Rose HA Reference Manual

Revision Record

Date	Description
July 1995	First published for the Rose HA 3.1 release.

Version Number: Rose HA 3.1 Release

If you have comments about this manual, please send your comments to:
Department of Rose HA
Rose Datasystems, Inc.
1580 Oakland Road C-108
San Jose, CA 95131

Rose HA Reference Manual

Contents

Introduction

Information is an important asset for a company in today's market. Computer systems must provide reliable and timely information and services to allow personnel to make informed decisions that are crucial to the daily operations of modern business.

Despite the rapid evolution of both computer hardware and software, numerous failure conditions can occur. These failures result in loss of data, time, and productivity. The availability of information becomes critical to the successful operation of a company's computer operations. Methods must be used to minimize exposure to as many of these failure conditions as possible.

The following subsections describe Rose HA, which provides the complete high availability solution needed by companies to protect critical computer applications, networks, and hardware from becoming unavailable due to failures.

Definition of system availability

Availability of critical information services is affected by both scheduled and unscheduled system downtime. Although scheduled downtime for system maintenance and upgrades is inevitable, it is fatal to information services that are considered non-interruptible.

In addition, unscheduled downtime is unpredictable and needs to be minimized. Events such as human errors, operating system failures, computer hardware failures, and network failures usually are the cause for most unscheduled downtimes.

To describe methods of preventing these failures, it is necessary to define the following types of systems: normal availability, fault tolerance, and high availability.

Normal availability systems (NASs)

Normal availability systems (NASs) are general purpose computer hardware and software systems that have no hardware redundancy or software enhancement to provide fault processing recovery. They require manual, human intervention to identify and correct/repair the failed component(s) and restart the system before normal operations are resumed.

Fault Tolerance systems

Fault tolerance systems consist of proprietary, expensive and tightly coupled duplicated systems. Fault handling capabilities are integrated into and are a function of the operating system. These systems have spontaneous and fully automatic response to system failures and provide uninterrupted services.

High availability systems

High availability systems are loosely-coupled NASs with redundant hardware components that are managed by software. This software provides fault detection and correction procedures to maximize the availability of the critical services and applications provided by that system.

High availability systems require no manual, human intervention to identify a failed component, execute a procedure to avert a system failure and provide notice of the averted failure. This configuration minimizes the possibility of immediate data loss and service interruption.

There are two high availability models for client/server architectures: the replicated services model and the failover model.

The replicated services model uses distributed applications and distributed databases on multiple servers in the LAN/WAN environment, and the data is replicated to some or all of the servers. When a server failure occurs, the data and applications are accessible from an alternate server.

The failover model uses duplicate server hardware configurations in which one server has the role of a active server for data and application services and the other is a standby server that monitors the state of the active server. When the standby server detects a hardware or software failure (that is, a failover event) on the active server, it takes over the role and identity of the active server.

Rose HA product description

Rose HA software is a high availability system that uses the failover model to deliver a highly reliable solution for building high availability for SPARC-based client/server environments.

Rose HA allows a site to provide a fully automatic failsafe system for NFS file servers, DBMS/transaction processing servers, and related applications, including support for multi-ported disk arrays and mirrored disk subsystems. If a fault is detected, Rose HA can either retry jobs or failover to a standby server.

The goal of Rose HA is to maintain open systems compatibility with SPARC-compliant architecture. Rose HA must provide reliable accessibility to applications and data through automatic failover processing, and GUI administration and alerting facilities. It must be flexible enough to adapt for differences in enterprise implementation requirements and future scalability.

Open Systems compatibility

Rose HA retains the performance, cost effectiveness, and technology advantages of Open Systems Architecture. It is compatible with existing services native to Solaris (for example, NFS, **telnet**, **ftp**, and so on.) and applications such as Sybase and other common hardware and software products that are commonly available through Sun Microsystems and third party vendors.

Reliability and accessibility

When a failure event occurs, Rose HA typically detects the failure in 5 seconds or less, and 10 to 120 seconds for failover to initiate. The failover processing occurs automatically without rebooting the standby server. This provides the highest data and service availability in a distributed SPARC architecture, client/server environment. There is no single point of failure to prevent access to the data and application services provided by the system.

Automatic resumption of services

The entire Rose HA failover process executes automatically. No human interaction is required for failure detection and system failover processing. Network identity and services of the active server are transferred to the standby server automatically. After a failover event has occurred, clients of the failed active server have access to the same data and applications automatically, without reconfiguration or system administrator intervention.

Administration

The Rose HA administration interface allows easy installation, configuration and real-time monitoring of Rose HA through a graphical user interface (GUI). It provides functions that can synchronize all the binaries and configuration files on a defined Rose HA host group. Real-time monitoring of the host group is presented as

graphical objects in an X-Windows display. Detailed descriptive information and system logs for all Rose HA hosts can be monitored on any desired host.

Flexibility

Rose HA is flexible enough to be integrated into an existing network, and is easily adapted to new configurations and modifications within the network environment. Investment into various hardware configurations depends upon the degree of desired redundancy and availability needed for a site.

Scalability

As a company grows, the capacity of its computer systems expands to meet the needs of growing volume of applications and information. Rose HA is capable of supporting multiple active servers and standby servers. The more active servers available means that the tasks can be distributed them. This results in better overall performance.

Rose HA Technical Overview

The following subsections describe a typical Rose HA configuration and the components that make up the configuration, including the following topics:

- A typical Rose HA configuration
- Active and standby servers
- The type of clients supported by Rose HA
- Modes of communication for Rose HA
- Storage devices used by Rose HA

The remaining subsections describe the failover detection process, including a definition of the HA manager and agents used to detect failover events, and how the processing of a failover event occurs.

Typical Rose HA configuration

Rose HA is capable of managing redundant servers, network communication links, network adapters, shared disk subsystems and SCSI disk adapters to achieve high availability. As shown in Figure 1, a typical Rose HA configuration consists of two SPARC servers, each with two SCSI interfaces and two Ethernet interfaces.

The servers are connected to an external disk subsystem that can be a single disk or a RAID disk array. The servers are also connected to two or more networks. One of the networks is a private connection shared between the two servers, which is used for exchanging status information between the servers. The other networks are the public networks that provide connection to the client workstations for services, data and applications.

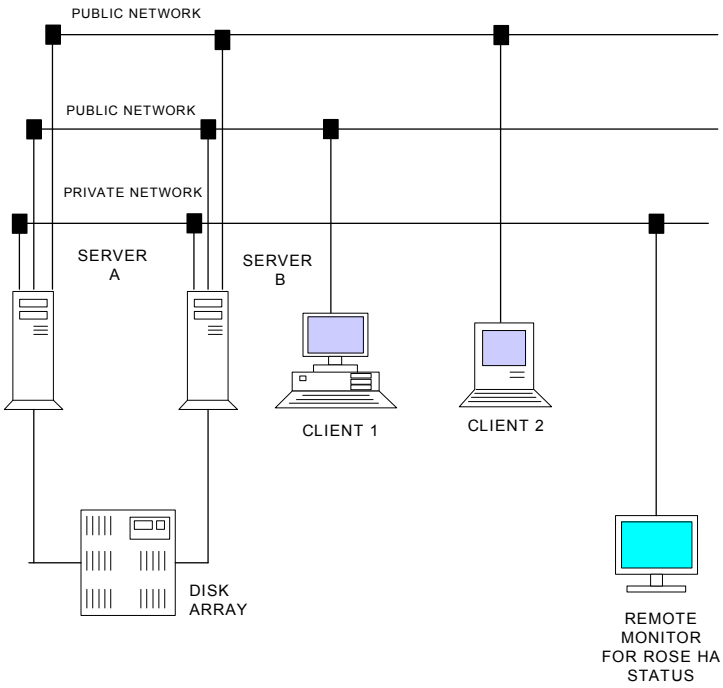

Figure 1. Typical Rose HA configuration

Active and standby servers

In a typical configuration, one of the servers is configured as an active server. An active server is the computer system that provides critical service, data, and/or applications to the client workstations.

The other server (or servers) is configured as the standby server. A standby server is a computer system that is configured to take over the role of the active server in the event the active server has a failover event. A standby server can be dedicated or non-dedicated. It can also be an active server.

The main function of a dedicated standby server is to wait for a failover event and take over the role of the active server. When configured as a non-dedicated standby server, it provides services, data, and/or applications to clients, as well as waits for a failover event to occur on the active server.

Multiple, non-dedicated standby servers can be configured to divide the workload of an active server that has experienced a failover event. The standby servers can take over the workload of the failed active server in a predefined scheme, which can be configured by the system administrator. This allows additional redundancy to be configured into the environment.

For example, if one of the standby servers fails to react properly to a failover event, another standby server can detect the failure and take over the workload for the failed standby server. This configuration allows for the standby server to have the role of the active server.

Clients

Client workstations are network nodes and/or terminals that access applications and/or services provided by the active servers. They can be Intel-based PCs using NFS

and RDBMS application services, X terminals, other UNIX servers using network time-sync services, DNS, NIS, or NFS mounted filesystems, network print devices, network modem pools, and so on.

