Primer on Clusters for High Availability

by Peter S. Weygant
Hewlett-Packard Company
Availability and Consolidation Solutions Laboratory
19111 Pruneridge Avenue—Mail Stop 44UA
Cupertino, CA 95014
peter_weygant@hp.com

In the last decade, high availability has become a top requirement for commercial UNIX systems, including the HP 9000. This paper describes some basic methods for achieving high availability through clustering, which allows users to eliminate single points of failure in their HP-UX systems, thereby reducing unplanned down time. Then it describes several forms of cluster architecture, including disaster tolerant clusters. Finally, it illustrates some tools for use in monitoring and managing HP-UX clusters.

1.0 What is Availability?

Availability is the percentage of **elapsed time** during which a system can be used. Elapsed time is continuous time, including both operating time and all inoperative times. One common formula for availability is as follows:

$$\%\ \text{Availability} = \frac{\text{Elapsed Time - Sum (Inoperative Times)}}{\text{Elapsed Time}} * 100$$

Availability is actually the probability that the system is operating normally. Availability is usually expressed as a percentage of hours per week, month, or year during which the system can be used for normal business.

High availability is simply a high probability that a component is up and running. The meaning of “high” varies, of course, with business needs. Measures of availability must be seen against the background of the organization’s expected period of operation of the system. For example, Table 1 shows data for several availability levels on a 24 x 7 system, that is, a system expected to be in use 24 hours a day, seven days a week, 365 days a year.

TABLE 1. Uptime and Downtime for a 24 X 7 System

Availability	Minimum Expected Uptime	Maximum Allowable Downtime	Remaining Time
99%	8672	88	0
99.5%	8716	44	0
99.95%	8755	5	0
99.999%	8760	5 minutes	0
100%	8760	0	0

Note that in this table, all the available time in the year (8760 hours) is accounted for. This means that all maintenance must be carried out either when the system is up or during the allowable downtime hours. In addition, the higher the percentage of availability, the less time is allowable for failure.

2.0 Obstacles to High Availability

It is important to understand the obstacles to high availability computing. This section describes some characteristics of these obstacles.

A specific loss of a computer service as perceived by the user is called an **outage**. The duration of an outage is **downtime**. Downtime is either planned or unplanned. Planned downtime is sometimes allocated for system upgrades, movement of an application from one system to another, physical moves of equipment, and other reasons.

Unplanned downtime occurs when there is a failure somewhere in the system. A failure is a cessation of normal operation of some component. Failures occur in hardware, software, system and network management, and in the environment. Errors of human judgment also cause failures. Not all failures cause outages, of course; and not all unplanned outages are caused by failures. Natural disasters and other catastrophic events can also disrupt service.

2.1 Duration of Outages

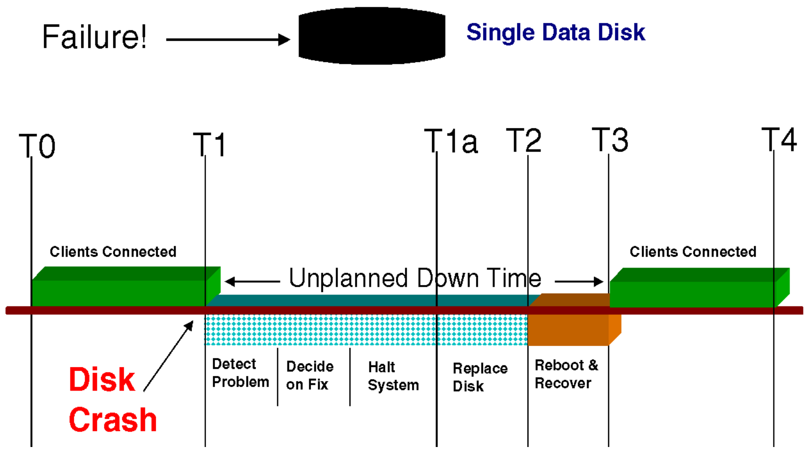
An important aspect of an outage is its duration. Depending on the application, the duration of an outage may be significant or insignificant. A 10-second outage may not be critical, but two hours may be fatal to one application, while another application may not even tolerate a 10-second outage. Thus, your characterization of availability must encompass the acceptable duration of outages.

As an example, if your goal is 99.5% availability on a 24x7 system, you are allowed a maximum of 44 hours of downtime per year. But you still need to determine what duration is acceptable for a single outage. A large number of 10-second outages might be acceptable (the total in 44 hours is 15,840 10-second outages); but most likely, a single outage of 44 hours would be unacceptable.

2.2 Time Lines for Outages

The importance of high availability can be seen in the following illustrations, which show the time lines for a computer system outage following a disk crash. Figure 1 shows a sequence of events that might take place when an OLTP client experiences a disk crash on a conventional system using unmirrored disks for data; when the disk crashes, the OLTP environment is unavailable until the disk can be replaced.

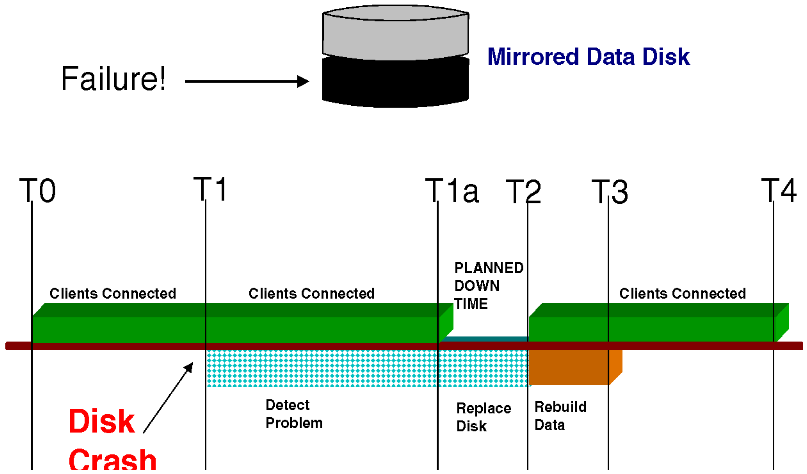
FIGURE 1. Time Line 1: Unplanned Down Time



The crash takes place at T1, and the user’s transaction is aborted. The system remains down until T3, following a hardware replacement, system reboot, and database recovery, including the restoration of data from backups. This sequence can require anything from a few hours to over a day. In this scenario, the time to recovery is totally unpredictable: downtime is unplanned, and therefore out of the organization’s control.

Figure 2 shows the same crash when the system uses a high availability feature known as disk mirroring, which prevents the loss of service.

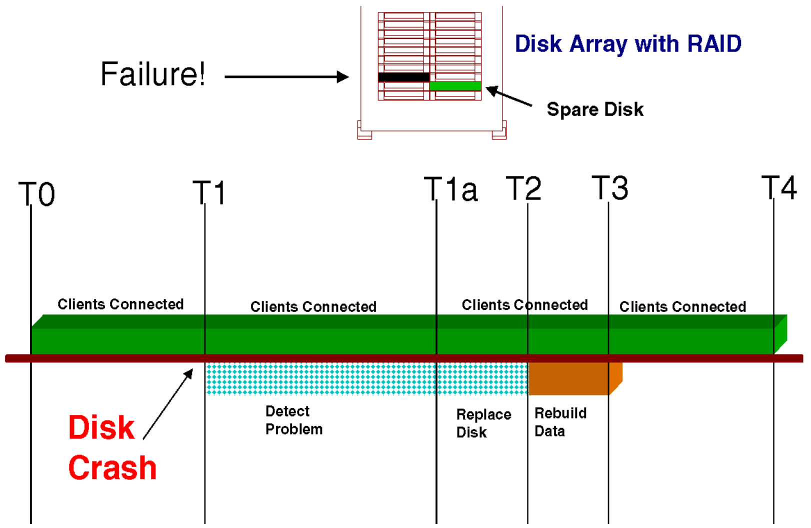
FIGURE 2. Time Line 2: Planned Down Time



When the crash occurs, the mirror disk continues to be available, so no data is lost, and service continues. Further, the replacement of the failed disk can be deferred to a period of planned maintenance. A significant difference between this scenario and the preceding one is that you can predict the amount of time needed for the repair, and you can plan the replacement for the least inconvenient time. With disk mirroring, an unpredictable amount of unplanned downtime is replaced by a shorter known period of planned downtime.

A third scenario, shown in Figure 3, includes a disk array with hot swappable disks. This configuration eliminates all downtime associated with the disk failure.

FIGURE 3. Time Line 3: Down Time Eliminated



When the crash occurs, a spare disk takes over for the failed mechanism. In this case, the disk array provides complete redundancy of disks, and the failed disk may be replaced by hot plugging a new disk mechanism *while the system is running*. After the replacement disk is inserted, the array returns to the state it was in before the crash.

