SUN BLUEPRINTS APPROVED

Robust Clustering: A Comparison of Sun™ Cluster 3.0 versus Sun Cluster 2.2 Software

By Tim Read - Sun Microsystems, Inc. and
Don Vance - Horizon Open Systems UK

Sun BluePrints™ OnLine - September 2001

Sun microsystems

http://www.sun.com/blueprints

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303 USA
650 960-1300 fax 650 969-9131

Part No.: 816-2123-10
Revision 01, 09/06/01
Edition: September 2001

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" ET AUCUNE GARANTIE, EXPRESSE OU IMPLICITE, N'EST ACCORDEE, Y COMPRIS DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DE LA PUBLICATION A REPONDRE A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.

Robust Clustering: A Comparison of Sun™ Cluster 3.0 versus Sun Cluster 2.2 Software

Introduction

Sun coined the phrase *The Net Effect* (`http://www.sun.com/neteffect`) to highlight the impact that increases in processor power, network bandwidth, and number of internet connected users and devices are having on the inter-networked data center. Applications must be re-architected as highly available network services, accessible to anyone, any time, anywhere, and on any device. This makes availability, something that has always been a key metric for data center managers, of paramount importance. The Sun™ Cluster 3.0 software can help to deliver the availability and enhanced scalability needed by data center managers and others.

Strong clustering products, such as the Sun Cluster 3.0 software, integrate tightly with the host operating system and interconnect technology, something only the operating system vendor can achieve. Developed as an extension to the Solaris™ 8 Operating Environment (Solaris OE), it is an integral part of a SunPlex™ solution combining Sun servers, storage, Solaris OE, and networking connectivity products to deliver Sun's Service Point Architecture vision. New ease of management features empower customers to "*manage the service, not the server,"* allowing consolidation and load balancing of multiple applications across the available resources a Sun cluster provides.

Sun Cluster 3.0 software is also a key component of Sun's SunUP™ Network availability program, focusing on a people, process, and product approach to delivering availability. SunUP Network service program allows SunTone^{sm} credential partners to deliver customers *webtone* levels of data availability.

Finally, Sun's SunTone™ Platforms group have created a number of Cluster Platforms (standard configurations) and Database Platforms (also known as VOS configurations). These combine an integrated hardware and software stack with best configuration and implementation practices, enabling customers to purchase cluster solutions quickly and easily. For more information, see `http://www.sun.com/integratedplatforms`.

This document is aimed at a technical audience. It gives a component-by-component comparison of the Sun Cluster 3.0 software with its predecessor, the Sun Cluster 2.2 software, and shows how the new product delivers greater levels of robustness and functionality.

An overview of Sun's clustering technology

The term "clustering" is widely used throughout the computer industry, and covers a range of architectures and topologies for connecting computers together to perform some computing task.

The Sun Cluster product addresses availability and scalability of business applications. Using multiple Sun servers tightly coupled together provides a way of bounding the outage time that applications experience when serious hardware or operating system errors occur. It can also provide scalability to specially written applications, most notably Oracle 8*i* Parallel Server (OPS) and Oracle 9*i* Real Application Clusters (RAC).

Cluster configurations demand redundancy of all hardware components. Configurations have multiple servers, interconnects, public network connections, and RAID protected storage.

Sun Cluster 2.2 software

Prior to the release of the Sun Cluster 2.0 software in October 1997, Sun had two products that offered clustering facilities named: Solstice™ HA and Sun Cluster PDB. Solstice HA, initially released in December 1995, was designed to support failover applications; its final, 1.3, release came in April 1997. Sun Cluster PDB, initially released in August 1995, supported parallel databases: Oracle Parallel Server, Informix XPS, and Sybase MPP. Its final, 1.2, release came in October 1996. The Sun Cluster 2.0 software started to merge the functionality of the two products into a single offering.

Sun Cluster 2.2 software, released in April 1999, completed this process. It delivered support for both High Availability (HA) applications using a simple failover mechanism, and enabled support for specific scalable applications in which a single instance of an application uses resources on more than one node concurrently. This support was restricted in Oracle Parallel Server versions 7.x and 8.x.

Architected as a layered product that ran on top of the Solaris OE, the software made use of shell scripts, user level programs, and daemons to provide its functionality.