Communication link

The mode of communication between client and server processes is TCP/IP. The site can define the physical delivery medium that best fits the requirements of the site. These links include Ethernet (IEEE 802.3), Token Ring (IEEE 802.5), FDDI, or Asynchronous SLIP and X.25 connections.

Rose HA requires two network links between the active and standby servers: a private network and a public network. The private network is used for a dedicated communication link between the active and standby servers for exchanging heartbeat messages that inform the machines of each other's HA agent and manager status. This private network can be an asynchronous communication link if the heartbeat link is point-to-point.

The public network is used for clients to access the applications and/or services provided by the active server. This is the "normal" path used for the general access point to these applications and/or services.

Storage device configuration

The storage devices used most often for Rose HA are SCSI and SCSI-2 hard disks. They provide good price/performance and price/storage capacity ratios. There are several configurations that can be implemented with Rose HA, which are as follows:

- Internal disk drives
- External disk drives
- Mirrored disk drives

- Redundant array of inexpensive disk (RAID) subsystems

Internal disk drives are used for storing the operating system, temporary spool areas, and applications and data that are not required to be accessible when a service failover process occurs. The same definition can be used for external (unshared) disk drives.

Mirrored disk drives allow for special redundancy and special processing on the active server. These devices provide the "first line of defense" in the event of a single failed disk device. When the HA agent detects a failure of the primary disk device, it can respond by accessing the respective mirror disk device before initiating a full system failover event.

RAID devices are usually external devices that are between two or more servers. When connected as multi-hosted devices, they can be simultaneously connected to both the active and standby servers. This provides the standby server with a direct physical data path to access the disk partition(s) or device(s), and the ability to launch the critical applications and/or access the critical data after a failover event has occurred. RAID devices are available in several different configurations, with the most popular being RAID-1, RAID-3, and RAID-5.

RAID-1 provides hardware disk mirroring. RAID-1 detects a failed disk device and processes the failure automatically without notifying the operating system. The HA agent can be configured to query the RAID controller, detect the failure and initiate a user-defined process accordingly.

RAID-3 and RAID-5 configurations allow for the failure of any one disk device without causing the failure of the entire disk subsystem. On these systems, the data is split

and written to multiple disk units, with a checksum entry associated with each disk write. When any one disk fails, the missing data can then be reconstructed from the checksum information. As with RAID-1, the RAID controller manages this automatically. Again, the HA agent can be configured to query the RAID controller, detect the failure and initiate a user-defined process accordingly.

Failover detection process

Two types of software entities are used on the Rose HA system: the HA manager and the HA agent.

The HA manager is the Rose HA kernel. It is the first daemon process initiated on each server. Each server that starts Rose HA is configured by HA manager as defined in the HA configuration file. The HA configuration file for each server is the same.

HA agents are daemon processes that monitor and manage the defined, critical services provided by the active server. The services that the agents monitor include communication, file, disk, network, NFS, NIS, DNS, and RDBMS. All the services and their corresponding agents are all defined in the HA configuration file. In addition, agents can be customized for special application services that are unique to the site.

These agents provide status signals for the critical services to the HA manager in the form electronic heartbeats. When the HA manager on the active server is receiving an "alive" or "healthy" heartbeat signal from all of its agent processes, it sends a heartbeat to the HA manager on the standby server.

This HA manager-to-HA manager heartbeat function informs the standby server that the active server is currently in good "health" and operating properly. When

this active-server-to-standby-server heartbeat is absent, the standby server assumes the active server has failed and initiates the defined failover processes.

If a critical service on the active server fails, the agent sends a “fail” heartbeat to the HA manager on the active server. The HA manager initializes the failover process as defined in the Rose HA configuration file.

If an agent on the active server fails, the HA manager on that server detects the absence of the agent’s heartbeats and, after a time-out (which is site-configurable), performs the designated tasks to failover to the standby server or pages the system administrator.

Any critical service on the active server can be monitored by more than one agent process. Each agent is designed to monitor a specific or unique aspect of that service. The service is considered to be available and “healthy” as long as the HA manager is receiving at least one heartbeat function from the individual agents that are monitoring that service.

Thus, if three agents are monitoring one service and one or two of the agents detect a failure, the HA manager does not initiate failover processing for that service until the third heartbeat function also signals a failure.

A service can be agentless, which means that there is no agent to watch over the availability of the service. The service is considered available as long as the service is properly started. A failover occurs only when the active server itself has failed.

When all the services managed by the HA manager are considered available, a server heartbeat is broadcast to the related standby server(s). The loss of a server heartbeat from an active server causes a failover of all of its services to the corresponding standby server(s).

Rose HA uses the standard UDP protocol to exchange information with the HA manager to agents, and HA manager-to-HA manager heartbeat functions. Use of the standard UDP protocol means that upgrades to new communication media can be done without any impact on the integrity of Rose HA.

Failover processing

When a failover event occurs, there is a brief interruption in services for the failure recognition and failover process to initialize the services on the standby server. This process occurs automatically without human interaction. Once the failover processing is completed, the services provided by the active server operates on the standby server.

A failover process can be initiated by either the active server or standby server, depending upon the type of heartbeat loss and the defined procedure to follow once that heartbeat function has failed.

A failover process involves transferring the following to the standby server:

- The network identity of the active server (which can be an IP address or X.25 address)
- The shared disk subsystem(s)
- The designated services provided by the active server to the standby server

The failover of stateless applications, such as NFS, NIS, and DNS, are transparent to the end user. For applications that have state, such as **ftp**, **rlogin**, and t**elnet**, the connection to the server application must be reestablished after the failover event.

Other processes, such as client/server database applications can be programmed to acquire the status of the application/service provided by the active server and then it resumes or reconnects to the service that is now provided by the standby server. This programming technique allows these stateful applications to appear as stateless applications to the end user.

Note: Terminals that are directly connected to serial ports on the active server are rendered unusable due to the nature of serial port interfaces. These interfaces cannot be failed over to another server. However, network terminal servers and client terminal processes such as X-Windows, **telnet**, and **rlogin** sessions are terminated, but the users can reestablish connections to the standby server and continue to access the applications / services that have been resumed on that server.

When the failed active server recovers, or has been repaired, the HA manager can be reconfigured to allow it to become the standby server. Otherwise, it can be configured to reclaim its original role as the active server and restore its network identity, resources, applications, and services from the standby server and return to active server status.

The following list contains some of the possible configurable responses to failed services:

1. The failure of the service is ignored

2. A hardware device has failed and an alternate is configured and available. Failover to an alternate device is initiated

3. A software service such as NFS or DNS has failed. Failover is initiated immediately to resume the failed service

4. The system providing the service is shut down

5. A software application has failed. A number of attempts to restart the service on the active server is triggered. If the service fails to restart, the user may choose to:

    - Ignore the failure of the service
    - Halt processing of the service
    - Failover the service to the designated standby server

Initialization process

Rose HA is initialized as follows:

1. HA manager daemon starts
2. Heartbeat signals are initialized
3. All necessary parameters are initialized
4. The configuration file is loaded

Rose HA Customization

Rose HA provides a very flexible and easy way to configure the system through the use of a configuration file (**config.ha**). It also supports shell scripts and agent application interface (API) definition that allow a site to

customize the Rose HA system to site-specific specifications.

The following list contains some of the events and configurations that can be defined for Rose HA:

- Failover scheme
- Hardware redundancy configuration
- Functionality of the standby server
- Specification of either active or standby role for each configured server
- Actions and/or scripts to execute for specific failure events
- Mechanisms for monitoring and managing new services

Server configuration

Configuration parameters are used to customize Rose HA for each site. Sample configuration files are provided in **/usr/HA/ha_file/config.nfs** and /**config.sybase**. The following list contains some of the events and mechanisms that can be set by the configuration parameters:

- The server(s) and their defined role(s)
- Network configuration information for the private network and public network
- The standby server(s) designated for specific active server(s)
- Critical services and their corresponding HA agents
- HA agent process names and their heartbeat failure time-out limits and failure processing/actions
- HA manager failure time-out limits and failure processing/actions
- The maximum number of restart attempts for a failed service before failover processing begins

- Prerequisite services that must be operating before failover processing is initiated

User-defined shell scripts

User-defined shell scripts can add functionality, reliable logging of events and notification processing in response to processing of a failed service. User defined shell scripts can perform many tasks, some of which are as follows:

- Starting and stopping various services
- Defining follow-up procedures for the failed service(s)
- Sending messages to the system console
- Writing log file information for troubleshooting purposes
- Writing a message to the system logger
- Notifying support personnel via a pager
- Notifying help desk personnel via e-mail or other system management software
- Broadcasting messages to all users

User-defined agents

Rose HA provides both the API and HA agent templates for user-defined agents that are specific for the site's requirements. You should have a working knowledge of C programming language, and the application or service that the new agent monitors to write user-defined agents. Only the component that interacts with the service needs to be programmed. See "Agent application interface (API) definition," for more information on user-defined agents.

