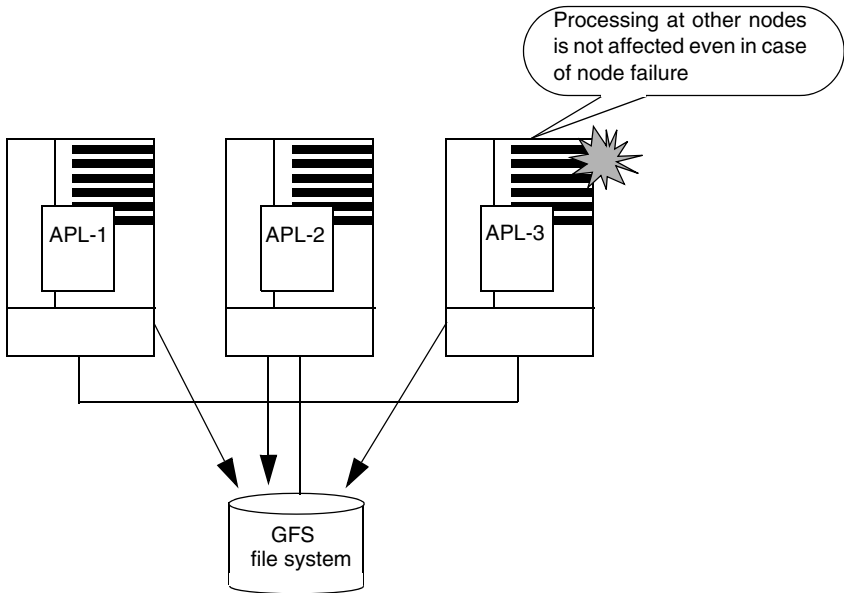


Figure 14: Operation continuation when a node is down

### Data accessibility performance

GFS enables a file system on a shared disk unit to be accessed from two or more nodes. With a conventional distributed file system, data is transferred to the client that requested access from the server on which file system data is managed by means of network communication over a LAN. The requested node directly accesses the disk unit. This reduces the network load from NFS and speeds up the response time required to read or write the request.

By allocating sequential blocks to the file data, the GFS file system enables collective I/O processing to improve the file system performance.

The Solaris GFS provides a feature that integrates multiple partitions into one file system. In the case a configuration with multiple partitions, the round-robin allocation method is used, so the file data area can be used from different partitions for each file. Therefore, the I/O load can be distributed into multiple disk units and the file system performance is improved. This function makes it easy to add the data partition described afterwards.

### Scalability (Solaris only)

With the GFS file system, the file system can be easily expanded by specifying an empty disk partition. Thus, a shortage of free space in a file system can be solved in a short time. A data partition can be added even while mounting.

#### 3.3.7.3 Benefits

GDS has the following benefits:

- The use of file cache on each system and high speed data access not using the LAN improve access performance.
- CPU load distribution of application is enabled via files on two or more systems assuring data integrity.
- A high availability system is provided by continuing file access on other node when the current node goes down.
- Area management on extent base and multi-volume (\*1) support provides high-speed file access. (\*1: multi volume is supported in Solaris only.)
- Multi-volume supports the use of large-scale file systems and addition of online volume (area extension) are enabled without disrupting or reconfiguring the file system, and it makes the resource management easy. (Solaris only)
- Operation of the file system using a GUI based Web browser interface eases environment configuration and management.

#### 3.3.8 GLS

Global Link Services (GLS) is a software product that enables high reliability communications through the use of multiple network interface cards (NICs) to create redundant multiple transmission paths to a local system. Global Link Services provides network solutions that are suitable for systems in which communications continuity is important. The benefits of Global Link Services are as follows:

- Multiple NICs can provide redundant transmission paths to offer high availability and path failure protection.
- The use of GLS means that applications can continue to run even in the event of changes in the LAN configuration, the redundant transfer route, or any network fault that may occur in a transfer route.



When using PRIMECLUSTER, you can choose the following mode for dual transfer routes: NIC switching mode - Creates a transfer route between a Switch/HUB and Solaris servers on the same network.

For details on the features of Global Link Services, see the PRIMECLUSTER *Global Link Services 4.0 Configuration and Administration Guide: Redundant Line Control Function.* and *PRIMECLUSTER Global Link Services 4.0 Configuration and Administration Guide: Multipath Function.*

### 3.3.8.1 NIC switching mode

In NIC switching mode, duplexed NICs (LAN cards) are connected on the same network and switching of the transfer route is controlled using the NIC. There are no restrictions on the devices that can be connected, and it is possible to communicate with hosts on another network. To duplex the entire communication route, it is also necessary to duplicate the network equipment, such as routers, on the transfer routes and on all hosts because the duplexing of the paths should extend up to the direct connections to switches and HUBs.

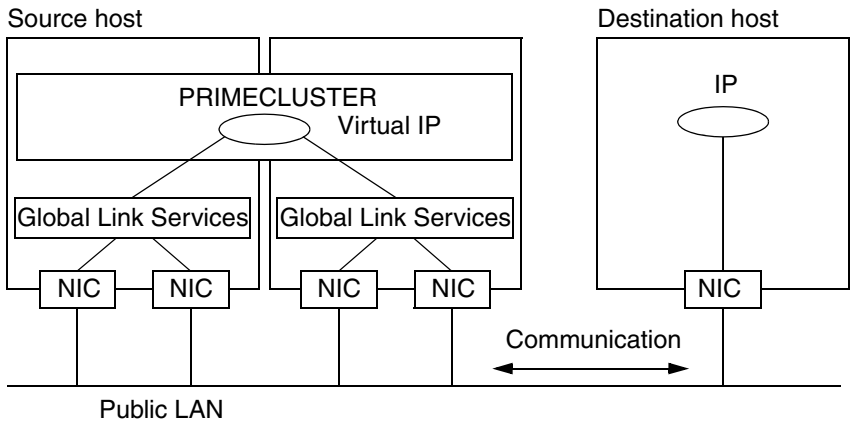


Figure 15: NIC switching mode

### 3.3.9 SNMP

The PRIMECLUSTER SNMP product collects and reports information and statistics on the status and configuration of PRIMECLUSTER modules. The information can be queried by any SNMP management station. The PRIMECLUSTER modules that currently display information by means of SNMP are as follows:

- Cluster Foundation (CF)
- Scalable Internet Services (SIS)
- Reliant Monitor Services (RMS)

The PRIMECLUSTER SNMP product consists primarily of MIBs (Management Information Bases) and subagents. MIBs are files that define what information will be provided via SNMP. The subagents are the programs that query PRIMECLUSTER components and display the information.

---

## 4 Cluster interconnect details

This chapter discusses the differences between a cluster interconnect and a network as well as the requirements of the PRIMECLUSTER cluster interconnect. It ends with a discussion of design considerations for the cluster interconnects.

This chapter discusses the following:

- The Section “Overview” provides an overview of the cluster interconnect.
- The Section “Cluster interconnect requirements” details the requirements of the PRIMECLUSTER cluster interconnect.

### 4.1 Overview

The cluster interconnect is the most fundamental part of the cluster. All of the cluster's services rely on the cluster interconnect to transport messages between nodes and to determine the state of a node by its ability to respond to heartbeat requests.

#### 4.1.1 A cluster interconnect is different from a network

The function of a cluster interconnect is different than the traditional use of a network in several ways; therefore, it is useful to think of them as separate technologies. The highest priority use of the cluster interconnect is to carry heartbeat requests and responses.

Heartbeat messages are used to determine the state of the nodes and the cluster interconnect. When a message fails to get through, the cluster software assumes that a failure has occurred and takes action to recover. However, no network is 100 percent reliable, and the PRIMECLUSTER ICF protocol tolerates errors such as lost packets or out of order delivery.

In a cluster interconnect, the physical connections are redundant so that if one fails, one or more remain to carry the messages. However, a sustained outage of all cluster interconnects results in the cluster management software taking action to recover as if a node had failed.

## 4.1.2 Network availability

Most users assume that their networks are always available. However, most networks have periods of downtime quite regularly. Network administrators need to reprogram switches and routers and perform other maintenance operations; they do this at times of low network usage and the outages last for only a few seconds. A cluster uses its interconnect continuously, 24 hours a day, 7 days a week. As a result, a routine network maintenance operation that would not even be noticed by a typical network user or application can result in a cluster recovery action that shuts down systems and moves applications.