2.3 Causes of Planned Downtime

Planned outages include stopping an application to perform a scheduled backup or install a software patch. Some others:

- Periodic backups
- Software upgrades
- Hardware expansion or repair
- Changes in system configuration
- Data changes

These outages do not normally cause problems if they can be scheduled appropriately. Some data processing environments can tolerate very little planned downtime, if any. Most can tolerate, and plan for, a regular down period every day or week.

An alternative to planned downtime is to carry out maintenance and other system operations while the system is on-line. Backup operations while the system is running are known as **on-line backups**. Hardware upgrades or repairs while the system is running are known as **hot plug operations**.

2.4 Causes of Unplanned Downtime

The following are some common causes of unplanned outages:

- Hardware failure
- File System Full error
- Kernel In-Memory Table Full error
- Backup failure
- Disk full
- Power spikes
- Power failure
- LAN infrastructure problem
- Software defects
- Application failure
- Firmware defects
- Natural disaster (fire, flood, etc.)
- Operator or administrator error

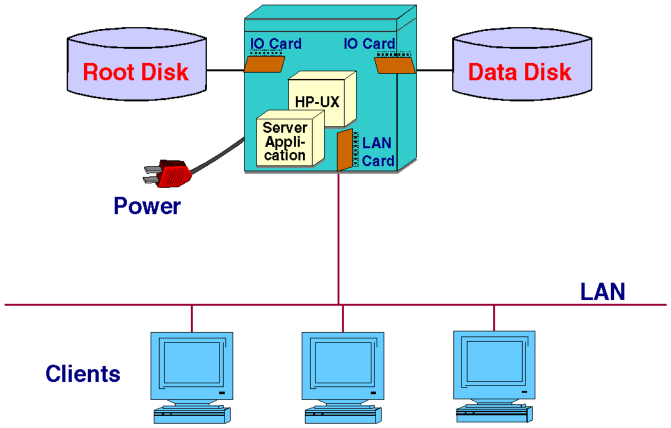
As far as severity is concerned, an unplanned service outage has a far greater negative impact on the enterprise than a planned outage. The effects of unplanned outages include customers waiting in line during a computer crash, airplanes unable to take off or land because of an air traffic control failure, an assembly line coming to a halt, doctors unable to obtain patient data from the Hospital Information System, and so on. In many cases, business is lost because transactions cannot be completed. An unplanned outage most often reduces customer satisfaction.

3.0 Eliminating Points of Failure

High availability computer systems are designed to eliminate or minimize planned and unplanned outages. In any high availability system, it is important to understand the different types of possible failures and how the system responds to them. Not all outages are caused by failures, but failures will definitely cause outages unless you take steps to intercept them. Availability can be seen as a chain of services that must remain unbroken. Failures are breaks in the chain. The weak links are known as **points of failure**. For each link in the chain that is a possible point of failure, you can reduce the chance of outage by providing a redundant or alternate link. This can be done in many ways.

A highly reliable stand-alone system still has many single points of failure. A single point of failure (SPOF) is a hardware or software element whose loss results in the loss of service. Usually, a component that is not backed up by a standby or redundant element becomes a single point of failure. Consider a typical client/server installation on a single HP 9000 system, as shown in Figure 4. Clients — that is, applications running on a PC or UNIX workstation — connect over the network to a server application that is executing on the SPU. The server application reads and writes records on behalf of the clients; these records are stored on the data disk. The HP-UX operating system, located on the root disk, handles client connections, data transfer, memory allocation, and a host of other functions on behalf of the executing application.

FIGURE 4. Reliable Basic System

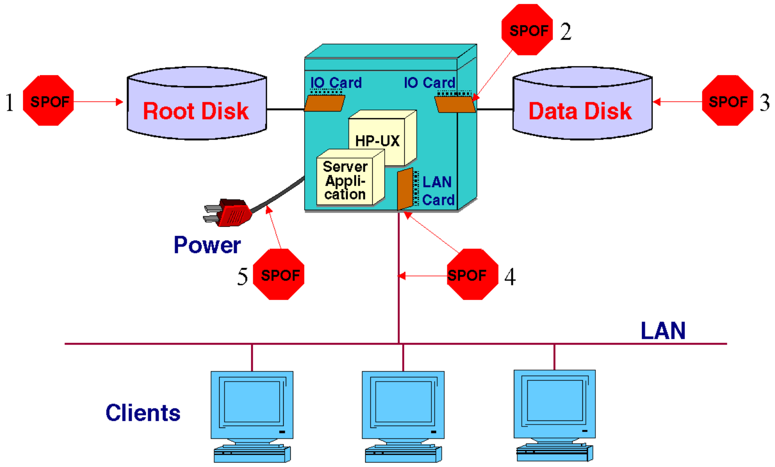


What can go wrong in this scenario? Here are just a few examples:

1. The system goes down because of a failure of the root disk.
2. A disk IO card fails, and transactions are lost.
3. A media failure on the data disk causes data loss and interruption of service.
4. The LAN cable is damaged, or a LAN card fails, and clients lose service.
5. A short power failure results in a system reboot and loss of data.

These components are all single points of failure, as shown in Figure 5.

FIGURE 5. Single Points of Failure



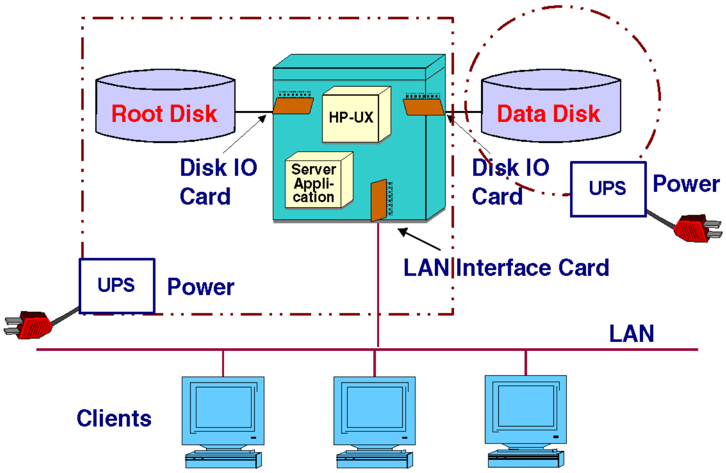
All these SPOFs can be quite easily eliminated by using cluster technologies.

By systematically eliminating these points of failure, you arrive at a design that has minimal downtime.

3.1 Eliminating Points of Failure in the Power Supply

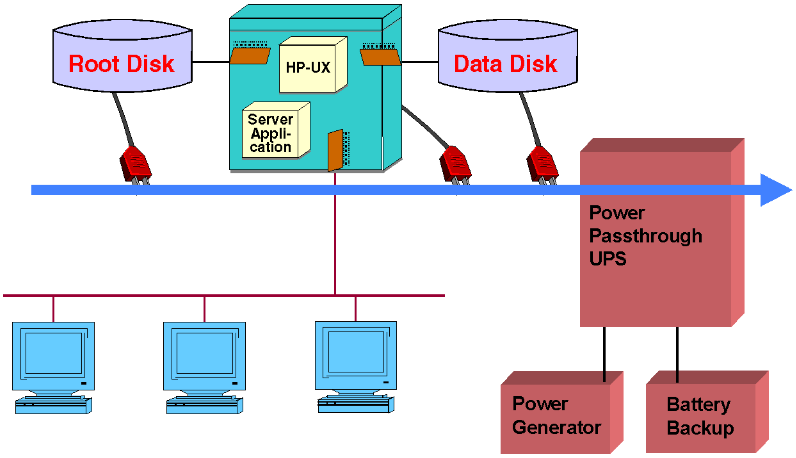
First, eliminate points of failure in the power supply by including **uninterruptible power sources** (UPS) in the configuration, as in Figure 6.

FIGURE 6. Eliminating Single Points of Failure in the Power Supply



The strategy of using a UPS is limited, because battery backup only lasts a short time. Another strategy is the power passthrough approach, in which the loss of power service results in the automatic switch-in of backup systems. This is shown in Figure 7.