System administration

System administration utilities include support for checking configurations, installing the Rose HA software onto a new server, and configuration and management of the Rose HA environment. All these functions can be

performed from a single node on the network or from a system console.
Rose HA provides a graphical user interface (GUI) for easy system administration and HA manager and agent monitoring from any character based or X-Windows terminal. See "System Administration" for more information.

From this GUI interface, the system administrator can issue commands to perform the following tasks:

- Starting and stopping the HA manager
- Starting and stopping HA agents
- Starting and stopping specific services
- Forcing a failover process to occur from either the active or standby server
- Monitoring and querying the status of servers, networks, services and agents
- Verifying HA server configuration(s)
- Installing Rose HA software on another server
- Configuring and managing the site's Rose HA environment

Supported Configurations

Rose HA supports any Sun kernel configuration (including 2.x and 4.1.x) and whatever memory is needed. It supports all Sun-compatible third party interface boards. See "Minimal system requirements" for more information.

The following subsections describe some of the standard configurations in more detail.

Hot standby - one active server

This configuration, which is shown in Figure 2, defines one server as a mission critical system and the standby server as the active server's immediate replacement. The only function of the standby server is to monitor the

heartbeat functions of the active server and wait for a failure event to process.
Both servers are connected to the private network, the public network, and a shared external disk subsystem. This configuration can achieve consistent response time after failover processing, but the resources of the standby server are not used to the maximum. This configuration must meet the following minimal requirements:

- One designated active server
- One designated standby server (which **must** have the identical internal configuration of the active server for memory, and so)
- One private network interface in each server
- One public network interface in each server
- Two (minimum) SCSI / SCSI-2 interfaces in each system
- One (minimum) external SCSI / SCSI-2 dual-hosted disk subsystem

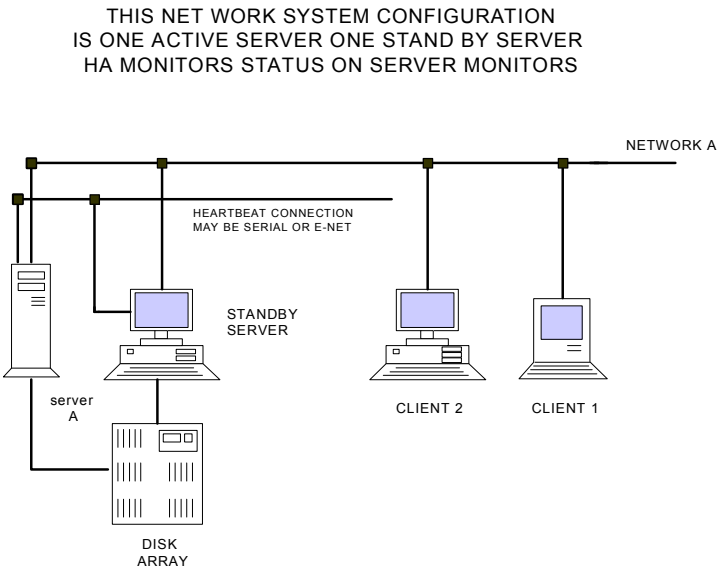

Figure 2. Hot standby - one active server configuration

Hot standby - two active servers

This configuration, which is shown in Figure 3, designates two servers as mission critical systems and the standby server as either of the active servers' immediate replacement. This configuration reduces the overhead of hardware expenses from 50% usage (as defined in the hot standby - one active server configuration) to approximately 33%.

To use this configuration, all servers must be connected to the same public network and a multi-hosted external disk subsystem. The standby server can be configured to achieve any desired level of performance.

For example, by planning against the probability of both active servers failing simultaneously, the standby server can be configured to resume all services from only one of active servers at a time. In the event the second server fails, the standby server can be configured to run in either a degraded state or failover only the most important, mission critical services. Conversely, the standby server can be fitted with the physical capacity to resume all services from both active servers at any time and operate within expected parameters.

This configuration must meet the following minimal requirements:

- Two designated active servers
- One designated standby server
- Two private network interfaces in each server
- Two (minimum) SCSI / SCSI-2 interfaces in each system
- One (minimum) external SCSI / SCSI-2 multi-hosted disk subsystem

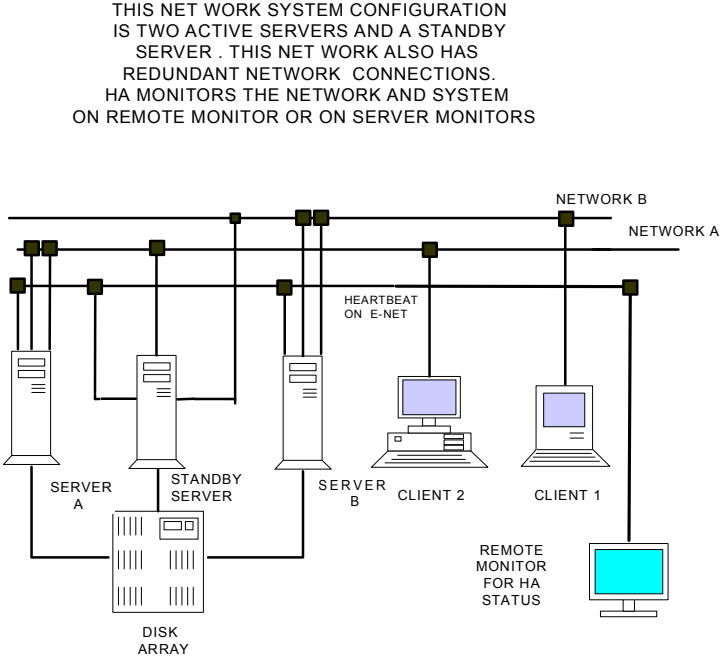

Figure 3. Hot standby - two active servers configuration

Warm standby - two active servers

In this configuration, which is shown in Figure 4, two mission critical systems are configured as a mutual standbys to one another. This configuration gains 100% usage of hardware expenses because the servers are both defined as mission critical.

In the event of a failed service, the other server resumes the failed service(s). If one of the servers fails entirely, the standby server can operate in a degraded state, depending upon how the server is physically configured for memory, CPU, and so.

To use this configuration, both servers must be connected to the same public network and an external shared disk subsystem. This configuration must meet the following minimal requirements:

- Two designated active servers
- Two public network interfaces in each server
- Two (minimum) SCSI / SCSI-2 interfaces in each system
- One (minimum) external SCSI / SCSI-2 dual-hosted disk subsystem

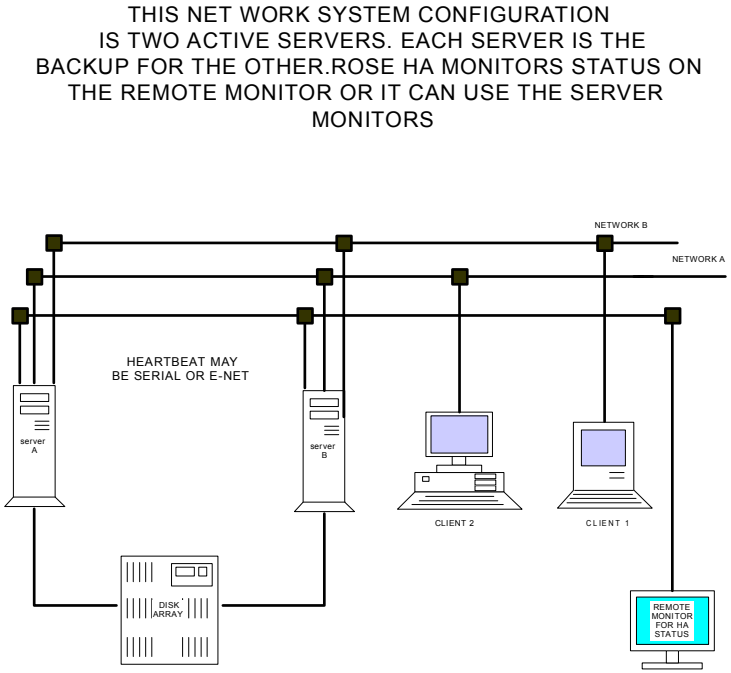

Figure 4. Warm standby - two active servers configuration

Minimal System Requirements

The following list defines the minimal system requirements for Rose HA:

Hardware

- Two or more SPARC-compliant computer systems, or two or more PCs running Solaris x86, or 2 or more Sun models running 4.x. Each machine must be running the same version of the operating system.

Memory

- 1 MB of RAM for Rose HA software

Disk capacity

- 2 MB of internal disk space for Rose HA software
- 1 MB of shared disk space for exchanging heartbeat messages

Minimum disk array requirements:

- Multi-ported or dual-ported SCSI or FDDI disk array

Hub requirements

- Need at least a four-port hub (final number depends on the size and number of client machines)

Network requirements

For two systems with serial heartbeat:

- One available serial port in each machine
- One null modem cable

For a non-redundant network:

- One Ethernet, one FDDI, or one token ring card or port in each machine

For a redundant network:

- Two Ethernet, two FDDI, or two token ring cards or ports in each machine

Two systems with a Ethernet private network:

- Ethernet port for each machine
- A cross-connect cable

Supported Applications

- UNIX services such as NFS, NIS, DNS, and so on
- Database (RDBMS) services, such as Sybase, Ingress, and so on
- User-defined client/server applications in fields such as banking, crucial network services, government services, and so on

To run the system administration GUI interface, you must have OpenWindows running on a graphics terminal.