Temporary outages can result from other causes as well. One of the most convenient improvements to Ethernet has been the speed-agile ports on hubs, switches, and routers. These allow the user to plug an Ethernet device into a port and not have to worry about speed matches between the network interface card (NIC) and the port. However, there is a cost to this technology. If a hub or switch gets too busy, it can lose its state and will have to renegotiate with the NIC. This renegotiation takes several seconds, during which time no messages are passed. As a result, a cluster interconnect on this type of device could disconnect long enough so that the cluster takes a recovery action such as shutting down a host.



To avoid renegotiation outages, it is recommended that users configure the network interface card, router, and hub at a fixed speed. If all of the redundant interconnects are unavailable for five seconds, then the software assumes that a node failure has occurred.


Another temporary outage can occur when an Ethernet device is first opened. This starts the chain of events that triggers the switch port to which the Ethernet device is attached to begin its configuration process. Some switches take over one minute for this configuration to complete. PRIMECLUSTER may interpret the lack of responses on this port incorrectly and assume no other cluster nodes are configured on this port.

Because different network configurations and devices have different requirements, PRIMECLUSTER allows the system administrator to tune the parameters associated with the cluster interconnect. Refer to the *Cluster Foundation Configuration and Administration Guide* for more details.

### 4.1.3 Interconnect protocol

PRIMECLUSTER supports either the Ethernet protocol or IP as the transport layer for the cluster interconnects. This does not mean that TCP/IP is the underlying protocol used by the cluster services. TCP/IP is designed to reliably deliver messages through a variety of obstacles such as different network types, multiple routers or switches, and failures in the network. TCP/IP works well and has been almost universally adopted as the protocol stack of choice for a wide variety of jobs. However, TCP/IP is not the right choice for all of the cluster's needs.

The ICF (Internode Communication Facility) protocol used by PRIMECLUSTER is designed specifically for cluster communications. It is a low-latency protocol that guarantees ordered delivery of messages. ICF has a lower overhead than TCP/IP, which is important for parallel applications. ICF uses the Ethernet protocol or is a service over IP (CF/IP).

 Devices that only accept TCP/IP cannot route the ICF protocol when it is configured directly on the Ethernet. In this instance, if you need to use a router, it must be a level-two router.

ICF is only usable by the CF internal components and is not available to user-level resources. To provide applications with an access to the cluster interconnect, the Cluster Interconnect Protocol (CIP) is used. CIP provides a standard TCP/IP protocol suite over ICF.

## 4.2 Cluster interconnect requirements

PRIMECLUSTER uses all significant forms of Ethernet and most devices that support TCP/IP, including mixtures of different technologies. The interconnect must be redundant; that is, there must be two or more independent connections between all of the nodes in the cluster. The connections must support all of the nodes in the cluster. Refer to the *Release Notice* as detailed in the Section "Documentation" for a list of all the Ethernet devices that were known to work at the time of release.

## 4.2.1 Redundancy

To be redundant, the cluster interconnect must use two or more independent connections and data paths. An independent connection means that the Ethernets do not share any common components. An independent data path means that the individual interconnects do not share any part of their route between nodes. An example of a redundant cluster interconnect is as follows:

- Only one port is used for each Ethernet board. If more than one port of the same board is used as a cluster interconnect, these ports would be a common point of failure.
- Connections to each host must be independent.
- No two interconnects can share the same hub, switch, or routers.

Figure 16 shows a typical four-node cluster. In this diagram, there are two hubs. Each of these hubs has their own power connection on different circuits. If the hubs were connected to the same power strip in a rack, for example, the power strip would be a single point of failure for the cluster interconnects.

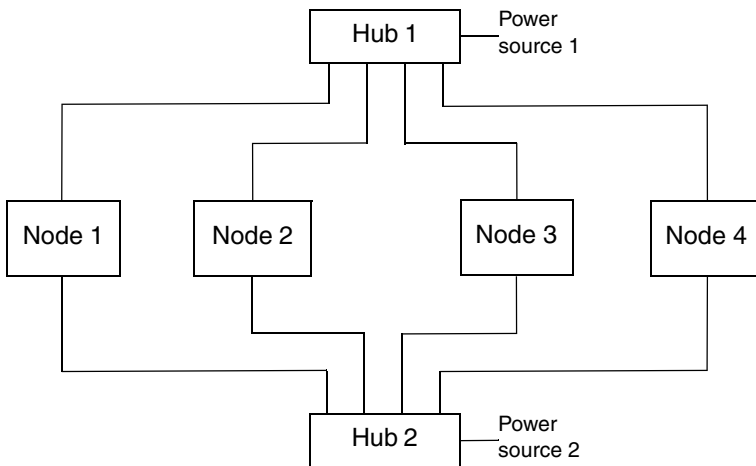


Figure 16: Typical four-node cluster

## 4.2.2 Routes

Since redundant interconnects are required for reliability, ICF is designed to use all of the available bandwidth. Each connection between two nodes in the cluster is called a route. When ICF has a message to send to another node, it chooses a route so that the message traffic is balanced between all the available routes. In the four-node cluster in Figure 16, each node has two routes to each of the other nodes.

### 4.2.2.1 Heartbeats

When everything is functioning properly in the cluster, all of the routes are available or `UP`. Every route participates in carrying message traffic if it functions at the same speed and is `UP` (mixed speed interconnects are discussed below.) In addition, every route is always used for heartbeat requests. If a certain number of heartbeat requests fail on a route, then that route is marked as `DOWN`. A `DOWN` route is never used for message traffic. However, heartbeat requests are still attempted on the `DOWN` routes, and if one succeeds, the route is returned to the `UP` state. This last behavior is sometimes called self-healing routes. The use of all the available routes for message traffic is called port aggregation or trunking.

Failures of heartbeat requests can come from several causes. When a network component fails, that route is not usable and the behavior is the same as described in the previous paragraph. When a node is `DOWN` to the only remaining route to another node, `PRIMECLUSTER` does not mark this route as `DOWN`, rather it marks the node as `LEFTCLUSTER`. Once a node is marked as `LEFTCLUSTER` or `DOWN`, no more heartbeat attempts are made on any of the routes to the node until the node rejoins the cluster.



The last route to a node is never marked `DOWN`, even if the node is in the `LEFTCLUSTER` or `DOWN` state.

Sometimes, events happen on a node that prevent the node from responding to heartbeat requests for some reason other than a failure. For example, a Fibre Channel controller can prevent other network device drivers from executing while attempting a link recovery. This can result in a false detection of node failure. To allow for some flexibility in responding to this condition, the number of missed heartbeat requests on the final route to a node before declaring the node as `LEFTCLUSTER` is tunable.

You can only adjust the number of failed-heartbeat requests that occur before declaring the node `LEFTCLUSTER`. The other parameters for ICF are not tunable. It is important that the route-detection algorithm marks routes as down as soon as possible, so that messages are switched to alternate routes without a noticeable delay. If a route is momentarily not available, for whatever reason, the self-healing mechanism will quickly reactivate the route after it becomes functional. The *Cluster Foundation Configuration and Administration Guide* has details on tuning the ICF parameters.

PRIMECLUSTER also supports the use of non-symmetric interconnects, for example, one interconnect could use gigabit Ethernet and a second fast-Ethernet. When making decisions about routing, the interconnect speed is also considered. In the previous example, the gigabit Ethernet would always be used in preference to the fast-Ethernet whenever it is available. If there are multiple interconnects at the same speed, the port aggregation is done across all the devices with the same speed. Heartbeat requests are always sent on every interconnect, independent of the interconnect speed.

### 4.2.3 Properties

When designing the cluster interconnect, it is important for users to consider the following:

- Bandwidth
- Latency
- Reliability
- Device interface
- Security

#### 4.2.3.1 Bandwidth

PRIMECLUSTER does not require much bandwidth for its own use. This is significant because the end-user's application requirements for bandwidth are much more important. PRIMECLUSTER requires less than 2 Kbits/s on each of the cluster interconnects. Since the cluster interconnects share bandwidth, the total bandwidth available to the user's application is nearly the sum of all of the bandwidth for each interconnect.