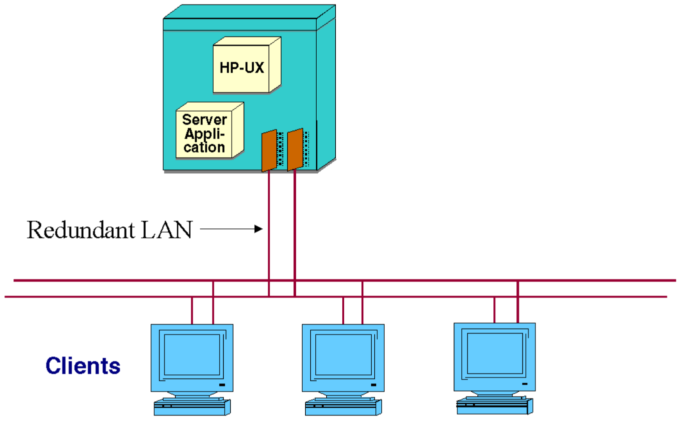
FIGURE 7. Power Passthrough UPS



3.2 Eliminating Points of Failure in Networks

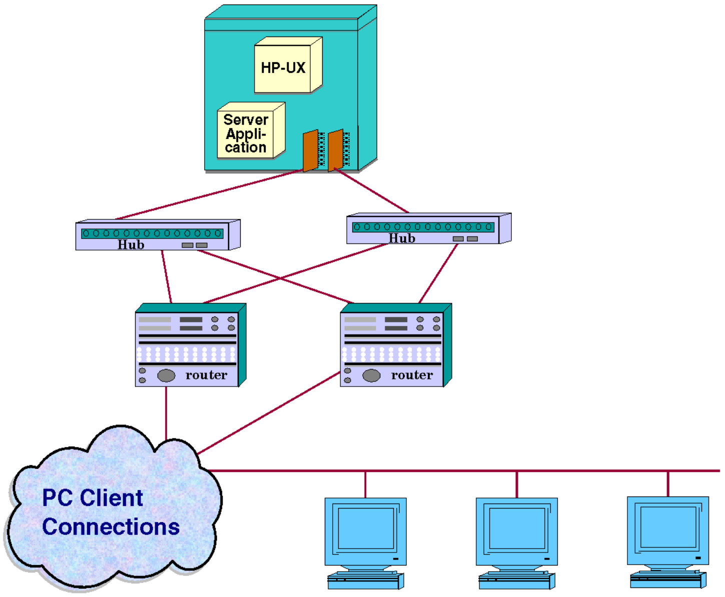
Single points of failure in network cards or cables can easily be eliminated by providing alternative paths for each communication type. This means redundant LAN cards and cables as shown in Figure 8.

FIGURE 8. Redundant LAN Hardware



A more complete approach to eliminating single points of failure in networking must also provide redundancy at the level of routers and switches as in Figure 9.

FIGURE 9. Highly Available Network



3.3 Eliminating Points of Failure in the Disk Subsystem

Another obvious single point of failure is the disks in a conventional reliable system. In Figure 4, two disks were shown—the root disk and a data disk. If there is a media failure on the root disk, the system may be unable to continue normal processing. The disk must be replaced and its contents replaced by reinstalling system software and/or restoring data from backups.

If the data disk fails in a conventional reliable system, the system may remain up, but application processing will stop until a new disk can be installed and the data recovered. For either root or data disk failure, the system must be rebooted, and the data restored from backups. In this case, data will be lost between the time of the last backup and the time of the media failure.

Redundancy is necessary to prevent the failure of disk media or a disk controller from damaging data or causing an outage to users. There are two methods available for providing data protection: using disk arrays in a redundant configuration and using software mirroring. Each approach has its own advantages.

3.3.1 Data Protection with Disk Arrays

One technique for providing redundant data storage is the use of disk arrays in RAID configurations that provide data protection. The acronym RAID stands for *redundant array of inexpensive disks*. A group of disks function together in a variety of configurable arrangements known as RAID levels. Some levels allow hardware mirroring, while others provide protection through the use of parity data, which allows the array to reconstruct lost data if a disk mechanism fails.

Common RAID levels are as follows:

- Level 0: the controller writes data to all disks in stripes. This level provides no data protection.
- Level 1: the controller writes data to mirrored groups of disks.
- Level 3: data is striped byte-wise, and the controller stores parity information on a separate disk so that lost data from any disk can be recovered.
- Level 5: data is striped block-wise, and the controller spreads parity information across all disks so that lost data from any disk can be recovered.

In addition, you can configure arrays in independent mode, which means that each member of the array is seen as an independent disk.

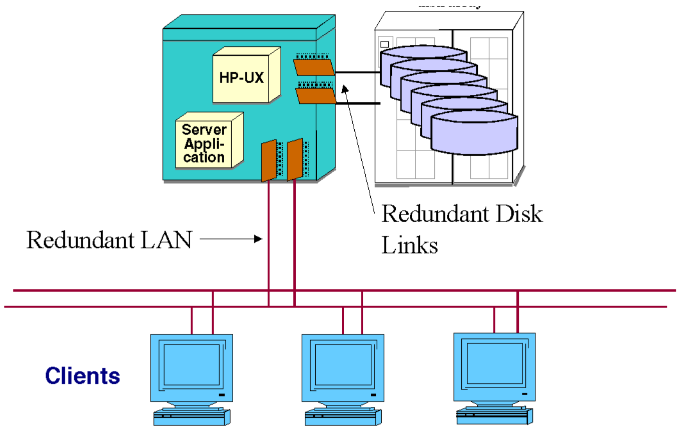
Some of the advantages of using disk arrays for protected data storage are as follows:

- Smaller overall footprint (rack and floor space) for a given amount of storage
- Easy on-line replacement of a failed disk spindle
- Capability of assigning a hot standby spindle to take over for a failed spindle
- Highest storage connectivity (multiple terabytes)
- Flexibility in configuration (different modes available)

- Potential for high performance in small I/O size read-intensive environments
- On some devices, dual controllers, power sources, and fans can eliminate additional single points of failure

A disk array configuration is shown in Figure 10.

FIGURE 10. Disk Array Configuration



3.3.2 Data Protection with Software Mirroring

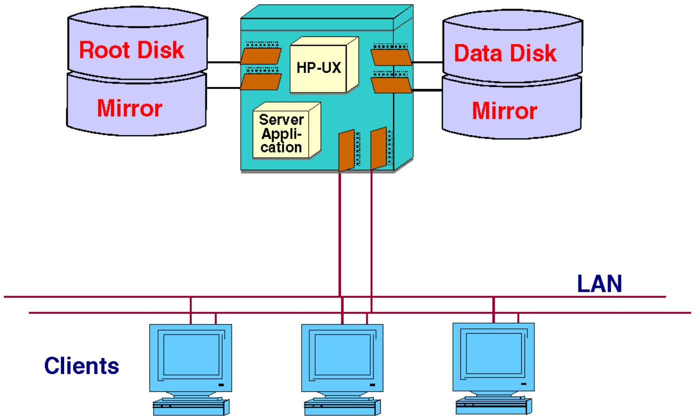
An alternative technique for providing protected data storage is the use of software mirroring, which is an implementation of RAID level 1 on individual disks. In HP-UX, software mirroring is created using Logical Volume Manager and the separate MirrorDisk/UX subsystem. Veritas volume management can also be used to create mirrored volumes.

Mirroring through software has several advantages:

- Two or three way mirroring is possible.
- One mirror copy can be split off for backup, perhaps on another system.
- There is the potential for better performance from the use of multiple read paths and multiple disk controllers.
- Each disk has its own controller, so the loss of one disk tray or tower does not prevent access to the data.
- There is control of data placement for better application performance tuning.
- You can take advantage of the mirrors being configured on different SCSI busses to “double” the I/O rate to a given data area.
- An individual disk controller is not a bottleneck to multiple disks.

Mirrors can be powered from different sources. An example of disk mirroring in software is shown in Figure 11.

FIGURE 11. Disk Mirroring in Software



3.4 Highly Available Stand-Alone System

A growing number of high-end systems today provide redundant components to eliminate many points of failure within a single box. These include:

- Processors
- LAN Interface cards
- IO cards
- Memory components
- Power supplies and connectors
- Hardware Partitions

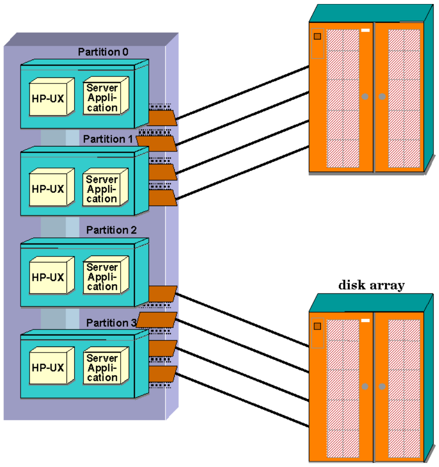
Very high-end servers like the HP Superdome configurations provide hardware cells that operate much like single systems. Cells can be swapped in or out as needed, and they can be combined into one or more partitions each of which runs its own instance of the operating system.

Advantages of systems like the SuperDome are capacity-on-demand, which lets you swap in additional components and pay for them at the time they are deployed. Another feature is extensive diagnostic monitoring by Hewlett-Packard with call-home capability. This kind of monitoring makes it possible to detect upcoming failures before they occur and replace components proactively.

One significant disadvantage of the HA single system is its cost of operation. The ability to hot-plug the central components involves the use of very expensive system designs.

A representation of a highly available stand-alone system is shown in Figure 12.