Installation Procedure

1. If you are currently running OpenWindows, exit to the system prompt. In the following examples, the system prompt is shown as **system-#1**; it will be different for your site. Commands that you should type in are shown in bold.

2. Determine the host ID for each server, as shown in the following examples:

```
system-1# /usr/ucb/hostid
8034dd8b
```

```
system-2#  /usr/ucb/hostid
80338f06
```

The host IDs are unique for your servers. That is, the ID returned when you execute the **hostid** command will be specific to your system.

At this time, you should fill out the License Request Form (located at the end of this manual), including the host ID numbers that were determined in step 2. You will need this information from your distributor to finish configuring your system (which is explained later).

3. Stop the Volume Management Daemon (**/usr/sbin/vold**), as shown in the following example:

```
system-1#
/etc/init.d/volmgt stop
```

4. Install the Rose HA software by using the package add utility, as shown in the following example:

```
system-1#  pkgadd -d
/dev/diskette
```

5. The package add utility is interactive. Insert the first diskette and type **go**, as shown in the following example. Follow the instructions as they appear on your screen.

Note that the path to the package base directory must be **/usr/HA**. It is required that the application be installed in this directory to initialize, operate and shutdown properly. If you choose to change the installation path, you must make a symbolic link from **/usr/HA** to the install directory.

The following output is provided as an example:

```
Insert diskette 1 of 3 into Floppy Drive.
Type      [go] when ready,
          or   [e] to eject the diskette,
          or   [q] to quit:  go

The following packages are available:
          1    Harmony  Rose High Availability
                        (SPARC-SunOS-5.x)
                        [ROSE-HA 3.1]

Select package(s) you wish to process
(or 'all' to process all packages).  (default:  all)
[?, ?? q]:  1

Processing package instance <ROSE-HA>
from </dev/diskette>

Rose High Availability
(SPARC-SunOS-5.x) [ROSE-HA 3.1]
Copyright (c) 1994 by Rose Datasystems, Inc.

Enter path to package base directory
[?,q] /usr/HA

The selected base directory </usr/HA> must
exist before installation is attempted.

Do you want this directory created now
[y,n,?,q] y
```

Using </usr/HA> as the package base directory.

Processing package information
Processing system information
Verifying disk space requirements
Checking for conflicts with packages
already installed
Checking for setuid/setgid programs

This package contains scripts which will be executed with super-user permission during the process of installing this package.

Do you want to continue with the installation of this package [y,n,?,] **y**

Installing Rose High Availability as <ROSE-HA>

Installing part 1 of 3
/usr/HA/ha_api/haapi.x
/usr/HA/ha_api/makefile
/usr/HA/ha_exe/ha_kill
/usr/HA/ha_exe/ha_openwin
/usr/HA/ha_exe/haadm.exe
/usr/HA/ha_lib/libIV.so.3.1.Z
/usr/HA/ha_util/page_adm
/usr/HA/ha_util/util_dev_id

##Install part 2 of 3

Insert diskette 2 of 3 for <ROSE-HA> package into Floppy Drive.
Type [go] when ready,
or [e] to eject the diskette,
or [q] to quit: **e**

```
                [/usr/bin/eject]

Type      [go] when ready,
          or   [e] to eject the diskette,
          or   [q] to quit:  go
/usr/HA/ha_api/haapi.c
/usr/HA/ha_exe/HA_cmd.x
/usr/HA/ha_exe/HAadm
/usr/HA/ha_exe/ag_det.x
/usr/HA/ha_exe/ha_mai.x
/usr/HA/ha_exe/win_st.x
/usr/HA/ha_file/config.nfs
/usr/HA/ha_lib/libUnidraw.so.3.1.Z
## Installing part 3 of 3

Insert diskette 3 of 3 for <ROSE-HA>
package into Floppy Drive.
Type      [go] when ready,
          or   [e] to eject the diskette,
          or   [q] to quit:  e
               [/usr/bin/eject]

Type      [go] when ready,
          or   [e] to eject the diskette,
          or   [q] to quit:  go
/usr/HA/ha_api/libha_api.a
/usr/HA/ha_exe/HA_mon.x
/usr/HA/ha_exe/HA_xxx.x
/usr/HA/ha_exe/HArmony
/usr/HA/ha_exe/ag_bea.x
/usr/HA/ha_exe/ha_SK99
/usr/HA/ha_exe/ha_autoexec
/usr/HA/ha_exe/win_hi.x
/usr/HA/ha_exe/win_yy.x
/usr/HA/ha_file/config.blank
/usr/HA/ha_file/config.help
/usr/HA/ha_file/config.sybase
/usr/HA/ha_file/hosts.sample
```

```
/usr/HA/ha_file/sybase_start
/usr/HA/ha_file/sybase_stop
/usr/HA/ha_util/util_color
/usr/HA/ha_util/util_font
/usr/HA/ha_util/util_kill
/usr/HA/ha_util/util_wait

[verifying class <none>]
##Executing postinstall script.
add ROSE-HA to /etc/r3.d/S99.HArmony
add ROSE-HA-Menu to openwin Workspace

Installation of <ROSE-HA> was successful

Insert diskette 3 of 3 for <ROSE-HA>
package into Floppy Drive.
Type      [go] when ready,
          or   [e] to eject the diskette,
          or   [q] to quit:  e
               [/usr/bin/eject]

Type      [go] when ready,
          or   [e] to eject the diskette,
          or   [q] to quit:  q
```

6. Verify the package installation, as shown in the following example. Note that the **BASEDIR** address will be the directory where you installed the package and the **INSTDATE** value will be the date that you installed the package. The **FILES** values can be different from those shown in the following example:

```
system-1#  pkginfo -l ROSE-HA
   PKGINST:      ROSE-HA
      NAME:      Rose High Availability
CATEGORY:        system
      ARCH:      SPARC-SunOS-5.x
   VERSION:      [ROSE HA 3.1]
   BASEDIR:      /usr/HA
    PSTAMP:      pp0940517221111
 INSTDATE:       Apr 21 1995 05:45
    STATUS:      completely installed
     FILES:         41  installed pathnames
                     5  directories
                    36  executables
                  6674  blocks used (approx)
```

7. Repeat the steps 1 through 6 for each server on which you will configure the Rose HA software. Proceed to the configuration procedure, which is described in the next subsection.

Configuration Procedure

This subsection provides instructions on how to set the configuration parameters for the Rose HA software.

1. Change to /**usr/HA/ha_file**, as shown in the following example:

```
system-1#  cd /usr/HA/ha_file
```

2. To create the **config.ha** configuration file, copy the **config.blank** file to **config.ha**, as shown in the following example:

```
system-1#  cp -p config.blank config.ha
```

3. Determine the IP addresses of the server names which are involved in Rose HA, as shown in the following example:

```
system-1#  more /etc/hosts
192.9.200.2     system-2
192.9.100.2     system-1
```

4. You then need to determine the physical addresses (the **Phys Addr** field in the following example) of the IP addresses that you defined in step 3, as shown in the following example. Write this information on a piece

of paper, as you will need it when setting the configuration parameters:

```
system-1# arp -a
Net to Media Table
Device IP Addr       Mask          Flags   Phys Addr
le0    system-2      255.255.255           08:00:20:1f:21:07

le0    system-1      255.255.255   SP      08:00:20:20:6f:8c
```

5. To aid in editing the **config.ha** file, you should start OpenWindows (by typing in **openwin** at the system prompt). You want the OpenWindows interface so you can open both the **config.help** file and the **config.ha** file.

   In the first command or tool shell window, **cd** to **/usr/HA/ha_file** and open the **config.help** file with the text editor of your choice (this part is not shown in the following example). This results in the display of the **config.help** file, which is an online version of the information found in "Description of Configuration Parameters". Displaying this file allows you to view the descriptions of each of the configuration parameters as you are in the actual process of setting these parameters for your site. Use of this file eliminates the need to flip through the pages of the hard-copy Rose HA manual.

   If you do not use the **config.help** file, refer to the descriptions of the configuration parameters in "Description of Configuration Parameters" for more information on each parameter. Regardless of which version of the configuration information you use, it is very important that you read the descriptions carefully. Your Rose HA system will not execute

properly if the configuration parameters are not set correctly.

6. Set the values for your **config.ha** file. Do this by opening another command or shell tool window, change to **/usr/HA/ha_file** and open the **config.ha** file, using the text editor of your choice, as shown in the following example. This example shows the **vi** editor being used to open and edit the **config.ha** file.

   The following example has sample entries for server hostnames, **ss0** and **pp0**, and jobs **nfs1** and **nfs2** for an active-active server configuration (your **config.ha** file will not have values assigned to the parameters). If you commented out all the parameters from **JOB=nfs2** to **STANDBY_DISK=/dev/dsk/clt0d0s0** in the following example, you would have an active-standby server configuration. You comment out a parameter by putting a # in front of it.