Refer to table 1 as an example of bandwidth use. Suppose that in the configuration in Figure 16 that there are two 100 Mbits/s Ethernets configured for the cluster interconnect. Assume that the available bandwidth for each cluster interconnect is 10 Mbytes/s, and assume that the end-user application needs 4.5 Mbytes/s on each node for the cluster file system and other activities. (This is an example—the actual bandwidth used by an application varies, depending on the application.)

| Item  | Bandwidth                               | Total bandwidth                 |
|---|---|---------------------------------|
| Two interconnects:<br>100 Mbits/s Ethernet                      | 10 MBytes/s                             | 20 MBytes/s                     |
| PRIMECLUSTER<br>requirements per inter-<br>connect and per node | 2 Kbits/s<br>(4 x 2 x 2 = 16 Kbits/s)   | 16 Kbits/s                      |
| User application per<br>node                                    | 4.5 MBytes/s<br>(4.5 x 4 = 18 MBytes/s) | 18 MBytes/s                     |
| Total   |   | 18.02 MBytes/s<br>90% total use |

Table 1: Example of interconnect with two 100 Mbits/s Ethernet boards

For this example, two fast-Ethernet interconnects use over 90 percent of the bandwidth.



It is recommended that an initial installation has at least 30 percent available bandwidth capacity because the latency of the interconnect is adversely affected when total use nears 100 percent.

For this example, workload and configuration, one additional fast-Ethernet interconnect should be added to provide the excess capacity. table 2 shows the same calculation with this addition.

| Item  | Bandwidth                               | Total bandwidth                 |
|---|---|---------------------------------|
| Three interconnects:<br>100 Mbits/s Ethernet                    | 10 MBytes/s                             | 30 MBytes/s                     |
| PRIMECLUSTER<br>requirements per inter-<br>connect and per node | 2 Kbits/s<br>(4 x 3 x 2 = 24 Kbits/s)   | 24 Kbits/s                      |
| User application per<br>node                                    | 4.5 MBytes/s<br>(4.5 x 4 = 18 MBytes/s) | 18 MBytes/s                     |
| Total   |   | 18.02 MBytes/s<br>60% total use |

Table 2: Example of interconnect with three 100 Mbits/s Ethernet boards

This new configuration gives a comfortable 40 percent available bandwidth margin. PRIMECLUSTER supports a maximum of eight interconnect devices.


#### 4.2.3.2 Latency

There are no absolute latency requirements in PRIMECLUSTER. As previously stated, PRIMECLUSTER relies on heartbeat requests and responses to determine that nodes or other resources are functional. When a heartbeat is not received in a preset interval, PRIMECLUSTER starts recovery actions. The intervals depend upon what PRIMECLUSTER is monitoring. The Cluster Foundation (CF) software on each node sends a heartbeat request every 200 ms on each interconnect to every other node in the cluster. By default, CF tries 50 times to get a response before a node is marked as LEFTCLUSTER.

The 200 ms interval is a reasonable choice for a maximum latency in the cluster interconnects. This interval is long enough so that a small message and response can span transcontinental distances. This interval also allows for recovery from lost packets or other errors that can cause a few requests or responses to be lost in the interconnect.


For interconnect latency, an end-user application can have a different set of requirements. For example, applications like OPS will find that 200 ms is far too long an interval for latency. This shows how users must consider the needs of the applications during the design of their cluster solution.


All commercially available Ethernet hubs and switches meet the latency requirements of PRIMECLUSTER and those of most applications. When latency is measured with hubs and fast Ethernet, typical results are between 15 and 60 microseconds for user-level applications.

 Some routers, especially in heavily used networks, can introduce excessively high latencies. For this reason, it is recommended that you do not use routers with the cluster interconnect. If routers cannot be avoided, it is essential that you address the latency concerns.


#### 4.2.3.3 Reliability

Ethernet as an interconnect technology has not shown any problems with PRIMECLUSTER. The communications protocol used by PRIMECLUSTER is ICF. ICF guarantees that messages are delivered correctly and in order to its clients. However, ICF was designed with fairly reliable communications in mind. When the cluster interconnect is reliable, ICF has very low overhead, but when it is unreliable, the overhead of ICF increases. This is similar to other protocols like TCP/IP; errors in the interconnect will result in messages being resent.

 Resending messages consumes bandwidth and increases latency and should be avoided at all times.

 An Ethernet error rate greater than 1 error per 1,000,000 bytes indicates that there is some problem with the Ethernet layer that should be investigated. (Use the command `netstat(1)` to find the error rate.)

#### 4.2.3.4 Device interface

 This section is for Solaris only.

PRIMECLUSTER depends on the DLPI (Data Link Provider Interface) for devices in Solaris. If a device does not support a DLPI interface, PRIMECLUSTER does not recognize the device as eligible for use as a cluster interconnect. In addition, the device must appear to be an Ethernet device. Some devices support TCP/IP, but are not Ethernet-type devices. Keep in mind that PRIMECLUSTER does not use TCP/IP for its cluster interconnects; rather it uses the Ethernet protocols. It is advisable to choose an interconnect device from the list of supported devices (refer to the *Release Notice* as detailed in the Section “Documentation”).

#### 4.2.3.5 Security

With the PRIMECLUSTER family of products, it is assumed that the cluster interconnects are private networks; however, it is possible to use public networks as cluster interconnects because ICF does not interfere with other protocols running on the physical media. The security model for running PRIMECLUSTER depends on physical separation of the cluster interconnect networks.



For reasons of security, it is strongly recommended not to use public networks for the cluster interconnect.

The use of public networks for the cluster interconnects allows any machine on that public network to join the cluster (assuming that it is installed with the PRIMECLUSTER products). Once joined, an unauthorized user, through the node, would have full access to all cluster services.

---

## 5 RMS

This chapter introduces the basic concepts of RMS (Reliant Monitor Services). It begins with an overview of the basic terms and concepts used to provide high availability by means of RMS, and then it goes into more detail on how RMS works.

This chapter discusses the following:

- The Section “RMS overview” describes the concepts of high availability, redundancy, and switchover.
- The Section “RMS monitoring and switchover” describes the components of RMS and explains how they work together to achieve high availability.
- The Section “RMS administration” introduces administering RMS by means of the GUI (graphical user interface) and the CLI (command-line interface).
- The Section “Customization options” discusses customizing RMS for a particular site.

### 5.1 RMS overview

RMS is a software monitor designed to provide system-level high availability (HA) to applications. A minimum of two hosts with shared access to common storage are administered from a cluster console. As the PRIMECLUSTER HA monitor, RMS uses detectors to monitor the state of components and resources for applications. If a resource or application fails, the event is noted by the detector and reported to RMS. RMS then takes the appropriate action as described in the sections that follow.

Figure 17 shows the components of a basic RMS cluster on Solaris.

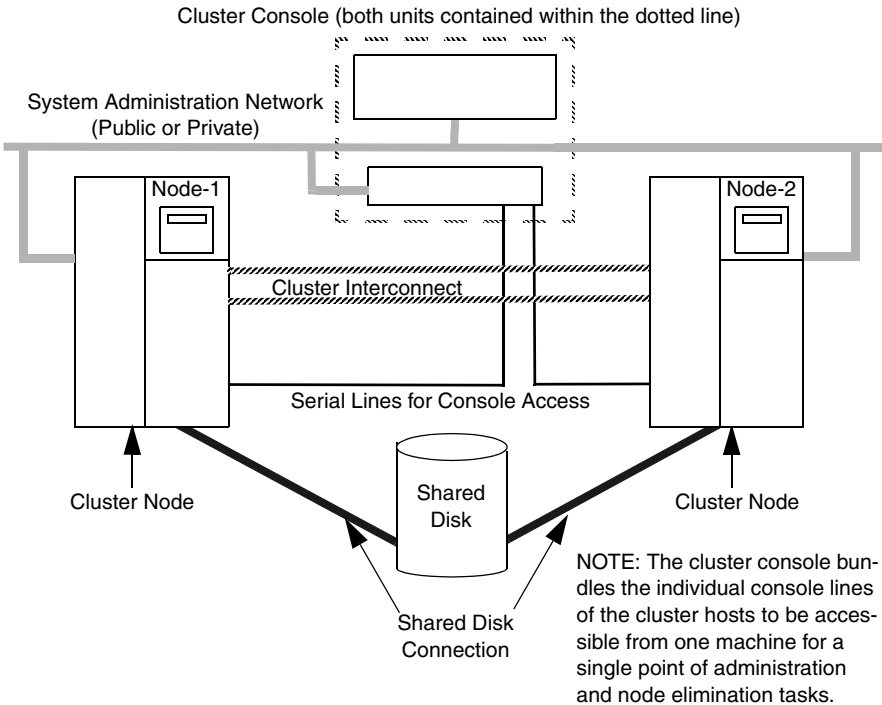


Figure 17: RMS cluster on Solaris

For RMS, high availability means *maximum availability of applications* rather than *uninterrupted availability of individual hosts*.