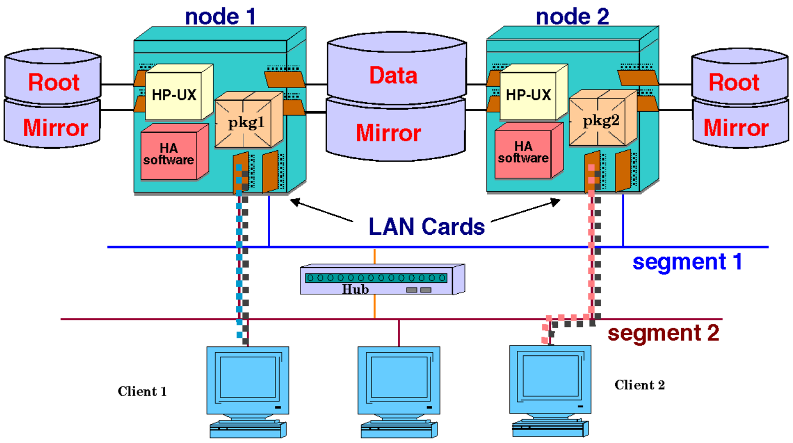
FIGURE 12. Highly Available Stand-Alone System



3.5 Highly Available Clustered System

A clustered system eliminates the CPU and memory components as single points of failure. In a way, this is overkill, because you are actually eliminating the entire system as a point of failure and allowing services to run on an alternate system instead. A cluster can be set up in many different ways, from a simple rack mounting of a networked group of systems to a high availability cluster in which software monitors the membership and health of all the member systems (nodes) within the cluster. An example is shown in Figure 13.

FIGURE 13. High Availability Cluster



A variety of cluster architectures is described in the next section.

4.0 HA Cluster Technology: Types of Architecture

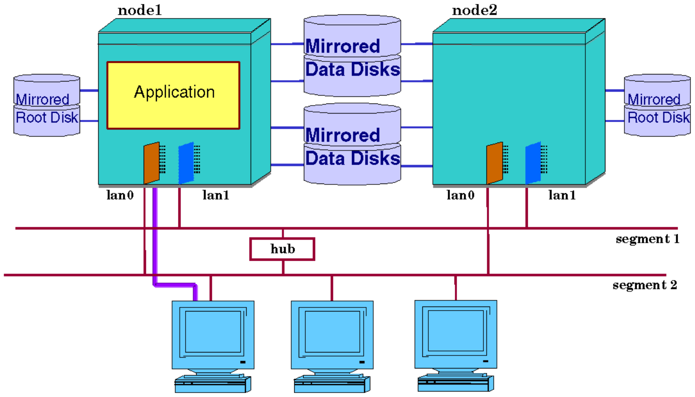
The cluster shown so far in this paper is a generic loosely coupled grouping of HP 9000 systems. In fact, each SPU can be connected to another SPU in a variety of highly available cluster configurations. In this section, we highlight a variety of cluster architectures, including the following:

- Active/Standby Clusters
- Active/Active Clusters
- Parallel Database Clusters
- Standby Parallel Database Clusters

4.1 Active/Standby Clusters

A flexible active/standby configuration is provided by ServiceGuard, which allows the application to start on the standby node quickly, without the need for a reboot. In addition, non-ServiceGuard applications run on the alternate system and continue running after failover. Figure 14 shows a two-node active/standby configuration using ServiceGuard. Applications are running on node 1, and clients connect to node 1 through the LAN.

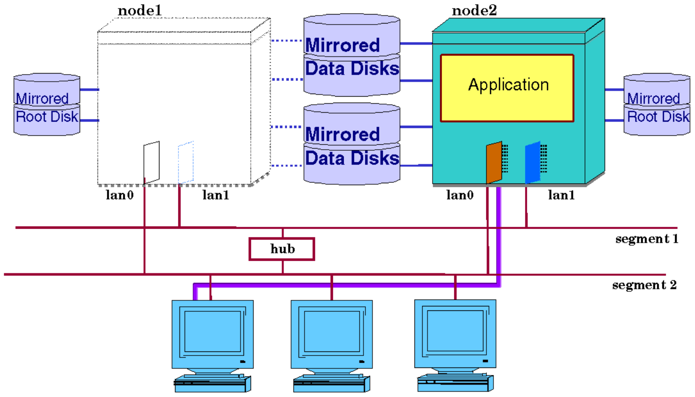
FIGURE 14. Active/Standby Cluster Before Failure



In this configuration, the first node is running the application, having obtained exclusive access to the data disks. The second node is essentially idle, though the operating system and the high availability software are both running.

The state of the system following failover is shown in Figure 15.

FIGURE 15. Active/Standby Configuration after Failure



After failover, the applications start up on node 2 after obtaining access to the data disks. Clients can reconnect to node 2.

Note that failure is not necessary for a package to move within the cluster. With ServiceGuard, the system administrator can move a package from one node to another at any time for convenience of administration. Both nodes remain up and running following such a voluntary switch.

4.2 Active/Active Clusters

In the active/active configuration, two or more SPUs are physically connected to the same data disks, and if there is a failure of one SPU, the applications running on the failed system start up again on an alternate system. In this configuration, different applications may run at the same time on all nodes.

With ServiceGuard there need not be any idle systems; all of the nodes can run mission critical applications. If one node fails, the applications it supports are moved and join applications that are in progress on other nodes.

The primary advantage of the active/active configuration is efficient use of all computing resources during normal operation. But during a failover, performance of applications on the failover node will be somewhat impacted. To minimize the impact of failover on performance, each node should be given the appropriate capacity to handle all applications that might start up during a failover situation.

4.2.1 Cluster Re-formation and Cluster Lock

Under normal conditions, a fully operating ServiceGuard cluster simply monitors the health of the cluster's components while the packages are running on individual nodes. If there is a node failure, the cluster re-forms in a new configuration without the failed node. The use of the **cluster lock** provides a tie-breaking capability in case a communication failure leads to a situation where two equal-sized groups of nodes are both trying to re-form the cluster at the same time. For example, if there is a communication failure in a two-node cluster, each node will independently try to re-form the cluster. To prevent both from succeeding (a situation known as a **split-brain** condition), only one node is allowed to acquire the cluster lock; the other node is halted to prevent any possible data corruption.

4.2.2 How Failover Works

Applications, together with disk and network resources used by applications, are configured in **packages** which can run on different systems at different times. Each package has one or more application **services** which are monitored by ServiceGuard; in the event of an error in a service, a restart or a failover to another node may take place. Any node running in the ServiceGuard cluster is called an **active node**. When you create the package, you specify a **primary node** and one or more **adoptive nodes**.

A particular benefit of ServiceGuard is that you can configure failover to take place following the failure of a package, or following the failure of individual services within a package. You can also determine whether to try restarting services a number of times before failover to a different node.

4.2.2.1 Use of Relocatable IP Addresses

Clients connect via LAN to the server application they need. This is done by means of IP addresses: the client application issues a *connect()* call, specifying the correct address. Ordinarily, an IP address is mapped to an individual hostname — that is, a single HP-UX system. In ServiceGuard, the IP address is assigned to a package and is temporarily associated with whatever host system the package happens to be running on. Thus the client's *connect()* will result in connection to the application regardless of which node in the cluster it is running on.

Figure 16 shows a cluster with separate packages running on each of two nodes. Client 1 connects to a package by its IP address. The package is shown running on node 1, but the client need not be aware of this fact.

FIGURE 16. Active/Active Configuration with Two Applications

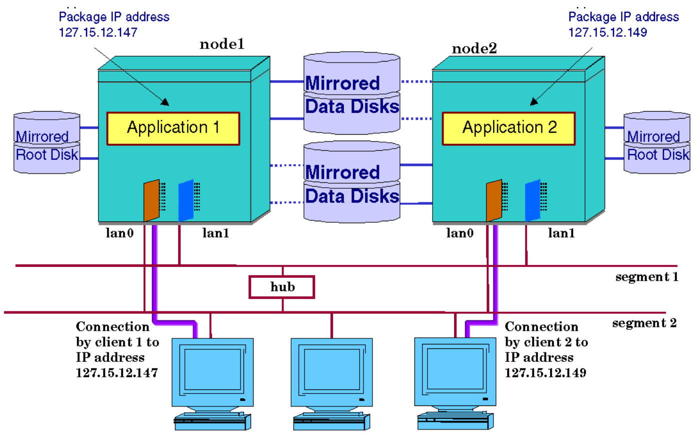
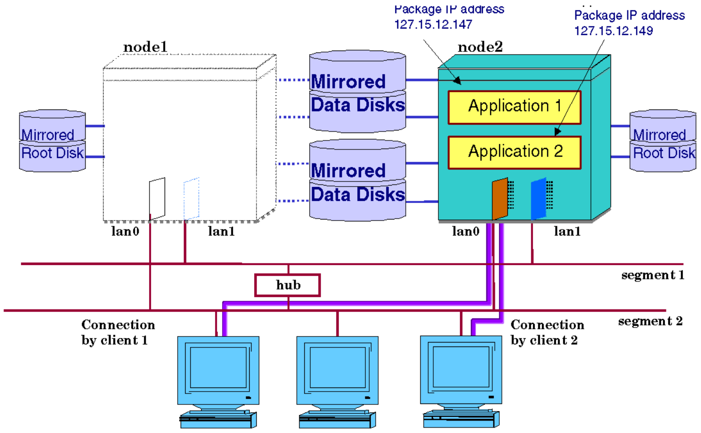


Figure 17 shows an active/active configuration after a failure on node 1. The package moves over to node 2. Note that the IP address of the package is the same. The second node still carries on with the applications that were previously running, but it now also carries the application that had been running on node 1 before the failure.