   Note that the following example is the same file that is found in the **/usr/HA/ha_file/config.nfs** example file. An example of an active-active server configuration for Sybase jobs can be found in **/usr/HA/ha_file/config.sybase**. Both files are provided as examples only; you must set the parameters in your **config.ha** file as needed for your site's configuration.

   User-defined hostnames must be configured in **/etc/hosts** as well as the **config.ha** file. These hostnames must match as the active and standby system definitions. Hostnames cannot exceed 8 characters.

```
system-1#  cd /usr/HA/ha_file
system-1#  vi config.ha


##HA configuration file
#
#
#####################################
#           License Definition          #
#####################################
#
LICENSE=ss0:DBEE7F31AG17051B
LICENSE=pp0:085F2A394271FD73
SERIAL=ss0:1234567890123456
SERIAL=pp0:1234567890123456
DATE=ss0:05201994
DATE=pp0:05201994
#
#
#####################################
#           Node Definition          #
#####################################
#
HOST_NODE=ss0,pp0
#
ORGINAL_IP=ss0:le0:ss0,le1:ss1
ORGINAL_IP=pp0:le0:pp0,le1:pp1
ORGINAL_ETHER=ss0:8:0:20:12:e8:9
ORGINAL_ETHER=pp0:8:0:20:a:11:19
ORGINAL_NETMASK=ss0:le0:255.255.255.0,le1:
255.255.255.0
ORGINAL_NETMASK=pp0:le0:255.255.255.0,le1:
255.255.255.0
#
#
#
#
#####################################
```

```
#            Job Definition            #
########################################
#
JOB_NAME=nfs1,nfs2
#
#
JOB=nfs1
#
ACT_NODE=ss0
ACT_LAN=le1
ACT_IP=act-ss0
ACT_NETMASK=255.255.255.0
#
SHARE_DISK=/dev/dsk/c1t0d0s0
MOUNT_POINTER=/share_nfs1
SUPPORT=NFS
SWITCH_BACK=yes
#MOUNT_OPTIONS=
#SHARE_OPTIONS=
#
STANDBY_NODE=pp0
STANDBY_LAN=le0
STANDBY_DISK=/dev/dsk/c1t0d0s0
#
#
JOB=nfs2
#
ACT_NODE=pp0
ACT_LAN=le1
ACT_IP=act-pp0
ACT_NETMASK=255.255.255.0
#
SHARE_DISK=/dev/dsk/c1t0d0s0
MOUNT_POINTER=/share_nfs2
SUPPORT=NFS
SWITCH_BACK=yes
#MOUNT_OPTIONS=
#SHARE_OPTIONS=
```

```
#
STANDBY_NODE=ss0
STANDBY_LAN=le0
STANDBY_DISK=/dev/dsk/c1t0d0s0
#
#######################################
#           Misc. Definition          #
#######################################
#
HEART_BEAT=ss0:/dev/term/a
HEART_BEAT=pp0:/dev/term/a
ASYNC_RATE=9600
#
ALIVE_CHECK_TIME=8
DEVICE_CHECK_TIME=10
#
##end
```

7. Create any job start and stop scripts that are required in **/usr/HA/ha_file.** These scripts execute the normal startup/shutdown commands or scripts related to the specific jobs. Any job specified in **config.ha** that is not running on the standby server must have a startup and stop script specified for it. Normal operating system processes (for example, the NFS daemon) do not need these scripts; RDBMS applications do need the scripts. See the **sybase_start** and **sybase_stop** example files in **/usr/HA/ha_file.**

8. Add the **/usr/HA** install directory to the **PATH** statement in the **.profile** file.

9. To streamline the configuration process, all parameters that control the managers and agents are entered into one **config.ha** file. Once the configuration information has been placed in this file, you must copy it to the other Rose HA servers via an appropriate mechanism (for example, **ftp**).

10. Reboot the server. When the system has restarted, you should see several status messages and the following message echoed to the console display during the boot process: **HArmony services started.**

    You should pay attention to the status messages as they appear because debugging and troubleshooting information may be contained in them. Rose HA generates the **log.[hostname]** file in **/usr/HA/ha_file** for each hostname specified in **config.ha**. The log file contains the messages generated during Rose HA boot time and other ongoing Rose HA activity (for example, at fail mode). Refer to this file if you are having problems at reboot time after defining **config.ha** or during other operational problems.

11. OpenWindows comes up automatically. Move into the Workspace menu and click on the ROSE-HA menu to access the Rose HA GUI system administration interface. See "System Administration" for more information on this interface.

Description of Configuration Parameters

The following paragraphs describe the Rose HA configuration file parameters and the proper syntax for each entry.

In the following descriptions, required configuration line entries are denoted by the use of < >, as shown in the following example:

```
SERIAL=<hostname1>:<serial number1>
SERIAL=<hostname2>:<serial number2>
```

In the following descriptions, optional configuration line entries are denoted by the use of [ ], as shown in the following example:

```
[SERIAL=<hostname3>:<serial number3>]
...
[SERIAL=<hostnameN>:<serial numberN>]
```

In the following descriptions, required replaceable parameters are denoted with < >, as shown in the following example:

```
<hostname> or <password>
```

In the following descriptions, optional parameters are denoted with [ ], as shown in the following example:

```
[,hostname2,hostname3,hostnameN]
or
[,job2,job3,jobN]
```

In the following descriptions, single required parameters contained in a list are denoted with a |, as shown in the following example:

```
{yes|no} or {9600|19200|38200}
```

Do not leave white space (for example, tab or spacebar) between entries on a single line. The delimiter is the comma (,) or a colon (:), as shown in the example configuration file in the Configuration Procedure section.

The configuration parameters are grouped in the **config.ha** file according to the following functions:

- License definition parameters: LICENSE, SERIAL, and DATE
- Node definition parameters: HOST_NODE, ORGINAL_IP, ORGINAL_ETHER, and ORGINAL_NETMASK
- Job definition parameters: JOB_NAME, JOB, ACT_NODE, ACT_LAN, ACT_IP, ACT_NETMASK, SHARE_DISK, MOUNT_POINTER, SUPPORT, SWITCH_BACK, MOUNT_OPTIONS, SHARE_OPTIONS, STANDBY_NODE, STANDBY_LAN, and STANDBY_DISK
- Miscellaneous definition parameters: HEART_BEAT, ASYNC_RATE, ALIVE_CHECK_TIME, and DEVICE_CHECK_TIME

The following subsections describe each configuration parameter in the order it appears in the **config.ha** file.

LICENSE configuration parameter

Syntax: LICENSE=<hostname1>:<license number1>
LICENSE=<hostname2>:<license number 2>
[LICENSE=<hostname3>,<license number 3>]
...
...
[LICENSE=<hostnameN>,<license number N>]

The information for the LICENSE parameter is provided by the software vendor from whom you purchased the Rose HA software. You obtain this information when you fill out and send the License Request Form (found at the end of this manual). You must have as many LICENSE entries as there are Rose HA active and standby servers.

There are two types of licenses: evaluation and permanent. The evaluation license is assigned for sites testing Rose HA and is only good until the expiration date assigned to the evaluation copy of the software. The permanent license number is assigned to you when you have purchased the Rose HA software. When you obtain the permanent license number you must replace the expiration license number with the permanent one in the **config.ha** file.

SERIAL configuration parameter

Syntax: SERIAL=<hostname1>:<serial number1>
SERIAL=<hostname2>:<serial number2>
[SERIAL=<hostname3>:<serial number3>]
...
...
[SERIAL=<hostnameN>:<serial numberN>]

Each SERIAL entry is unique for each copy of Rose HA. You can find the serial number for your copy of Rose HA on the printed label on the installation diskette. You must have as many SERIAL entries as there are Rose HA active

and standby servers, although the same number is used for each entry.

DATE configuration parameter

Syntax: DATE=<hostname>:<mmddyy>

The information for the DATE parameter is provided by the software vendor from whom you purchased the Rose HA software. You obtain the information for this parameter when you fill out and send the License Request Form at the end of this manual. You must have as many DATE entries as there are Rose HA active and standby servers.

HOST_NODE configuration parameter

Syntax HOST_NODE=<hostname1>,<hostname2>,
<hostnameN>

The HOST_NODE parameter specifies all the nodes that are either active or standby Rose HA servers that are capable of monitoring the agent and manager heartbeat pulses. There is no limit to the number of nodes that can be specified. However, the hostnames that are specified as Rose HA active and standby servers must have license information. Systems that are defined for monitoring status only do not need license entries. You can obtain the hostnode information by executing the **uname -n** command.