RMS accomplishes high availability in two ways: redundancy and switchover.

### 5.1.1 Redundancy

To provide high availability of RMS resources, RMS takes advantage of the following redundancy scheme:

- Multiple hosts, each configured to assume the resource load of any other RMS host
- Duplication of stored data using mirror disks, hardware RAID, and remote mirroring

- Multiple-path access to storage media
- Multiple communications channels dedicated to interhost communications

### Multiple hosts

An RMS configuration is made up of multiple hosts, each containing identical operating systems and RMS software. The maximum number of hosts in an RMS configuration is theoretically unlimited. Refer to the *Release Notice* as detailed in the Section “Documentation” for the qualified number of nodes.

### Shared storage

All hosts in the RMS configuration must have access to whatever data is shared. Typically, all nodes have the ability to access shared disks over a SAN. However, other access methods, such as Network Attached Storage are possible.

### RMS network

RMS uses standard TCP/IP protocols for communication between the hosts in the configuration. Because this communication is to monitor the status of other nodes and because any communication failures would be likely to endanger consistency, RMS uses the Cluster Interconnect Protocol (CIP) interface provided by PRIMECLUSTER (refer to Section “Interconnect protocol” for more information). The RMS network is layered over the PRIMECLUSTER interconnect, and they both share the same physical media.

## 5.1.2 Application switchover

RMS operates on an object-oriented basis. The objects can be almost any system component, such as virtual disks, file system mount points, processes, and so on. These objects are defined as *resources*. A resource is categorized into a grouping called a *object type*. An object type has specific properties, or attributes, which limit and define what monitoring or action can occur in relation to that resource. Resources are monitored by programs called *detectors*. This very general object-oriented design permits a high degree of flexibility in the type and level of monitoring.

Resources which are dependent on each other can be grouped together to form logical applications. Failure of any resource in such a group generally triggers a reaction by the entire application.

### 5.1.2.1 Automatic switchover

Any failure of a resource triggers a user-defined reaction. In most cases, the most significant reaction to failures is switchover.

A *switchover*, sometimes known as failover, consists of first bringing an application into a well-defined `Offline` state and then restarting the application under the control of RMS on another host. RMS supports symmetrical switchover, which means that every RMS host is able to take on resources from any other RMS host. For example, if the host that is running an application fails, RMS automatically shuts down the host where the failure occurred and redistributes its application load to another operational RMS host.

The details of performing automatic switchover are defined in user-specified scripts and configuration files that RMS accesses when a failure is recognized. The RMS wizards are generally used to create these files.

### 5.1.2.2 Manual switchover

Resources can be switched manually for such purposes as hardware maintenance. For example, in a two-host RMS configuration, all applications can be temporarily moved to one host while maintenance is performed on the other. Then all applications can be switched back to the operational host while maintenance is performed on the second host. The only impact to users might be a momentary interruption in service while the applications are being switched, and perhaps a slowdown in response time while all applications are operating on a single host.

### 5.1.2.3 IP aliasing

RMS uses IP aliasing to allow switchover of the communication link. It is possible for several IP addresses (aliases) to be allocated to one physical network interface. With IP aliasing, the user is able to continue communicating with the same IP address, even though the application is now running on another host.

### 5.1.2.4 Data integrity

RMS ensures that all resources of an application are offline on the current host before initiating a switchover. These resources are then switched online on another operational host. This behavior ensures that multiple hosts do not access the same resource, thus protecting against data loss.



In the rare event of simultaneous multiple faults, RMS protects against data corruption by preventing a switchover. This design means that in certain circumstances, switchover may be prevented entirely.

Although high availability is the goal of RMS, in any case where data might be jeopardized, data integrity takes priority over high availability.

## 5.2 RMS monitoring and switchover

The RMS software is composed of the following processes, scripts, files, and parameters that work together to maintain high availability of the RMS resources:

- Base monitor
- Configuration files
- Configuration scripts
- Detectors
- RMS environment variables

Configuration files, scripts, and environment variables can be customized, allowing switchover scenarios to be tailored for site-specific needs (for information on customization, refer to the Section “Customization options”).

### 5.2.1 Base monitor

The base monitor is the central process that monitors a host from the RMS cluster. The base monitor performs the following functions:

- Controls and coordinates all state changes in RMS
- Ensures that the appropriate action is taken if there are any problems with the monitored resources
- Obtains information concerning resources and the action to be taken from an RMS configuration file, which the system administrator creates, according to the requirements for use by RMS

The base monitor is started by the Cluster Admin GUI, or with the `hvc` command, and runs as a process under the name `bm` (base monitor) on every host in the cluster.

## 5.2.2 Configuration file

An RMS *configuration file* is a text file that is normally generated by the Wizard Tools. It provides input to the base monitor that consists of definitions of the resources that are to be monitored by RMS, including their interdependencies.

### 5.2.2.1 Interdependencies

In RMS, the terms *parent* and *child* are used to represent dependent relationships between objects and their resource objects. The term *leaf object* is used to indicate that an object in a system graph has no children. In the configuration file, the leaf object definition is at the beginning of the file. A leaf object cannot have children, while a child can be both a parent and a child.

Figure 18 helps illustrate the parent/child relationships between objects.

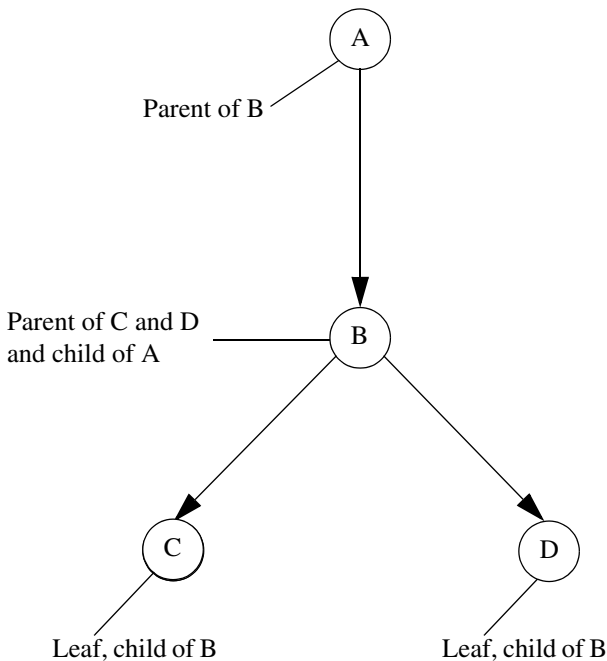


Figure 18: Configuration file interdependencies

Figure 18 shows the following relationships:

- Object A requires the resource B to function properly, making A a parent of B and B a child of A.
- Object B requires the resources C and D to function properly, making B a parent of C and a parent of D.
- Objects C and D are leaf objects because they do not have any children. They are also children of B.

### 5.2.2.2 Object types

A *object type* is a collection of similar resources monitored as a group (for example, all object types in a group using the same disk drives). Each object type has certain properties, or *attributes*, which limit or define what monitoring can occur for that type of resource. The attributes associated with a particular object type define how the base monitor acts and reacts for a resource of that object type during normal operations. Attributes commonly specify a device name or a script, and can be specified in any order in the object definition for that resource.

Some attributes are mandatory; others are optional. An example of a mandatory attribute is a device name for a `vdisk` object type. Typical optional attributes are scripts which are executed under certain conditions. Most attributes can be used for most object definitions.

Some attributes are valid only for particular object types, while some object types require that specific attributes be included in their object definitions.

### 5.2.2.3 Object definitions

A *object definition* is a statement in the configuration file, beginning with the keyword `object`, that describes a particular resource in terms that RMS understands. Specifics of an object definition include the following:

- Object type
- Resource name
- Attributes
- Child resources that the particular resource depends upon

After RMS has been installed and verified, the configuration file must be created before RMS can begin monitoring resources. If a resource does not have an object definition in the configuration file, the resource is *not* recognized by RMS.

### 5.2.3 Scripts

An RMS configuration *script* is a shell program or executable that reacts to and/or invokes a change in the state of an RMS resource.

The following states are possible for all RMS resources:

- Faulted
- Offline
- Online
- Unknown
- Wait
- Deact
- Inconsistent
- Standby

All RMS actions and reactions are executed as scripts. Without scripts, RMS is only capable of monitoring resources, not activating any changes. Scripts are identified as attributes in the object definition for the particular resource, and are run by the base monitor as needed in response to detector-reported state changes. For example, if the state of an RMS network resource changes from `Online` to `Faulted`, the base monitor responds by initiating the fault script listed in the object definition for that resource.