FIGURE 17. Active/Active Configuration After Failover



In the active/active configuration, ServiceGuard does not use a dedicated standby system. Instead, the applications that were running on the failed node start up on alternate nodes while other processing on those alternate nodes continues.

4.2.3 Fast Recovery from LAN Failures

ServiceGuard monitors the status of the LANs used within each node of the enterprise cluster. If any problem affects the LAN, ServiceGuard will quickly detect the problem and activate a standby LAN within the same node. This detection and fast switch to an alternate LAN is completely transparent to the database and attached clients. This feature eliminates the downtime associated with LAN failures and further strengthens the enterprise cluster environment for supporting mission critical applications.

4.3 Parallel Database Clusters

In the parallel database configuration, two or more SPUs are running applications that read from and write to the same database disks concurrently. This is the configuration used on HP clusters by Oracle Parallel Server (OPS), a relational database product provided by Oracle Corporation. OPS works in conjunction with HP's ServiceGuard OPS Edition software. Oracle Parallel Server (OPS) is a special relational database design. OPS enables multiple instances of the Oracle database to function transparently as one logical database. Different nodes that are running OPS can concurrently read from and write to the same physical set of disk drives containing the database.

In the event one cluster node fails, another is still available to process transactions while the first is serviced. Figure 18 shows the parallel database configuration before the failure of one node.

FIGURE 18. Parallel Database Configuration with Two OPS Instances

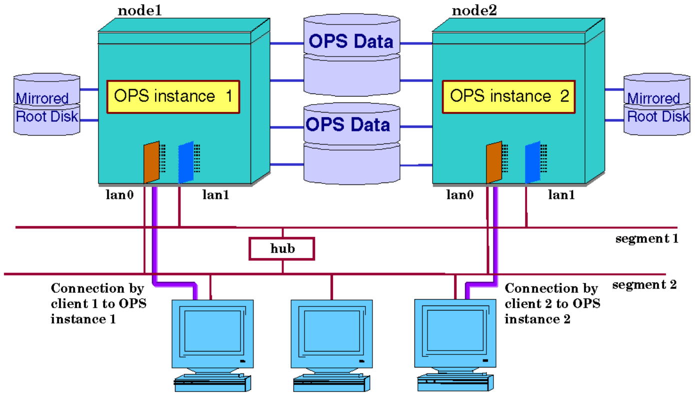
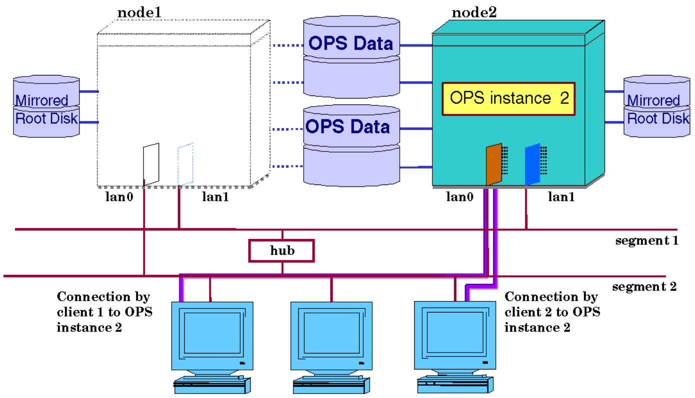


Figure 19 shows the parallel database cluster after the failure of one node. The second node remains up, and users now may access the database through the second node.

FIGURE 19. Parallel Database Configuration After Node Failure



4.3.1 How ServiceGuard Works with OPS

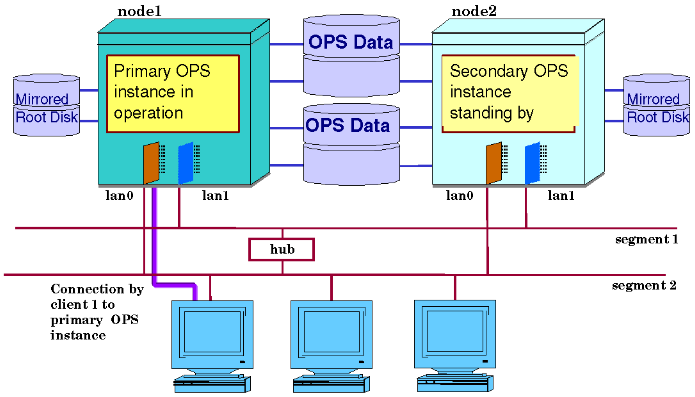
ServiceGuard OPS Edition is a special-purpose high availability software product that allows HP 9000 servers to be configured with OPS. OPS Edition lets you maintain a single database image that is accessed by the HP 9000 servers in parallel, thereby gaining added processing power without the need to administer separate databases. ServiceGuard OPS Edition handles issues of concurrent access to the same disk resources by different servers and ensures integrity of Oracle data.

The OPS Edition uses the same underlying cluster mechanism as ServiceGuard. This means that you can create and manipulate packages as well as OPS instances on the cluster. Note the difference, however: packages run on only one node at a time; whereas OPS applications may run concurrently on all nodes in the OPS cluster.

4.4 Standby Parallel Database Clusters

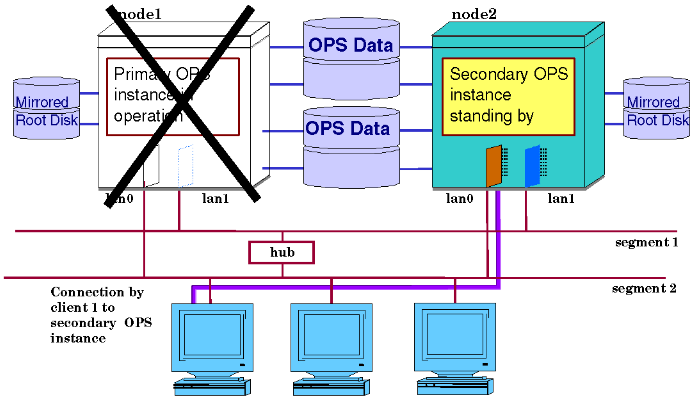
A new database product from Oracle is known as Oracle Parallel FailSafe, which uses the OPS cluster as a means of providing a hot standby environment in which a single Oracle 8i instance can move very rapidly from one HP 9000 system to another. In this model, only one instance at a time is operating on a given database, but the Oracle 8i processes are running, data structures are allocated and buffers are pre-warmed on the failover node, so that failover can take place more quickly than in the case where the Oracle 8i instance must be started from scratch. An example showing a single instance is given in Figure 20.

FIGURE 20. Parallel Failsafe OPS Cluster Before Failure


node1
node2
OPS Data
OPS Data
Primary OPS instance in operation
Secondary OPS instance standing by
Mirrored Root Disk
Mirrored Root Disk
lan0
lan1
lan0
lan1
hub
segment 1
segment 2
Connection by client 1 to primary OPS instance

After the failure of one node, the client connects to the standby instance on the other node, as shown in Figure 21.

FIGURE 21. Parallel Failsafe OPS Cluster After Failure


node1
node2
OPS Data
OPS Data
Primary OPS instance in operation
Secondary OPS instance standing by
Mirrored Root Disk
Mirrored Root Disk
lan0
lan1
lan0
lan1
hub
segment 1
segment 2
Connection by client 1 to secondary OPS instance

5.0 Disaster Tolerant Clusters

Most of the solutions so far described in this paper are implemented within a single data center, with a group of up to 16 clustered systems located in at the same geographical site. *Disaster tolerant solutions* actually extend the scope of high availability from the level of the hosts in an individual data center to the level of alternate data centers on the same site, alternate sites in a large metropolitan area, or over distances of hundreds or thousands of miles. They do this by placing cluster nodes in locations that are remote from each other, or by creating backup clusters that can run the mission critical applications in the event of disaster. In this way, the cluster itself can be eliminated as a point of failure.

In a ServiceGuard cluster configuration, high availability is achieved by using redundant hardware to eliminate single points of failure. This protects the cluster against hardware faults, such as LAN, disk, or node failure.

For some installations, this level of protection is insufficient. Consider the order processing center where power outages are common during harsh weather. Or consider the systems running the stock market, where multiple system failures, for any reason, have a significant financial impact. For these types of installations, and many more like them, it is important to guard not only against single points of failure, but against **multiple points of failure (MPOF)**, or against single massive failures that cause many components to fail, as in the failure of a data center, of an entire site, or of a small area. A **data center**, in the context of disaster recovery, is a physically proximate collection of nodes and disks, usually all in one room.

Creating clusters that are resistant to multiple points of failure or single massive failures requires a different type of cluster architecture called a **disaster tolerant architecture**. This architecture provides you with the ability to fail over automatically to another part of the cluster or manually to a different cluster after certain disasters. Specifically, the disaster tolerant cluster provides appropriate failover in the case where a disaster causes an entire data center to fail.