ORGINAL_IP configuration parameter

Syntax: ORGINAL_IP=<hostname1>:<network interface1>:<server name1>,<network interface2>:<server name2>
ORGINAL_IP=<hostname2>:<network interface1>:<server name3>,<network interface2>:<server name4>

The ORGINAL_IP parameter specifies the native address of the host. It is the original address of the host before Rose HA changes its address to the active address. **<server nameN>** is the name that must be defined in the **/etc/hosts** file. You cannot use the IP addressing (for example, 192.168.1.1) to replace the **<server nameN>** value. Each network interface must have a unique server name. Some server names can be the same as the hostname. The following is an example of ORGINAL_IP entries:

ORGINAL_IP=ss0:le0:ss0,le1:ss1
ORGINAL_IP=pp0:le0:pp0,le1:pp1

The **/etc/hosts** file would have an IP address for six server names (hostnames), as follows:

IP1 (e.g., 192.9.200.2)	ss0
IP2	pp0
IP3	ss1
IP4	pp1
IP5	act_ss0
IP6	act_pp0

For this example, if you used the **uname -n** command on the **ss0** system, it would show that the hostname is the same as the server name. NOTE: Hostname cannot exceed 8 characters.

ORGINAL_ETHER configuration parameter

Syntax: ORGINAL_ETHER=<hostname1>:<enet hardware address1>
ORGINAL_ETHER=<hostname2>:<enet hardware address2>
[ORGINAL_ETHER=<hostname3>:<enet hardware address3>]
...
...
[ORGINAL_ETHER=<hostnameN>,<enet hardware addressN>]

The ORGINAL_ETHER parameter specifies the physical Ethernet number for each Rose HA server and monitor. To record the physical Ethernet hardware address, execute the **config -a** command; this command reports information for all network interfaces on each server. The information is displayed in hexadecimal form. Leading zeros are not displayed. This information was also obtained in step 4 of the configuration procedure. You must have as many ORGINAL_ETHER entries as there are Rose HA active and standby servers. Each entry should be unique.

ORGINAL_NETMASK configuration parameter

Syntax: ORGINAL_NETMASK=<hostname1>:<network interface1>:<netmask>,<network interface2>:<netmask>
ORGINAL_NETMASK=<hostname2>:<network interface1>:<netmask>,<network interface2>:<netmask>
[ORGINAL_NETMASK=<hostnameN>:<network interface1>:<netmask>,<network interface2>:<netmask>]

The ORGINAL_NETMASK parameter specifies the physical Ethernet mask for each Rose HA server and monitor. To record the netmask, execute the **config -a**

command; this command reports information for all network interfaces on each server. The information is displayed in hexadecimal form. You must have as many ORGINAL_NETMASK entries as there are Rose HA active and standby servers. Each entry should be unique.

JOB_NAME configuration parameter

Syntax: JOB_NAME=<jobname1>[,<jobname2>, <jobnameN>]

The JOB_NAME parameter describes all the jobs (that is, processes such as, NFS, Sybase, and so on) that operate under Rose HA monitoring agents. Job names are entered on a single line separated by commas. Do not put tab or space characters between job names. This entry is used to reference each job description.

This entry references a job's **[jobname]_start** and **[jobname]_stop** scripts. Each job that is defined in the JOB_NAME entry that is not running on the standby server must have a start up and stop script defined for it. The start up file must be called **job_name_start**, where **job_name** is the name of the job (for example, **sybase**) to be started. The stop file must be **job_name_stop**, where **job_name** is the name of job to be stopped.

There are sample start and stop scripts in **/usr/HA/ha_file/sybase_start** and **/usr/HA/ha_file/sybase_stop**, respectively. Normal operating system processes (for example, the NFS daemon) do not need these scripts; RDBMS applications need them.

JOB configuration parameter

Syntax: JOB=<jobname>

The JOB parameter specifies the start of a block that contains information related to management of each specific job. It can contain all or some of the following parameters, depending upon the required actions desired for failover:

ACT_NODE, ACT_LAN, ACT_IP, ACT_NETMASK, SHARE_DISK, MOUNT_POINTER, SUPPORT, SWITCH_BACK, MOUNT_OPTIONS, SHARE_OPTIONS, STANDBY_NODE, and STANDBY_LAN

You can have as many JOB entry blocks in the **config.ha** file as required. Rose HA manager determines the end of the JOB entry when it scans the next JOB=<jobname> line or any configuration parameter that is not listed above.

ACT_NODE configuration parameter

Syntax: ACT_NODE=<servername>

The ACT_NODE parameter defines the active server that executes the job under normal operating conditions.

ACT_LAN configuration parameter

Syntax: ACT_LAN=<network interface>

The ACT_LAN parameter defines the network interface that is used by the job under normal operating conditions. Examples of interface entries are **le0** or **le1**.

ACT_IP configuration parameter

Syntax: ACT_IP=<network interface>

The ACT_IP parameter allows the Rose HA software to dynamically change the identity of each server. This provides the capability for the software to operate in two modes: active and maintenance. It also enables a failed server to boot under its original identity after a repair, without interfering with network addressing by introducing a duplicate IP address. Refer to the **hosts.sample** file in **/usr/HA/ha_file**.

ACT_NETMASK configuration parameter

Syntax: ACT_NETMASK=<network interface>

The ACT_NETMASK parameter is the site-specific netmask used to determine subnet schemes. It is denoted in the dot notation form; a typical standard class-C subnet mask is 255.255.255.0. Some sites use non-standard netmasks, such as 255.255.255.192, to allow for more subnets in the site's configuration. To determine the netmask, execute the **config -a** command on each server. It will report the information for all network interfaces configured in the server.

SHARE_DISK configuration parameter

Syntax: SHARE_DISK=</dev/[r]dsk/cWtXdYsZ>
[,</dev/[r]dsk/cWtXdYsZ>]

The SHARE_DISK parameter informs the Rose HA server what partition(s) to mount for the current JOB entry. Multiple partitions can be mounted for a single job. The maximum number of partitions that you can specify is two.

MOUNT_POINTER configuration parameter

Syntax: MOUNT_POINTER=<absolute path1>[,<absolute path2>,<absolute pathN>]

The MOUNT_POINTER parameter coincides with the first disk device, the second disk device and so on. Each SHARE_DISK partition should mount to a logical point (which is a path to a specific directory).

SUPPORT configuration parameter

Syntax: SUPPORT=<NFS|[space]>

The SUPPORT parameter informs Rose HA that the job is NFS; do not specify this parameter for any job except NFS. If you specify this parameter, Rose HA exports automatically the file system that is defined by the MOUNT_POINTER parameter.

SWITCH_BACK configuration parameter

Syntax: SWITCHBACK={yes|no}

The SWITCH_BACK parameter instructs Rose HA whether to allow automatic switch back to the active server from the standby server once the failed service is restored. If **yes** is set, once the failed active server is repaired and rebooted with the Rose HA software, all jobs that were switched over to the standby server automatically transfer back to the active server. If **no** is set, the jobs must be manually switched back to the active server.

MOUNT_OPTIONS configuration parameter

Syntax: MOUNT_OPTIONS=<option1>[,<option2>, <optionN>]

The MOUNT_OPTIONS entry describes the system mount options for each disk. All standard Solaris mount options are supported.

For example, to specify a mount option for a read-only filesystem, the entry would be as follows:

MOUNT_OPTIONS=-o ro (read only)

SHARE_OPTIONS configuration parameter

Syntax: SHARE_OPTIONS=<option1>[,<option2>, <optionN>]

The SHARE_OPTIONS entry describes the partition mount options for each disk. All standard Solaris partition mount options are supported.

For example, to specify a partition mount option for a read-only filesystem, the entry would be as follows:

SHARE_OPTIONS=-o ro (read only)

STANDBY_NODE configuration parameter

Syntax: STANDBY_NODE=<hostname1> [,<hostname2>,<hostnameN>]

The STANDBY_NODE parameter informs Rose HA which server(s) is accessed in the event of a failed active server. The first hostname in the list is the initial machine to which the defined process(es) and/or service(s) fail over.

The ACT_NODE identity of the failed machine is then passed to the machine that takes over the failed process(es) and/or service(s). Multiple hostnames can be assigned to the STANDBY_NODE if there are more than two Rose HA servers present on the network. In a multiple-named STANDBY_NODE configuration, if the first standby server fails, the next live server is used.

STANDBY_LAN configuration parameter

Syntax: STANDBY_LAN=<lan interface1>[,<lan interface2>,<lan interfaceN>]

The STANDBY_LAN parameter defines which failover network port is assigned to the existing clients and also the network port of the failed machine for a local (internal) failover. This allows for one of the active server's two network interfaces to fail and the Rose HA manager running on that server to be configured to redirect data to the secondary network interface on that machine without initiating a complete failover. This is called a local failover.

STANDBY_DISK configuration parameter

Syntax: STANDBY_DISK=</dev/dsk/cWtXdYsZ>, </dev/dsk/cWtXdYsZ>

The STANDBY_DISK parameter defines the mount disk for the standby server, which can be different than the what is defined by SHARE_DISK. The partition name for the standby server is often different than the one for the active server.

HEART_BEAT configuration parameter

Syntax: HEART_BEAT=server1:{/dev/term/<terminal device>|lan}

The HEART_BEAT parameter defines which port is to be monitored for the heartbeat information for the Rose HA software. Currently, supported ports are Ethernet (IEEE 802.3) for the LAN port and an RS-232 compatible serial port (/dev/term<terminal device>).