RMS distinguishes between scripts which are designed to change a state (request-triggered scripts) and scripts which represent a reaction to a specific state (state-triggered scripts).

Request-triggered scripts are as follows:

- `PreOnlineScript`
- `PreOfflineScript`
- `PreCheckScript`
- `OnlineScript`
- `OfflineScript`
- `OfflineDoneScript`

State-triggered scripts are as follows:

- PostOnlineScript
- PostOfflineScript
- FaultScript

For further information on resource states and scripts, refer to the *RMS Configuration and Administration Guide*.

## 5.2.4 Detectors

A *detector* is a process that monitors the state of a certain type of resource, such as mirrored disks. A detector is started by the base monitor when RMS is started.

Each detector uses an internal table to track information about one or more resources. The detector scans each resource listed in its internal table, obtains the current state of each resource, and then compares the current state to the previous state. If the current state is not the same as the previous state, the detector reports the current state to the base monitor through the detector report queue for that type of resource. Based on the information from the configuration file, the base monitor then determines, what, if any, action is necessary for the particular situation involving that resource.

Figure 19 illustrates the interactions between the base monitor and the RMS detectors.

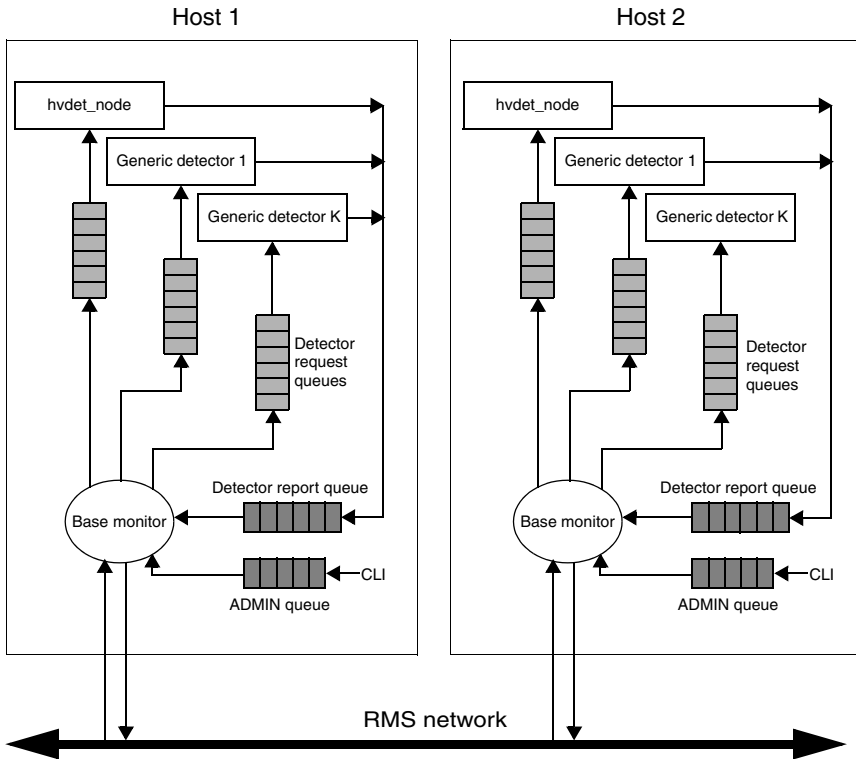


Figure 19: Base monitor interactions

### 5.2.5 RMS environment variables

*Environment variables* specify values for the base monitor during startup system events. RMS comes with a number of high availability environment variables, such as `HV_AUTOSTART_WAIT`, `RELIANT_PATH`, and so on. The default settings of these RMS environment variables can be adjusted as needed for individual clusters and applications.

## 5.3 RMS administration

You can administer RMS from the command-line interface (CLI) or from the graphical user interface (GUI). The preferred method for RMS administration is the GUI, which is called Cluster Admin.

## 5.4 Customization options

Fault detection and recovery schemes can be customized to fit the needs of each site by modifying configuration files, detectors, and scripts for the resources that are to be monitored by RMS. These modifications should be planned in advance and implemented after RMS installation is completed, and before RMS is declared operational.

### 5.4.1 Generic types and detectors

RMS provides a generic resource type for defining special resources that cannot use the system-level supplied resource types. The generic detector interface allows the definition of up to 64 custom resource types and their detectors.





---

## 6 RMS wizards

This chapter introduces the basic concepts of the RMS wizards. After a brief overview, it discusses the two main wizard products and some of the functions they provide.

This chapter discusses the following:

- The Section “Wizard overview” introduces the wizard products.
- The Section “Wizard architecture” describes the two wizard products: RMS Wizard Tools and RMS Application Wizards.
- The Section “RMS Wizard Tools” specifies the functions of the RMS Wizard Tools.
- The Section “RMS Application Wizards” details the functions of the RMS Application Wizards.

### 6.1 Wizard overview

Creating an RMS configuration for multiple customer applications can be a complex job, requiring skilled specialists for both the application and the RMS environment. RMS wizards simplify both the configuration and the operation of RMS. RMS wizards can manage the system, RMS, and applications. However, RMS, RMS Wizard Tools, and RMS Application Wizards are separate products.

### 6.2 Wizard architecture

The RMS wizards are divided into the following two main product areas:

- **RMS Wizard Tools**—Contains the wizard framework and interfaces with the RMS Base. This product simplifies the configuration setup and works with RMS and RMS Application Wizards as part of the HA (high availability) cluster environment.
- **RMS Application Wizards**—Enable an RMS HA configuration to be built for specific, customer-oriented applications. RMS Application Wizards are separately distributed software options for creating optimized HA environments for individual enterprise user applications. (Using RMS Application Wizards requires that RMS Wizard Tools be purchased and installed.)

**i** For information on the availability of the RMS Application Wizards, contact your local customer support service.

Figure 20 shows the two products and how they relate to RMS and each other. RMS Wizard Tools are a prerequisite to use RMS Application Wizards, and RMS Application Wizards are available only to authorized consultants.

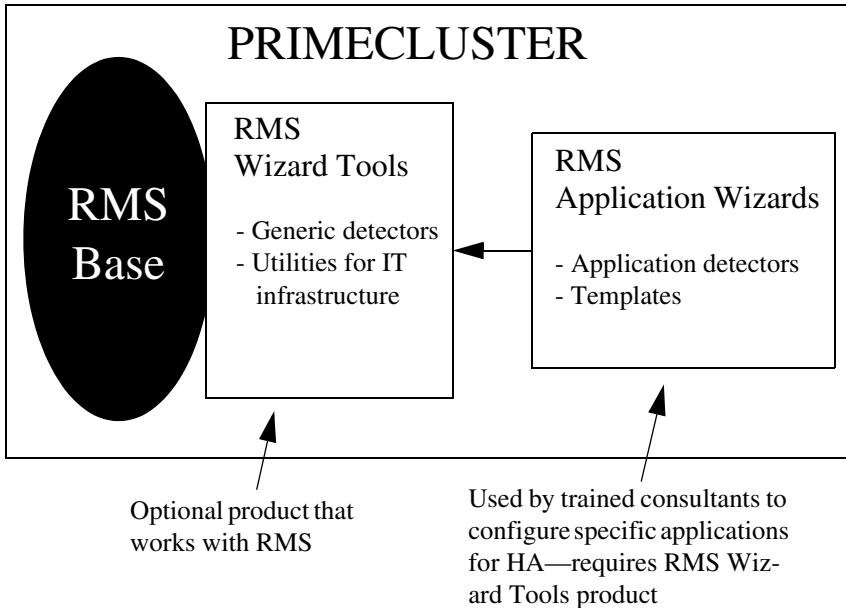


Figure 20: Wizard architecture

### 6.3 RMS Wizard Tools

The optional RMS Wizard Tools is a product that creates and helps maintain HA RMS configurations. RMS Wizard Tools provides the following functions:

- Works in conjunction with RMS Application Wizards to create RMS configurations
- Runs configured applications on a host within the cluster
- Copies and distributes complete configurations
- Prevents or allows configuration changes

To support additional HA configurations, there are some utility wizards offered as part of the RMS Wizard Tools that manage some system standard resources and applications not covered by a specific application wizard, such as the following:

- Mounting/unmounting file systems
- Installing/uninstalling IP aliases
- Configuring/unconfiguring virtual disks

### 6.3.1 Shared-storage applications

Shared-storage application wizards configure RMS for different types of disk storage. For example, you can configure software-based solutions such as GDS or Veritas VxVM, or you can configure hardware solutions like EMC. These wizards can also configure multiple SAN (Storage Area Network) modules, allowing you the capacity to stack modules.