5.1 Single Cluster Solution: MetroCluster

As the name suggests, a **metropolitan cluster** uses a technology that operates within a metropolitan area; nodes may be separated by geographic distances no greater than 50 kilometers. A metropolitan cluster operates much like an ordinary ServiceGuard cluster, with some additions:

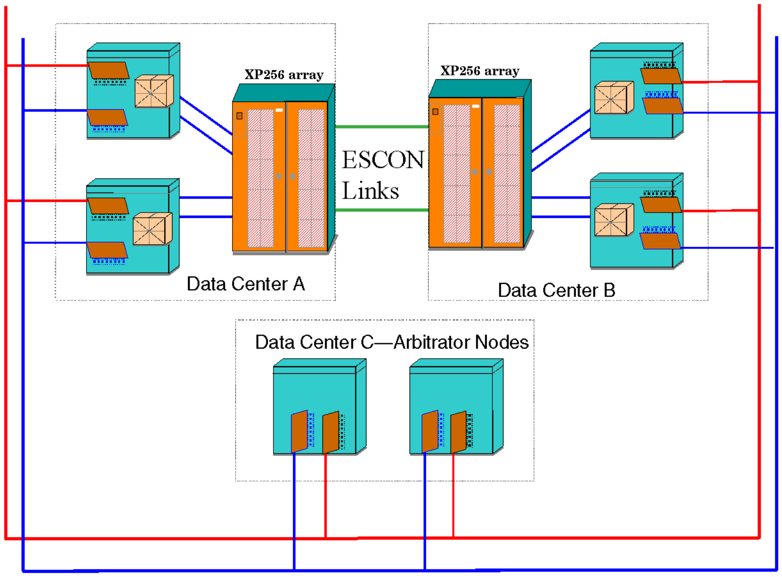
- Special disk array hardware and software for data replication are used.
- Special control scripts are employed for packages.
- Arbitrator systems are used instead of disk locks.

A metropolitan cluster has nodes located in different parts of a city or in adjacent cities. Putting nodes further apart increases the likelihood that alternate nodes will be available for failover in the event of a disaster. However, there is a tradeoff: Increasing the distance also increases the expense of the cluster and the complexity of the environment.

The use of arbitrators provides a tie-breaking capability in case a communication failure leads to a situation where two equal-sized groups of nodes, each group at a different data center, are trying to re-form the cluster at the same time. The arbitrators decide which data center will succeed, and the other nodes are halted.

Metropolitan clusters are very similar to ordinary clusters with the exception that metropolitan clusters often require right-of-way from local governments or utilities to lay network and data replication cable. This can complicate the design and implementation. Metropolitan clusters also require a different kind of tie-breaker system for ensuring that split-brain situations do not arise. Typically, metropolitan clusters use an arbitrator site containing additional cluster nodes instead of the cluster lock disk. An example of a metropolitan cluster is shown in Figure 22.

FIGURE 22. MetroCluster Architecture



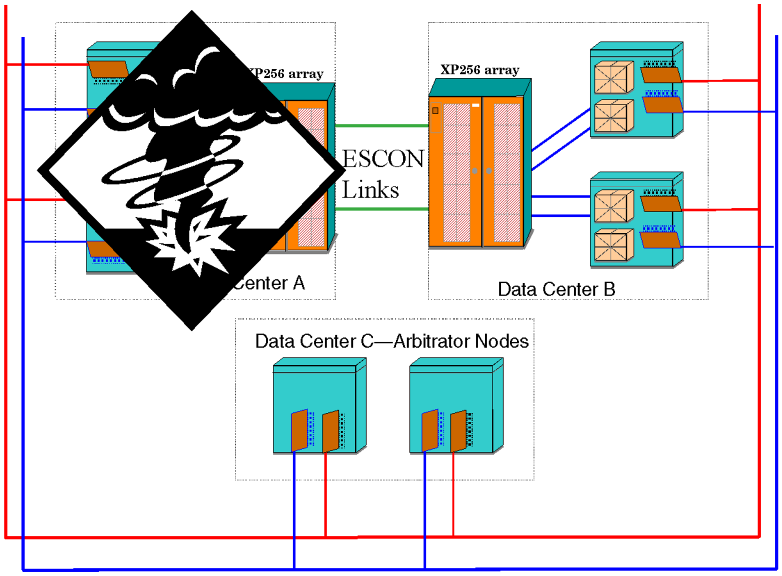
The distance separating the nodes in a metropolitan cluster is limited by the data replication and network technology available. Metropolitan cluster architecture is implemented through two HP products:

- MetroCluster with Continuous Access XP
- MetroCluster with EMC SRDF

A key characteristic of metropolitan clusters is the data replication technology used. Ordinary clusters can use mirroring or RAID to provide redundant data within a single data center. But disaster tolerant solutions must protect against the loss of an entire data center, and this means there must be separate copies of data in different data centers. To achieve this, metropolitan clusters use data replication based on the capabilities of the HP SureStore E Disk Array XP256 or the EMC Symmetrix array, which allow long distances between copies of the data. Data is replicated between data centers using special XP 256 or EMC software products.

What happens, then, in a disaster involving a metropolitan cluster? Suppose an explosion destroys Data Center A in the example shown in Figure 22. The result is given in Figure 23—operations continue at Data Center B.

FIGURE 23. MetroCluster Following a Disaster

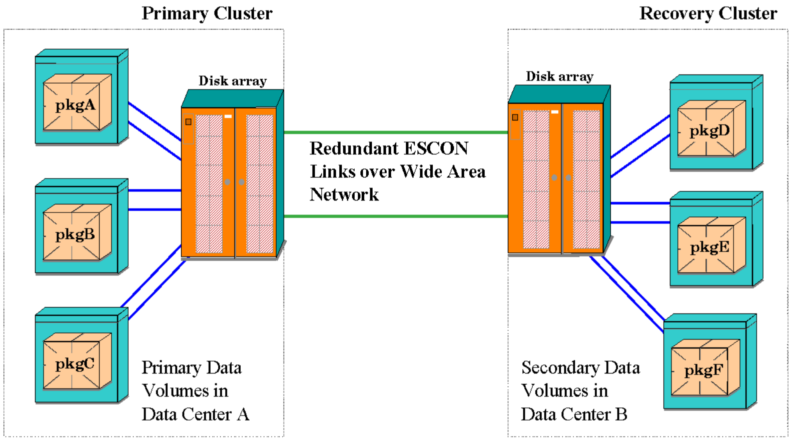


5.2 Multiple Cluster Solution: Continental Cluster

A **continental cluster** provides alternate *clusters* that are separated by large distances so that wide area networking is used between them. This allows you to configure clusters that back each other up over wide geographic distances and permits a failover from one cluster to another. All protected applications are started in the new location when a determination is made that the other cluster is no longer available.

The design is implemented with two distinct ServiceGuard clusters located in different geographic areas. In this architecture, each cluster maintains its own quorum, so an arbitrator data center is not used. ContinentalClusters is architected to use any WAN connection via a TCP/IP protocol; however, due to data replication needs, high speed connections such as T1 or T3/E3 leased lines or switched lines may be required. Continental cluster architecture is shown in Figure 24.

FIGURE 24. Continental Cluster

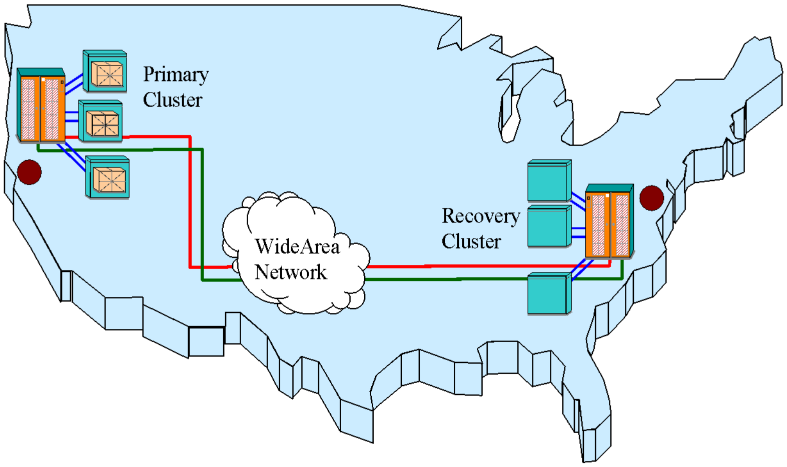

Primary Cluster
Recovery Cluster
Disk array
Disk array
pkgA
pkgB
pkgC
pkgD
pkgE
pkgF
Redundant ESCON Links over Wide Area Network
Primary Data Volumes in Data Center A
Secondary Data Volumes in Data Center B

The key issues concerning a WAN cluster are:

- Inter-cluster connections for ContinentalClusters are TCP/IP based connections.
- The physical connection is one or more leased lines managed by a common carrier. Common carriers cannot guarantee the same reliability that a dedicated physical cable can. The distance can introduce a time lag for data replication, which creates an issue with data currency. This could increase the cost by requiring higher speed WAN connections to improve data replication performance and reduce latency.
- Tools such as Transaction Processing Monitors or database replication tools that work across a WAN are needed to make sure the data replication maintains data consistency.
- Operational issues, such as working with different staff with different processes, and conducting failover rehearsals, are made more difficult the further apart the nodes in the cluster are.