If **lan** is specified as the monitored port, the Rose HA manager process automatically chooses the physical port (for example, **le0**, **le1**, **le2,** depending upon the number of Ethernet interfaces that is installed in the server). Use of this parameter is required if there are more than two Rose HA servers.

The serial port selection is efficient and inexpensive if there are only two Rose HA servers configured within a close proximity to each other.

ASYNC_RATE configuration parameter

Syntax: ASYNC_RATE={9600|19200|38200}

The ASYNC_RATE parameter specifies the baud rate of the serial port if it is the selected option for the heartbeat monitoring function.

ALIVE_CHECK_TIME configuration parameter

Syntax: ALIVE_CHECK_TIME=<seconds>

The ALIVE_CHECK_TIME parameter informs the Rose HA software the interval (in seconds) to wait after missing a heartbeat signal before initiating a switchover action. The HEARTBEATTIME is calculated as follows: HEARTBEATTIME = ALIVE_CHECK_TIME / 2.

NOTE: The minimum value that can be entered is 5. If the parameter is commented out, Rose HA uses a default value of 8 seconds.

DEVICE_CHECK_TIME configuration parameter

Syntax: DEVICE_CHECK_TIME=<seconds>

The DEVICE_CHECK_TIME parameter specifies the interval (in seconds) between broadcasting the complete Rose HA server's configuration status to the other Rose HA server(s). Because the normal HEART_BEAT function broadcasts only a small amount of information, the DEVICE_CHECK_TIME parameter allows the Rose HA servers to inform other servers on the network of the current configuration parameters.

NOTE: A small value can use system resources than is desirable. Test and monitor this entry carefully. If this parameter is commented out, Rose HA uses a default value of 10 seconds.

System Administration

After the Rose HA system boots and OpenWindows comes up, the GUI system administration interface appears on the screen, as shown in Figure 5. The hostnames for the servers are shown in the display, along with the **Admin** and **Exit** selections. A dot in the screen indicates that particular system is running; if the screen has no dot, it means the system has stopped. When the dots are blinking, it indicates that the applications are running.

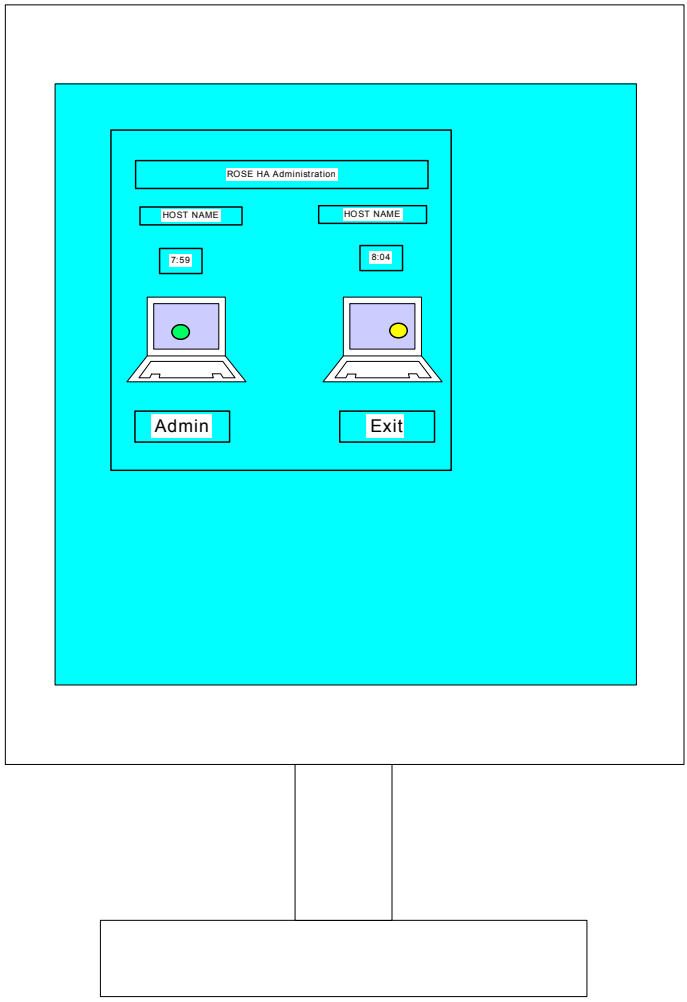

Figure 5. System administration GUI interface

If you click on **Admin,** the Admin Control screen appears, as shown in Figure 6. The Admin Control screen contains the following options:

- Start Rose HA
- Stop Rose HA
- Start Monitor

- Stop Monitor
- Switch Service
- Takeover Service
- Download Software
- Trace On
- Trace Off

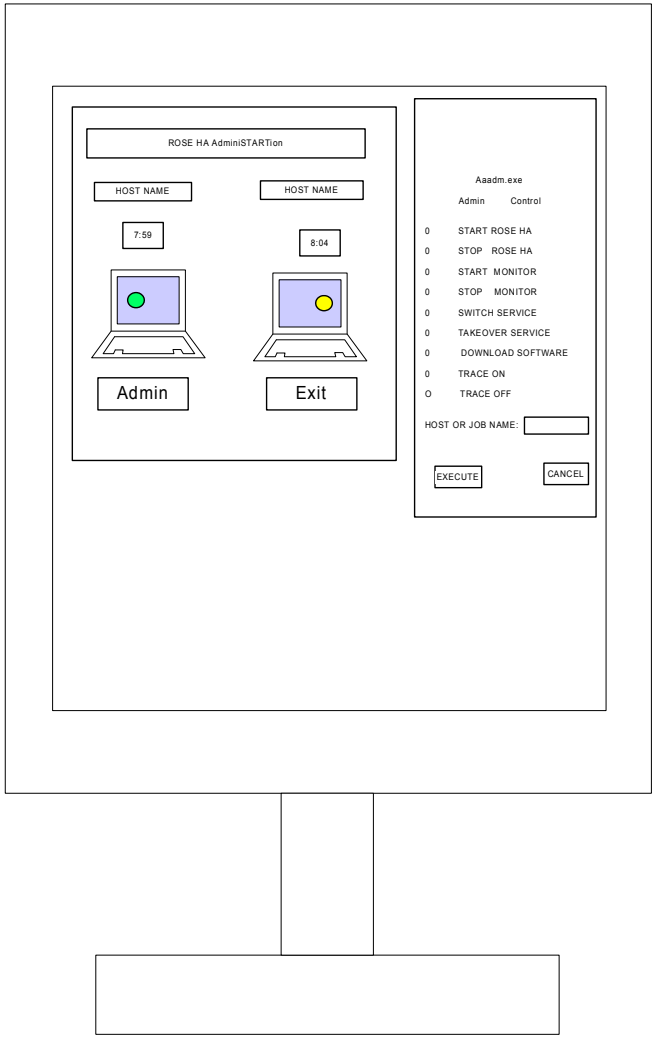

Figure 6. Rose HA administration interface

Start Rose HA

The **Start Rose HA** selection starts the Rose HA daemon. To use this selection, do the following:

1. Click on dot next to **Start Rose HA** selection
2. Type in hostname in the **Host or Job Name** box
3. Click on the **Execute** selection

Stop Rose HA

The **Stop Rose HA** selection stops the Rose HA daemon. To use this selection do the following:

1. Click on dot next to **Stop Rose HA** selection
2. Type in hostname in the **Host or Job Name** box
3. Click on the **Execute** selection

Start Monitor

The **Start Monitor** selection, shown in Figure 7, starts the status and history window. To use this selection do the following:

1. Click on dot next to **Start Monitor** selection
2. Type in hostname in the **Host or Job Name** box
3. Click on the **Execute** selection

Stop Monitor

The **Stop Monitor** selection stops the status and history window. To use this selection do the following:

1. Click on dot next to **Stop Monitor** selection
2. Type in hostname in the **Host or Job Name** box
3. Click on the **Execute** selection

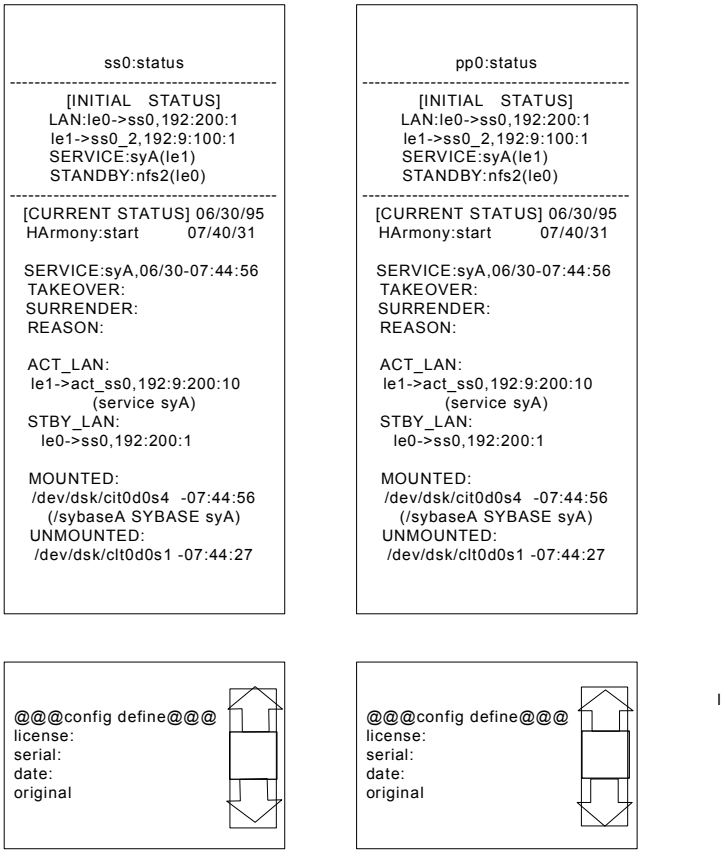

Figure 7. Start Monitor display

Switch Service

The **Switch Service** selection moves the service job of the host on which you are executing to the other host. To use this selection do the following:

1. Click on dot next to **Switch Service** selection
2. Type in job name in the **Host or Job Name** box
3. Click on the **Execute** selection

For example, assume that a job called **SyA** is running on host **pp0** and a job called **nfs** is on host **ss0**. If you click

on the dot next to the **Switch Service** selection and type **nfs** into the **Host or Job Name** box, then host **pp0** will have both jobs **SyA** and **nfs** running on it. Host **ss0** will have no jobs running on it.

Takeover Service

The **Takeover Service** selection takes the service job of the other host to the host on which you are executing. To use this selection do the following:

1. Click on dot next to **Takeover Service** selection
2. Type in job name in the **Host or Job Name** box
3. Click on the **Execute** selection

Download Software

The **Download Software** selection downloads all Rose HA files from the host on which you are executing to the host that you specify (that is, duplicates the installation). To use this selection do the following:

1. Click on dot next to **Download Software** selection
2. Type in hostname in the **Host or Job Name** box
3. Click on the **Execute** button

Trace On

The **Trace On** selection routes all messages related to Rose HA to the OpenWindows console. To use this selection, do the following:

1. Click on dot next to **Trace On** selection
2. Click on the **Execute** selection

Trace Off

The **Trace Off** selection turns off the routing of messages related to Rose HA to the OpenWindows console. To use this selection, do the following:

1. Click on dot next to **Trace Off** selection

2. Click on the **Execute** selection

Agent Application Interface (API) Definition

Rose HA provides function calls in the C language format. These function calls allow different applications to send commands to Rose HA, send a status to Rose HA, switch the service job, and take over the service job. Examples of these function calls are available in the **/usr/HA/ha_api/haapi.c** file, which is installed during the Rose HA installation procedure.

One set of function calls results in the same tasks as the selections in the system administration GUI interface. For example, the following function call is the same as the **Start Rose HA** selection in the GUI interface:

```
HA_command("Start Rose HA","ss0")
```

The second set of function calls are described in the following paragraphs.

The **HA_api_open(char*jobname,char*timeout)** function call makes the application agent feed the **HA_api_ok(char*jobname)** function call to Rose HA once every number of seconds defined by **timeout**. Otherwise, Rose HA switches the job to the other node. An example of this function call is as follows:

```
HA_api_open("nfs1","12")
```

In this example, the application agent needs to feed the **HA_api_ok("nfs1")** to Rose HA once every 12 seconds.

Otherwise, Rose HA switches the job (**nfs1**) to the other node.

Once you open a job and fill in the **timeout** parameter in the **HA_api_open** function call, Rose HA knows the job exists. It then monitors the job by receiving the **HA_api_ok(char*jobname)** function call within the limit defined by **timeout**. The use of this function call is shown in the previous example.

The **HA_api_error(char*jobname)** function call allows the agent to inform Rose HA to switch the service job immediately.

The **HA_api_close(char*jobname)** function call terminates the agent monitoring the job that was opened by the **HA_api_open** function.

The following is a display of the **haapi.c** file:

```
/*
HA_command(char*command,char*host_node_job_
name)

      HA_command(“start RoseHA”,“ss0”);
      HA_command(“stop RoseHA”,“ss0,pp0”);
      HA_command(“start monitor”,“ss0,pp0,ii0”);
      HA_command(“stop monitor”,“pp0,ii0”);

      HA_command(“switch service”,“nfs1,nfs2”);
      HA_command(“takeover service”,“nfs1”);

      HA_command(“download software”,“pp0,ii0”);

HA_api_open(char*jobname,char*timeout)
      HA_api_open(“nfs1”,“12”)

HA_api_ok(char*jobname)

HA_api_error(char*jobname)

HA_api_close(char*jobname)
```

System administrator paging script

Rose HA provides a script file that pages an administrator. This script allows one or more system administrators to be notified of a failure condition via a pager or email. To customize this file for your site, edit the **page_adm** file in **/usr/HA/ha_util**. Add the names of the system administrators and associated phone numbers to this file.

Error logging

Rose HA generates the **log.[hostname]** file in **/usr/HA/ha_file** for each hostname specified in **config.ha**. The log file contains the messages generated during Rose HA boot time and other ongoing Rose HA activity (for example, at fail mode). Refer to this file if you are having problems at reboot time after defining **config.ha** or during other operational problems.

Rose HA software files

The following lists contain the software files associated with the Rose HA software. The files are grouped by function.

/usr/HA/ha_api directory

The **/usr/HA/ha_api** directory contains all the Rose HA application interface (API) files, which are as follows:

File	Description
ha_api/haapi.c	Rose HA agent example program
ha_api /haapi.x	Rose HA agent example execute
ha_api/libha_api.a	Rose HA API library
ha_api/makefile	Rose HA example make file

/usr/HA/ha_exe directory

The **/usr/HA/ha_exe** directory contains all the Rose HA execute files, which are as follows:

File
ha_exe/HA_cmd.x
ha_exe/HA_mon.x
ha_exe/HA_xxx.x
ha_exe/HAadm
ha_exe/HArmony
ha_exe/ag_bea.x
ha_exe/ag_det.x
ha_exe/ha_SK99
ha_exe/ha_autoexec
ha_exe/ha_kill
ha_exe/ha_mai.x
ha_exe/ha_openwin
ha_exe/haadm.exe
ha_exe/win_hi.x
ha_exe/win_st.x
ha_exe/win_yy.x

/usr/HA/ha_file directory

The **/usr/HA/ha_file** directory contains examples of configuration files, examples of start and stop script files, and the log file, which are as follows:

File	Description
ha_file/config.ha	Rose HA configuration file. You copy the **config.blank** file to **config.ha** and then modify it for your site.
ha_file/config.blank	Blank configuration file
ha_file/config.help	Contains description of configuration parameters; use during configuration process for help with configuring site's **config.ha** file.
ha_file/config.nfs	Example of an active-active server configuration file for a NFS job
ha_file/config.sybase	Example of an active-active server configuration file for a Sybase job
ha_file/hosts.sample	Example of a host definition file
ha_file/log.[hostname]	Rose HA log file
ha_file/[jobname]_start	Job start script that you must specify before you start Rose HA
ha_file/[jobname]_stop	Job stop script that you must specify before you start Rose HA
ha_file/sybase_start	Example of a start script file for a Sybase job
ha_file/sybase_stop	Example of a stop script file for a Sybase job

/usr/HA/ha_lib directory

The /usr/HA/ha_lib directory contains all the Rose HA libraries, which are as follows:

File
ha_lib/libIV.so -> **/usr/HA/ha_lib/libIV.so.3.1**
ha_lib/libIV.so.3.1
ha_lib/libUnidraw.so ->
/usr/HA/ha_lib/lib/Unidraw.so.3.1
ha_lib/libUnidraw.so.3.1

/usr/HA/ha_util directory

The **/usr/HA/ha_util** directory contains all the Rose HA utilities, which are as follows:

File	Description
ha_util/util_color	Can be used to look up the OpenWindows color and its ID
ha_util/util_dev_id	Used to identify the major disk ID and minor ID. If you are running a NFS job, use this file to ensure the disk has the same ID on each server
ha_util/util_font	Used to look up the OpenWindows font and its ID
ha_util/util_kill	Used to kill a process
ha_util/util_wait	
ha_util/page_adm	Used to page an administrator if a failure condition occurs.

License Request Form

The following form must be filled out and faxed to the distributor from whom you purchased the Rose HA software. The information that is returned is needed to configure your system. You must fill out the first (unshaded) box before faxing; the vendor will fill out the shaded boxes and return the information to you either via email or FAX number as you specified. Your license information will be returned to you within one business day. Please type or print clearly.

Server Model:
1.
2.
3.
4.
Server Hostnames.
1.
2.
3.
4.
Server HostIDs:
1.
2.
3.
4.
Your name:
Your company name:
Your company address:
Your city/state/zip code
Send via email:
or send via fax number:
(indicate return method by providing email address
or fax number)

Vendor Use Only
Date/time received:
Date/time sent:

HostIDs:	**Expiration Date:**	**License Number:**