## 6.4 RMS Application Wizards

The RMS Application Wizards configure an RMS configuration for a unique enterprise application. The wizard produces a complete RMS configuration for the user, which is then used by RMS Wizard Tools and RMS. These wizards are distributed separately from the RMS Wizard Tools software.



For information on the availability of the RMS Application wizards, contact your local customer support service or refer to the RMS wizards documentation package (refer to the Section “Documentation”).

RMS Application Wizards allow the administrator to adapt to the user’s need for HA for a specific application. RMS Application Wizards contain predefined application-specific detectors, rules, and templates for entering user parameters for the application.

Application-specific detectors monitor RMS resources for a particular application. Failure of an RMS resource triggers a user-defined response. See the Section “Detectors” for more information on how detectors work within RMS.



---

# 7 SIS

This chapter introduces the basic concepts and components of the Scalable Internet Services (SIS). After a brief overview, it explains how the SIS components function and the services that they provide.

This chapter discusses the following:

- The Section “SIS overview” introduces the concepts of the SIS technology.
- The Section “SIS design” shows an example of how a SIS cluster might function.
- The Section “VIP load-balancing algorithms” details the algorithms available for users of SIS.
- The Section “Proxy addresses” describes the uses and capabilities of proxy addresses.
- The Section “Private addresses” describes the uses and capabilities of private addresses.
- The Section “Failover” discusses the concepts and capabilities of backup nodes.

## 7.1 SIS overview

SIS provides scalable and fault tolerant network services based on the underlying PRIMECLUSTER technology. SIS enables the PRIMECLUSTER to act as a scalable, reliable, and easily managed network server system. The nodes in a SIS cluster are accessed via one or more Virtual Interface Provider (VIP) addresses and appear to the clients as a single network server.

Nodes in a SIS cluster share the load for different services. With SIS, users can configure load sharing per service, and they can fine-tune unique application and site needs with a variety of load-balancing algorithms.

SIS eliminates single points of failure and ensures availability as follows:

- If any of the SIS nodes or services fail, SIS schedules requests around the failed nodes.
- If any of the SIS modules fail, they gracefully recover.
- After a failed node restarts and you have started SIS on it, the node will seamlessly join the cluster to restore maximum performance.

### 7.1.1 Features

SIS has the following features:

- Single IP target address for all external users
- Ease of adding nodes and services
- TCP and UDP services configurable on a per-port basis
- Many load-balancing algorithms available
- Seamless handling of node failures, service failures, and component failures
- Flexible backup-node management
- Proxy addresses for cluster use
- Private communication between nodes
- Software-based solution
- GUI-based configuration and administration interface

The sections that follow discuss the various types of SIS modules.

### 7.1.2 Service nodes

Service nodes offer network services such as web services and directory services. If a service node fails, services are scheduled around it. When a failed node comes back up, it rejoins the SIS cluster seamlessly.

### 7.1.3 Gateway nodes

There is one gateway node per VIP address. All incoming packets are received by this node and forwarded to the selected service node, depending on the scheduling algorithm for the service. If a gateway node fails, another node seamlessly assumes the role of the failed gateway node.

### 7.1.4 Primary database node

The primary database node keeps the static and dynamic data of the SIS cluster. The static information may include the list of nodes in the SIS cluster, the VIP address and services offered, and the scheduling algorithms. The dynamic information includes the current list of connections and the current status of the SIS cluster.

### 7.1.5 Backup database node

The backup database node assumes the role of the primary database node when the primary database node fails for any reason. There can be more than one backup database node. Each backup database node contains the static configuration details, and SIS collects the dynamic configuration items from all the available nodes; therefore, network disruption is kept to a minimum.

## 7.2 SIS design

In Figure 21, the SIS cluster consists of one gateway node and two service nodes. The steps that follow give an example of how a SIS cluster performs after receiving a request from a client.

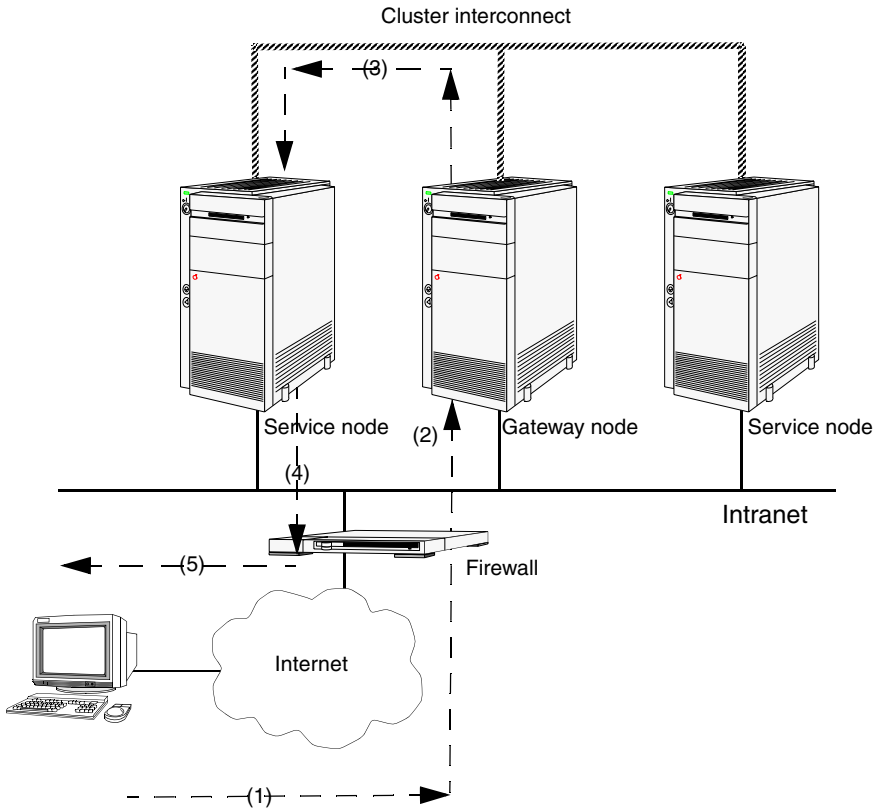


Figure 21: SIS client/server illustration

Referring to Figure 21, the SIS cluster functions as follows:

1. Client requests services through the network, seeing only one IP address.
2. The gateway node receives the initial connection request.
3. The gateway node chooses a service node, based on the scheduling policy and the availability services. Only the initial request requires this scheduling. Subsequent data packets are routed directly to the service node.
4. The service node processes the request and responds directly to the client.
5. The client receives the response from the network.



## 7.3 VIP load-balancing algorithms

SIS provides the following load-balancing algorithms, which can be configured on a per-port basis:

- Keep local—The gateway node answers the request without any processing overhead.
- Client based—The service node is chosen based on the client's IP address.
- System load—The system with the minimum system load is chosen. The system load is calculated by SIS using an internal algorithm.
- Round robin—All available nodes are chosen in a circular way.
- Spill over—If the system load on all primary nodes is equal or greater than the configured threshold, a backup node is chosen to lessen the load.
- Weighted connection count—SIS chooses the system with the least number of connections. Each node may be assigned a weight, and the connection count is normalized according to this weight.

## 7.4 Proxy addresses

Proxy addresses can be defined to assign one or more virtual addresses to a node. Since proxy addresses have failover capabilities, they also provide high availability. The possible uses are as follows:

- Co-hosting multiple addresses to one node
- Assigning external connectivity to nodes that do not have connections to the Internet
- Allocating backup nodes to a node

## 7.5 Private addresses

Private addresses can be configured on each of the available nodes in a SIS cluster. These addresses are commonly used for inter-node communication by means of the interconnect and the IP protocol. Private addresses are secure, fast, and use the redundant interconnect to assure high availability.

## 7.6 Failover

One or more nodes can be service nodes for each service. Each of these service nodes can have one or more backup nodes. The backup nodes can be either hot standby or nodes which are doing something else. If a node or nodes fail, then the configured backup nodes replace the failed node or nodes as determined by the configuration file.