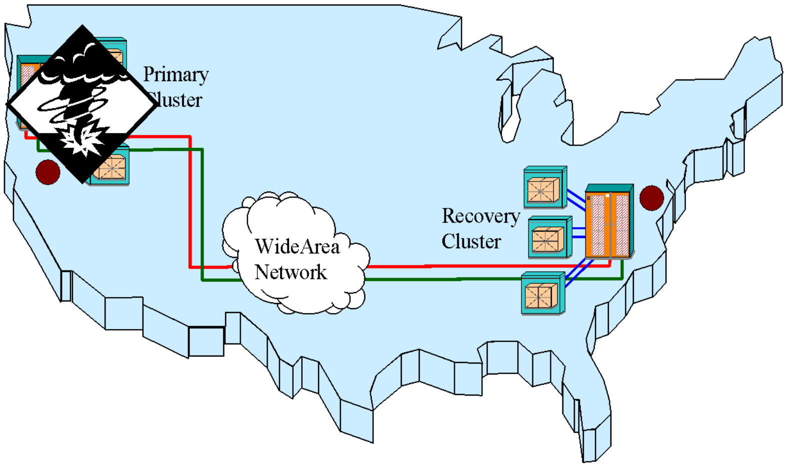
Figure 25 shows a continental cluster operating normally with applications running on the West coast and systems available on the East coast for quick failover.

FIGURE 25. Continental Cluster in Normal Operation



The aftermath of a disaster on the West Coast is shown in Figure 26.

FIGURE 26. Continental Cluster in Disaster Recovery Mode



Continental clusters are implemented using the following product:

- HP ContinentalClusters—This product is used in conjunction with ServiceGuard clusters that are set up in two or more separate geographic regions such that one cluster backs up another. This solution uses physical data replication provided by the EMC Symmetrix or HP's XP Series disk arrays, together with high-speed long-distance links and ESCON converters. As an alternative, it is possible to use logical data replication via software products such as Oracle Standby Database and similar products from other vendors.

5.3 Extensions to ContinentalClusters

Extensions of the basic continental cluster architecture might include the following:

- Two data centers backing each other up
- One recovery data center backing up several primary data centers
- Metropolitan clusters backed up by a recovery cluster in another geographic region

6.0 Monitoring and Management Tools

6.1 ServiceGuard Manager

A new generation of monitoring tools is represented in the Java-based GUI ServiceGuard Manager, which displays cluster maps and shows all the objects in a cluster together. ServiceGuard Manager also lets you save graphical images of clusters and data files containing the details of cluster implementation.

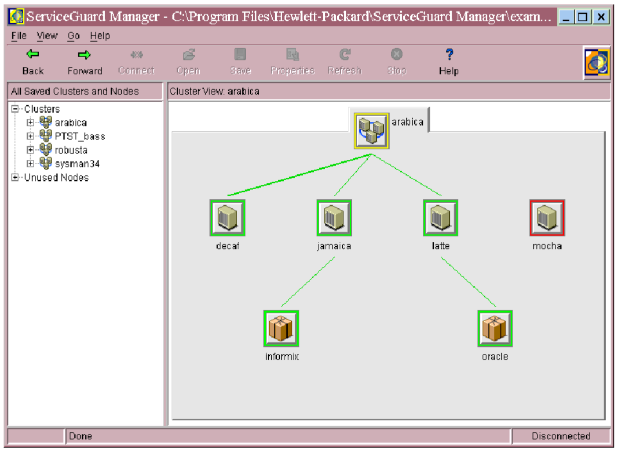
ServiceGuard Manager provides a graphical map that displays all the configured objects in a ServiceGuard cluster and shows all their properties on property sheets. This tool lets you see a cluster’s configuration at a glance and determine whether everything is running correctly. Understanding status in this way lets the administrator quickly identify the areas where intervention are needed.

This kind of mapping tool with the use of color codes, flags, and so forth, is actually a very sophisticated extension of the simple basic monitor described in the beginning of this chapter. The graphical user interface builds an analog for the “flashing light” on the device that indicates failure.

Of course, ServiceGuard Manager is doing far more than that as well. It shows the relationships among the cluster objects, and it allows you to view the properties of particular parts of the cluster. It permits refreshing the display to show the dynamic shifting of states among the cluster objects, and it lets you capture data and graphics in files for later use.

In a typical display, cluster objects appear in a Windows Explorer-style list at the left, and a map of the selected objects appears on the right, as shown in Figure 27.

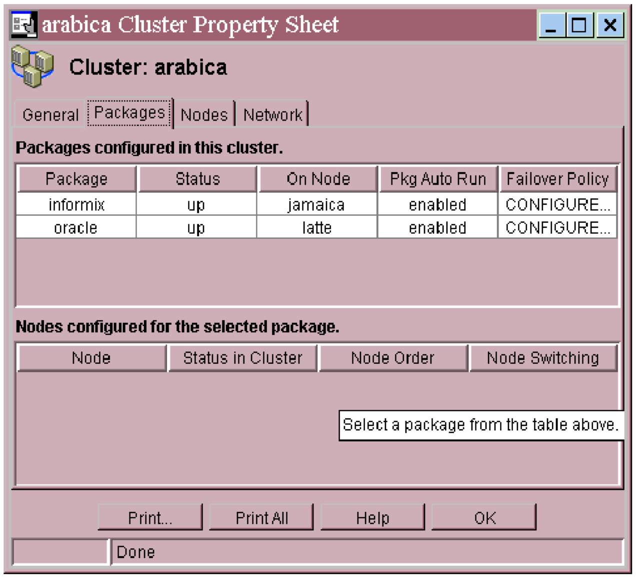
FIGURE 27. ServiceGuard Manager Object List



You can capture a graphic image of this map in a file for use in documenting the cluster at any point in its life.

Properties of clusters, nodes, packages and other objects can be seen through a very complete set of property sheets. An example of the property sheet for a specific cluster is shown in Figure 28.

FIGURE 28. ServiceGuard Manager Property Sheet



Cluster: arabica

Packages configured in this cluster.

Package	Status	On Node	Pkg Auto Run	Failover Policy
informix	up	jamaica	enabled	CONFIGURE...
oracle	up	latte	enabled	CONFIGURE...

Nodes configured for the selected package.

Node	Status in Cluster	Node Order	Node Switching

Select a package from the table above.

Done

Sheets like this can be used to plan changes to the cluster and to observe characteristics of the cluster as it runs. When the state of an object changes—for example, when a package is moved from one node to another—the property sheets show the details as soon as the display is refreshed.

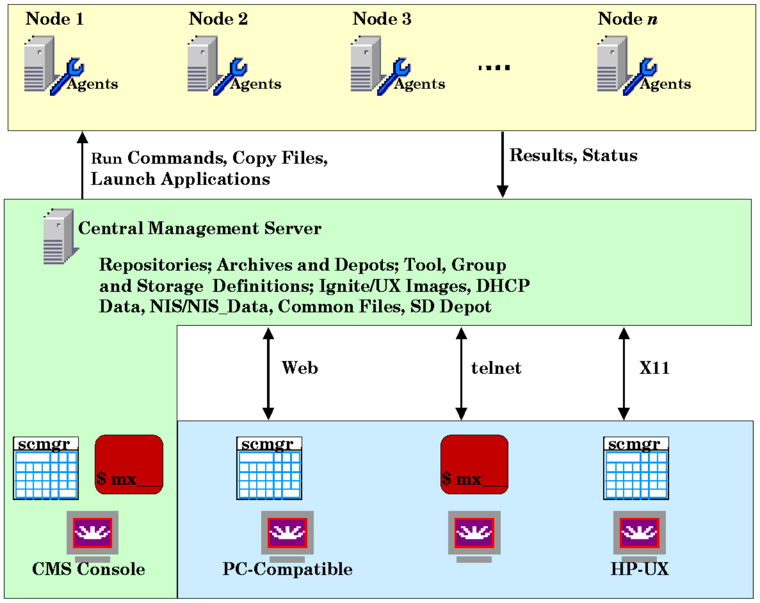
6.2 ServiceControl Manager

Today's high availability clusters are increasingly complex, with a large number of nodes and a growing array of hardware components. To avoid the problem of complexity leading to unwanted downtime, it is essential to choose system management tools that simplify and streamline the routine tasks needed to keep the cluster running smoothly. This applies to all system management tasks, from the creation of accounts through the definition of networked file systems. But for clusters, it is essential to carry out the following multi-node tasks carefully:

- Installing software
- Managing the cluster configuration
- Managing security

HP's ServiceControl Manager (SCM) provides an environment that can simplify these processes. Using SCM, you create a central management server that contains a repository of the software and configuration settings required for your environment. Then, using distributed management tools, you can quickly set up a group of systems with identical software versions, appropriate kernel configurations, similar cluster software configurations and security arrangements. Figure 29 shows a high-level view of this approach.

FIGURE 29. ServiceControl Central Management

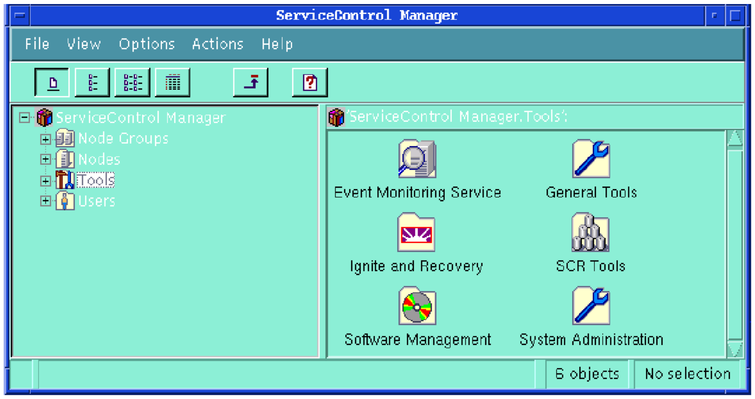

Node 1
Node 2
Node 3
....
Node *n*
Agents
Agents
Agents
Agents
Run Commands, Copy Files,
Launch Applications
Results, Status
Central Management Server
Repositories; Archives and Depots; Tool, Group
and Storage Definitions; Ignite/UX Images, DHCP
Data, NIS/NIS_Data, Common Files, SD Depot
Web
telnet
X11
scmgr
$ mx
CMS Console
scmgr
PC-Compatible
$ mx
scmgr
HP-UX

6.2.1 SCM Tools

ServiceControl Manager is the graphical user interface that coordinates the activities of defining the software content and configuration of groups of systems and then taking steps to automate the creation of groups of correctly configured systems. The Tools screen from this interface is shown in Figure 30.

The following sections describe some of these tools in a little more detail.

FIGURE 30. ServiceControl Central Management Tools



6.2.1.1 Event Monitoring

Event monitoring is the configuration of monitors across groups of nodes. As shown earlier in this chapter, monitors provide early detection of faults that can result in loss of availability. This section of ServiceControl lets you configure monitors across groups of cluster nodes rather than individually on each node. Where large groups of systems are involved, setting up monitors can be time-consuming without special tools like this.

6.2.1.2 Ignite and Recovery

This area in ServiceControl lets you define the content of a system very precisely and then implement the definition on particular nodes using HP’s Ignite/UX software. You can also restore systems to a known baseline or recover systems whose software has become non-functional or misconfigured. This area also lets you capture and store baseline images of software that can be implemented on other systems.

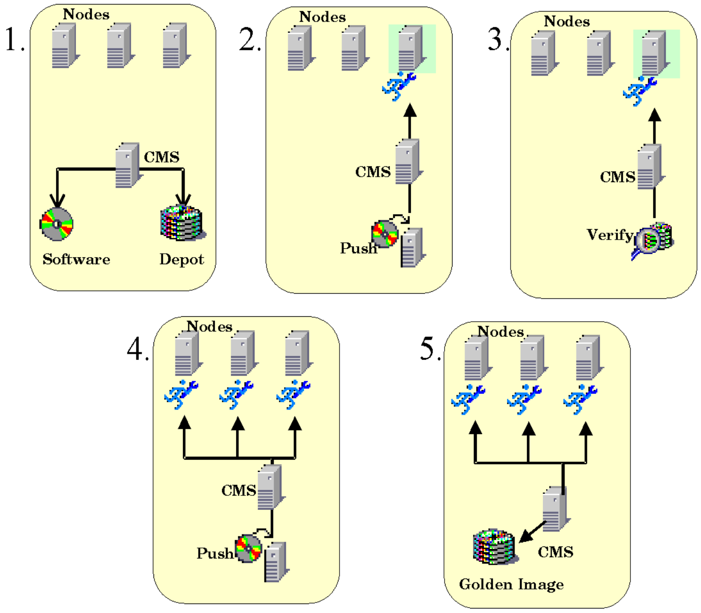
6.2.1.3 Software Management

ServiceControl Manager also provides a set of software management tools that can be used on multiple systems. The essential functionality is provided by Software Distributor. Through SCM, you can create Software Distributor depots from which the installation of multiple systems can be staged. After installing software once on the central management station, you can then install to a large group of nodes in the most efficient manner. The process has several operations:

1.Create a depot.
2. Install from the depot to a system.
3. Verify the installation
4. Install to multiple systems
5. Create a baseline golden image of the configuration

Some of these operations are shown in Figure 31.

FIGURE 31. ServiceControl Software Management



6.3 Event Monitoring Services

A high availability environment requires proactive system administration to keep the components of a cluster running. This means *monitoring the cluster*—carefully observing the behavior of cluster components, replacing faulty hardware or software as soon as possible. In the simplest terms, monitoring means checking specific information on a system to detect only unusual circumstances within the cluster or one of its parts. In these cases, the **Event Monitoring Service** (EMS) lets you choose the type of monitoring you want to do and decide how the information from monitoring is to be used.

Specific monitors are available as a part of HP’s High Availability Monitors product, which includes a disk monitor, database monitors, and MIB monitors for system and cluster status monitoring. In addition, EMS **hardware monitors** are supplied along with many HP and third-party products such as disk arrays, network components, and others.

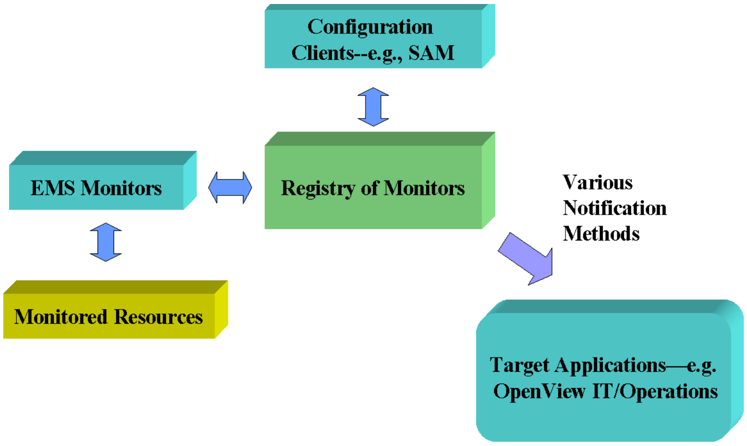
6.3.1 Using Individual EMS Monitors

EMS monitors give you control over the response to particular changes in status, allowing you to send messages that trigger administrative response. The Event Monitoring Service is a framework with the following:

- Tools for configuring resource monitoring
- A group of different notification methods for sending messages when a resource experiences an unusual event or reaches a critical value
- Easy integration of new EMS resource monitors using a standard API

The central focus of monitoring is on the **resource**, that is, the thing you are monitoring. Monitors track the occurrence of predefined conditions known as **events** that are considered important within the high availability framework. An event could be the failure of a component, loss of availability (that is, detection of a single point of failure), or even the gradual degradation of a cluster element that indicates the element should be replaced. Figure 32 shows the process at its simplest.

FIGURE 32. EMS Monitor Operation


Configuration Clients--e.g., SAM
EMS Monitors
Registry of Monitors
Various Notification Methods
Monitored Resources
Target Applications—e.g. OpenView IT/Operations

7.0 Additional Sources of Information

Many types of information are available about Hewlett-Packard HA products. This lists a few resources

7.1 Books and User’s Manuals

- Peter S. Weygant, *Clusters for High Availability: a Primer of HP-UX Solutions*. Contains basic information similar to that presented in this paper. Hewlett-Packard Professional Books: ISBN 0-13-494758-4.
- *Managing MC/ServiceGuard*. HP Part Number B3936-90026.
- *Configuring OPS Clusters with ServiceGuard OPS Edition*. HP Part Number B5158-90026.

- *Designing Disaster Tolerant High Availability Clusters*. HP Part Number B7660-90003.
- *Using High Availability Monitors*. HP Part Number B5736-90025.
- *HP-UX ServiceControl User's Guide*. HP Part Number B9105-90001.

7.2 White Papers (available on web sites listed below)

- Bob Sauers, “Understanding High Availability”
- Pam Dickerman and Michael Hayward, “Highly Available Networks”
- Bob Sauers, “Disaster Tolerant Highly Available Cluster Architectures”
- Hayden Brown, “Application Consolidation”

7.3 Web Sites

- docs.hp.com: Technical documentation for HP-UX systems
- docs.hp.com/hpux/ha: Technical documentation for High Availability solutions
- www.hp.com/go/ha: Marketing
- Other web sites are available through HP Sales and Support personnel.