---

# Glossary

## 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.

## Administrative LAN

In PRIMECLUSTER configurations, an Administrative LAN is a private local area network (LAN) on which machines such as the System Console and Cluster Console reside. Because normal users do not have access to the Administrative LAN, it provides an extra level of security. The use of an Administrative LAN is optional.

See also *public LAN*.

## API

See *Application Program Interface*.

## application (RMS)

A resource categorized as a `userApplication` used to group resources into a logical collection.

## 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 RMS Application Wizards to create object definitions for a specific type of application.

## Application Wizards

See *RMS Application Wizards*.

## 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 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*.

### **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*.

### **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.

**CF**

See *Cluster Foundation*.

**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*.

**Cluster Foundation**

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 PRIMECLUSTER communications.

**Cluster Join Services** (CF)

This PRIMECLUSTER module handles the forming of a new cluster and the addition of nodes.

**configuration file** (RMS)

The RMS configuration 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 base monitor.

**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)*.

**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)*.

**ENS (CF)**

See *Event Notification Services (CF)*.

**environment variables (RMS)**

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*.

**GFS**

See *Global File Services*.

**GLS**

See *Global Link 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.

**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**

This concept applies to the use of redundant resources to avoid single points of failure.

**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*.

**JOIN (CF)**

See *Cluster Join Services (CF)*.



**keyword**

A word that has special meaning in a programming language. For example, in the 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 base monitor, wizards, and detectors can have their own log files.

**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.

### **Meta Data Server**

GFS daemon that centrally manages the control information of a file system (meta-data).

### **mount point**

The point in the directory tree where a file system is attached.

### **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 which is a member of a cluster. A computer node is the same as a computer.

### **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)**

In the configuration file or a system graph, this is a representation of a physical or virtual resource.

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. The keyword associated with an object definition is *object*.

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.

**OPS**

See *Oracle Parallel Server*.

**Oracle Parallel Server**

Oracle Parallel Server allows access to all data in a database to users and applications in a clustered or MPP (massively parallel processing) platform.

**OSD** (CF)

See *operating system dependent (CF)*.

**parent** (RMS)

An object in the configuration file or system graph that has at least one child.

See also *child (RMS)*, *configuration file (RMS)*, *system graph (RMS)*.

**primary node** (RMS)

The default node on which a user application comes online when RMS is started. This is always the nodename of the first child listed in the `userApplication` object definition.

### **private network addresses**

Private network addresses are a reserved range of IP addresses specified by the Internet Assigned Numbers Authority. They may be used internally by any organization but, because different organizations can use the same addresses, they should never be made visible to the public internet.

### **private resource (RMS)**

A resource accessible only by a single node and not accessible to other RMS nodes.

See also *resource (RMS)*, *shared resource*.

### **queue**

See *message queue*.

### **PRIMECLUSTER services (CF)**

Service modules that provide services and internal interfaces for clustered applications.

### **redundancy**

This is the capability of one object to assume the resource load of any other object in a cluster, and the capability of RAID hardware and/or RAID software to replicate data stored on secondary storage devices.

### **public LAN**

The local area network (LAN) by which normal users access a machine.

See also *Administrative LAN*.

### **Reliant Monitor Services (RMS)**

The package that maintains high availability of user-specified resources by providing monitoring and switchover capabilities.

### **remote node**

A node that is accessed through a telecommunications line or LAN.

See also *local node*.

### **remote node**

See *remote 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 Application Wizards**

RMS Application Wizards add new menu items to the RMS Wizard Tools for a specific application.

See also *RMS Wizard Tools*, *Reliant Monitor Services (RMS)*.

**RMS commands**

Commands that enable RMS resources to be administered from the command line.

**RMS configuration**

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 Tools**

A software package composed of various configuration and administration tools used to create and manage applications in an RMS configuration.

See also *RMS Application Wizards*, *Reliant Monitor Services (RMS)*.

### **SAN**

See *Storage Area Network*.

### **Scalable Internet Services (SIS)**

Scalable Internet Services is a TCP connection load balancer, and dynamically balances network access loads across cluster nodes while maintaining normal client/server sessions for each connection.

### **scalability**

The ability of a computing system to dynamically handle any increase in work load. Scalability is especially important for Internet-based applications where growth caused by Internet usage presents a scalable challenge.

### **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)*.

### **shared resource**

A resource, such as a disk drive, that is accessible to more than one node.

See also *private resource (RMS)*, *resource (RMS)*.

### **simple virtual disk**

Simple virtual disks define either an area within a physical disk partition or an entire partition. (Applies to transitioning users of existing Fujitsu Siemens products only.)

See also *striped virtual disk*, *virtual disk*.

**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)*.

**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.

**striped virtual disk**

Striped virtual disks consist of two or more pieces. These can be physical partitions or further virtual disks (typically a mirror disk). Sequential I/O operations on the virtual disk can be converted to I/O operations on two or more physical disks. This corresponds to RAID Level 0 (RAID0). (Applies to transitioning users of existing Fujitsu Siemens products only.)

See also *simple virtual disk*, *virtual disk*.

**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)*.

**system graph (RMS)**

A visual representation (a map) of monitored resources used to develop or interpret the 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**

With virtual disks, a pseudo device driver is inserted between the highest level of the Solaris logical Input/Output (I/O) system and the physical device driver. This pseudo device driver then maps all logical I/O requests on physical disks. (Applies to transitioning users of existing Fujitsu Siemens products only.)

See also *simple virtual disk*, *striped virtual disk*.

**Web-Based Admin View**

This is a common base to utilize the Graphic User Interface of PRIMECLUSTER. This interface is in Java.

**wizard (RMS)**

An interactive software tool that creates a specific type of application using pretested object definitions. An enabler is a type of wizard.



---

# Abbreviations

**AC**

Access Client

**API**

application program interface

**bm**

base monitor

**CCBR**

Cluster Configuration Backup/Restore

**CF**

Cluster Foundation

**CIM**

Cluster Integrity Monitor

**CIP**

Cluster Interconnect Protocol

**CLI**

command-line interface

**CRM**

Cluster Resource Management

**DLPI**

Data Link Provider Interface

**ENS**

Event Notification Services

**GDS**

Global Disk Services

**GFS**

Global File Services

## Abbreviations

---

|             |                                  |
|-------------|----------------------------------|
| <b>GLS</b>  | Global Link Services             |
| <b>GUI</b>  | graphical user interface         |
| <b>HA</b>   | high availability                |
| <b>ICF</b>  | Internode Communication Facility |
| <b>I/O</b>  | input/output                     |
| <b>JOIN</b> | cluster join services module     |
| <b>LAN</b>  | local area network               |
| <b>MDS</b>  | Meta Data Server                 |
| <b>MIB</b>  | Management Information Base      |
| <b>NIC</b>  | network interface card           |
| <b>NSM</b>  | Node State Monitor               |
| <b>OPS</b>  | Oracle Parallel Server           |
| <b>OSD</b>  | operating system dependant       |
| <b>PAS</b>  | Parallel Application Services    |

**PRIMECLUSTER SF**

PRIMECLUSTER Shutdown Facility

**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

**VIP**

Virtual Interface Provider



---

# Figures

|            |   |    |
|------------|---|----|
| Figure 1:  | Typical two-node cluster . . . . .                    | 8  |
| Figure 2:  | Cluster partition in two-node cluster . . . . .       | 11 |
| Figure 3:  | Diagram of a typical PRIMECLUSTER setup . . . . .     | 18 |
| Figure 4:  | PRIMECLUSTER framework overview . . . . .             | 19 |
| Figure 5:  | MA normal operation . . . . .                         | 26 |
| Figure 6:  | MA operation in the event of node failure . . . . .   | 27 |
| Figure 7:  | Node recovery . . . . .                               | 28 |
| Figure 8:  | Single cluster console configuration . . . . .        | 30 |
| Figure 9:  | Distributed cluster console configuration . . . . .   | 31 |
| Figure 10: | RMS resource monitoring . . . . .                     | 33 |
| Figure 11: | Example of a SIS cluster . . . . .                    | 35 |
| Figure 12: | SAN (Storage Area Network) . . . . .                  | 37 |
| Figure 13: | Mirroring between disk units (Solaris only) . . . . . | 38 |
| Figure 14: | Operation continuation when a node is down . . . . .  | 41 |
| Figure 15: | NIC switching mode . . . . .                          | 43 |
| Figure 16: | Typical four-node cluster . . . . .                   | 48 |
| Figure 17: | RMS cluster on Solaris . . . . .                      | 56 |
| Figure 18: | Configuration file interdependencies . . . . .        | 60 |
| Figure 19: | Base monitor interactions . . . . .                   | 64 |
| Figure 20: | Wizard architecture . . . . .                         | 68 |
| Figure 21: | SIS client/server illustration . . . . .              | 74 |



---

# Index

## A

- administering
  - Cluster Admin 24
  - RMS 65
- allowing configuration changes 68
- Application Wizards
  - See* RMS Application Wizards
- applications
  - monitored resource 32
  - PAS 35
- assigning external connectivity 75
- attributes
  - mandatory 61
  - node 61
  - object properties 57
  - optional 61
- availability 20
  - design feature 21
  - high 8

## B

- backup database node 73
- bandwidth, interconnect 50
- base monitor
  - description 59
  - for central monitoring 59

## C

- CF
  - See* Cluster Foundation
- CF/IP 47
- client-based algorithm 75
- cluster
  - architecture 17
  - partition 10
  - SIS 34
- Cluster Admin 22, 24
  - CF 24
  - RMS 65
- cluster consoles
  - multiple 29

## SCON 29

software 29

*See also* single console

## Cluster Foundation 22, 44

Cluster Admin 24

driver 23

heartbeats 52

cluster interconnect 9, 45

properties 50

reliability 53

sustained outage 45

*See also* interconnects

## clustering 7

### clusters

parallel databases 35

SIS 71

configuration file 59, 60

configuration scripts 62

### configurations

copying 68

creating 68

preventing changes 68

SCON 29

supporting additional 69

### configuring

allowing changes 68

Cluster Admin 24

hub 46

copying complete configurations 68

creating RMS configurations 68

customizing RMS 65

## D

### data

integrity 9, 21

Data Link Provider Interface 53

### detectors 63

application specific 69

generic 65

device and network access 18

diagnostics services 24

## Index

---

- distributed
  - computing 7
  - environment 16
- DLPI
  - See* Data Link Provider Interface
- documentation
  - additional 2
  - PRIMECLUSTER 2
- DOWN, state 49
- drivers
  - CF 23
  - network device 49
  - pseudo device 23
- duplexed network interface cards 43
- E**
- ENS
  - See* Event Notification Services
- environment variables 64
- Ethernets
  - error rate 53
  - fast 50
  - Gigabit 50
  - network availability 46
- Event Notification Services 24
- F**
- failover 58, 76
- features
  - availability 21
  - data integrity 21
  - scalability 20
- Fibre Channel controller 49
- file systems
  - high availability 40
  - mount points 57
  - network 32
  - update log 40
- files
  - cluster access 9
  - clusterwide system 20
  - configuration 59, 61, 65
  - custom 65
  - MIBs 44
  - RMS software 59
  - same source 23
  - user-specified configuration 58
- G**
- gateway nodes 73
- Gigabit Ethernet 50
- Global Disk Services 22
- Global File Services 22
- Global Link Services 22
  - NIC switching mode 43
  - redundant NICs 42
- GLS
  - See* Global Link Services
- guaranteed data integrity 20, 21
- H**
- heartbeats 45
  - request failures 49
  - requests 45, 52
  - responses 45
  - RMS 9
- high availability 8
  - clustering 7
  - file systems 40
  - multiple NICs 42
- host elimination 29
- hub, configure 46
- I**
- ICF
  - See* Internode Communication Facility
- installing aliases 69
- interconnects
  - bandwidth 50
  - fast switching mode 43
  - latency 52
  - NIC switching mode 43
  - non-symmetric 50
  - protocol 47
  - See also* cluster interconnect
- interface for cluster administration 22



- 
- Internode Communication Facility 23, 47
  - IP aliasing 58
  - J**
  - JOIN
    - See* join services
  - join services 24
  - K**
  - keep local algorithm 75
  - L**
  - latency 52
  - LEFTCLUSTER 80, 83, 84
  - LEFTCLUSTER, state 49
  - link recovery 49
  - load-balancing algorithms 75
  - M**
  - Management Information Bases 44
  - manual switchovers 58
  - memory allocations 18
  - MIBs
    - See* Management Information Bases
  - modular components 22
  - modularity 20
  - monitoring
    - agent 14, 25
    - applications 9
    - components 9
  - multiple network interface cards 42
  - N**
  - network interface cards
    - configuring 46
    - multiple 42
    - speeds 46
  - networks
    - availability 46
    - configuring NIC 46
    - file systems 32
    - interfaces 32
    - NIC switching mode 43
    - NICs
      - See* network interface cards
    - nodes
      - assigning weight 75
      - backup 76
      - backup database 73
      - client IP address 75
      - communication 75
      - definition 61
      - gateway 73
      - monitored resource 32
      - multiple proxy addresses 75
      - primary database 73
      - service 72
      - type 61
    - non-symmetric interconnects 50
  - O**
  - operating system dependant 23
  - operations 24
  - OSD
    - See* operating system dependant
  - OS-independent interface 23
  - overview 44
  - P**
  - Parallel Application Services 22
    - description 35
    - modularity 20
  - PAS
    - See* Parallel Application Services
  - path failure protection 42
  - platform independence 20
  - preventing configuration changes 68
  - primary database node 73
  - PRIMECLUSTER modules 22
  - protecting
    - data integrity 9
  - proxy addresses 75
  - pseudo device driver 23, 90
  - R**
  - RAID 89
-

## Index

---

- redundancy scheme 56
- redundant
  - cluster interconnect 48
  - components 8
- reliability, interconnects 53
- Reliant Monitor Services 44
- resources
  - automatic switchover 58
  - detectors 63, 65
  - heartbeats 52
  - high availability 56
  - manual switchover 58
  - monitoring 62
  - states 62
  - switchover 57
- RMS 20, 22, 32
  - administering 65
  - configuration script 62
  - environment variables 64
  - HA manager 9
  - overview 59, 64
  - wizards 67, 69
- RMS Application Wizards 15, 34, 67
  - components 69
  - detectors 69
- RMS Wizard Tools 15
  - description 34, 67
  - functions 68
- round robin algorithm 75
- routers
  - configuring 46
  - latencies 53
- S**
- SAN
  - See* Storage Area Network
- SAP R/3 15
- scalability 7, 20
  - basic types of applications 16
  - network services 71
  - quantity of nodes 20
  - SIS 71
- scalable applications 16
- Scalable Internet Services 20, 34, 44
  - architecture 22
  - description 34
  - network services 71
- scripts
  - optional attribute 61
  - RMS configuration 62
- service nodes 72, 76
- services
  - parallel application 35
  - scalable Internet 22
- Simple Network Management Protocol 22
- simple virtual disks 88
- single console
  - configuration 30
  - software 29
  - See also* cluster consoles
- SMAWRscon 29
- spill over algorithm 75
- spin locks 18
- states
  - controlling changes 59
  - detectors 63
  - DOWN 49
  - LEFTCLUSTER 49
  - monitoring changes 59
  - Offline 58
  - resources 62
  - scripts 62
  - UP 49
- Storage Area Network
  - high availability 8
  - wizards 69
- subagents 44
- sustained outage 45
- switchover 58
- synchronizations 18
- system load algorithm 75
- U**
- uninstalling aliases 69
- UP, state 49
- update log 40
- utility wizards 69

**V**

virtual disks  
    simple 88

**W**

Web-Based Admin View 22  
Weighted Connection Count algorithm  
    75  
Wizard Tools  
    *See* RMS Wizard Tools  
wizards  
    RMS 15  
    RMS Application Wizards 69  
    RMS Wizard Tools 68  
    utility 69



Fujitsu Siemens Computers GmbH  
User Documentation  
33094 Paderborn  
Germany

Comments  
Suggestions  
Corrections

**Fax: (++49) 700 / 372 00001**

email: [manuals@fujitsu-siemens.com](mailto:manuals@fujitsu-siemens.com)  
<http://manuals.fujitsu-siemens.com>

---

Submitted by

---

Comments on PRIMECLUSTER™  
Concepts Guide (Solaris, Linux)





Fujitsu Siemens Computers GmbH  
User Documentation  
33094 Paderborn  
Germany

Comments  
Suggestions  
Corrections

**Fax: (++49) 700 / 372 00001**

email: [manuals@fujitsu-siemens.com](mailto:manuals@fujitsu-siemens.com)  
<http://manuals.fujitsu-siemens.com>

---

Submitted by

---

Comments on PRIMECLUSTER™  
Concepts Guide (Solaris, Linux)