Features include:

- Support for Solaris 2.6, 7, and 8 OE.
- Support for Solstice DiskSuite™ and Veritas Volume Manager (VxVM) software.
- Application support for Oracle (standard and Parallel Server), Sybase, SAP, NFS, DNS, iPlanet™ software products, Apache, Tivoli, and many other applications.
- Support for up to four nodes.
- Four server/storage topologies: clustered pair, N+1, ring, and scalable.

Sun Cluster 3.0 software

Sun Cluster 3.0 software, released in December 2000, is the result of nearly six years of research, development, and testing work stemming from the Solaris MC project begun by Sun Labs in early 1995. For more information, see `http://research.sun.com/research/techrep/1996/abstract-57.html`. The product offers enhanced functionality compared to its predecessor, including: global devices, a global file service, and global networking. The new features form a superset of that found in the previous release, with an additional class of scalable applications being possible where certain criteria are met. See: `http://www.sun.com/software/whitepapers/wp-clusterapi/`

Sun Cluster 3.0 software forms a core part of Sun's Service Point Architecture (SPA) vision, allowing data centers to consolidate applications onto available hardware, network, and storage resources, while ensuring that user service levels agreements are met. SPA addresses the problems of data center complexity and server sprawl by allowing resources to be brought together in a SunPlex solution running Sun Cluster 3.0 software and managed by the Sun Management Center 3.0 software. The consolidation of resources simplifies application provisioning, implementation, change, and systems management, allowing application service levels to be managed dependent on business requirements.

Unlike 2.2, the Sun Cluster 3.0 software is tightly integrated with the Solaris 8 OE. Many of the components that previously existed as user level programs have now become kernel agents and modules.

Features include:

- Global devices, global file, and global network services.
- Ease of installation and administration through the SunPlex Manager, a browser based cluster tool.
- Ease of agent development (both scalable and HA) through the SunPlex Agent Builder.
- Monitoring through Sun Management Center 3.0 software.
- Support for the Solaris Logical Volume Manager (previously known as Solstice DiskSuite software), Veritas Volume Manager.
- Application support for Oracle (standard and RAC), Sybase, SAP, NFS, DNS, LDAP, and iPlanet software products and Apache web servers (failover and scalable mode).
- Co-existence with the Solaris Resource Manager™ 1.2 software.
- Three server/storage topologies: clustered pairs, pair + M, and N+1.
- Currently supports up to eight nodes, and more nodes will be added in future releases.

Cluster interconnects

The cluster interconnects, also known as the private interconnects (in 3.0) or heartbeat networks (in 2.2), are made up of physical network connections solely for the use of the Sun Cluster framework. The cluster interconnects provide the infrastructure for the transmission of heartbeat messages to determine connectivity between nodes, and thus decide on membership; application level messages, e.g. for the Oracle Parallel Server DLM and RAC *Cache Fusion* data, and data transfer for the Sun Cluster 3.0 software global features. See `www.oracle.com` for more details of Oracle 9*i* RAC and Cache Fusion.

Both Sun Cluster 2.2 and 3.0 products require a minimum of two connections for resilience. Because Sun Cluster 3.0 software places far greater demand on the interconnects, a maximum of six interconnects are supported. This offers increased redundancy, greater scalability, and throughput. The interconnects are automatically striped by the cluster framework, so multiple requests generated by any of the global components can transfer data in parallel, without any effort on the users part.

The currently supported network interface cards for both products include 100baseT and Gigabit Ethernet. Additionally, Sun Cluster 2.2 software supports SBus and PCI SCI cards. For clusters containing more than two nodes, multiple switches are used to achieve connectivity, rather than simple back-to-back cables.

The Sun Cluster 2.2 Switch Management Agent (SMA)

The Sun Cluster 2.2 Switch Management Agent daemon, smad, is a user level daemon responsible for:

- Managing communication sessions on the private interconnect.
- Performing SCI heartbeat checks on remote nodes (for SCI based clusters).
- Detecting communication failures between the nodes.
- Performing failover to the next subnet where applicable.
- Informing the Cluster Membership Monitor (CMM) when a change in cluster membership is required.
- Acting as an interface for other software components if they require the cluster interconnect status.

The smad process communicates with its peers by sending and receiving UDP packets at regular intervals. This is often described as the cluster heartbeat mechanism. It also interfaces to a kernel SMA module called smak. Unlike the Sun Cluster 3.0 product, only one heartbeat network is active at any one time.

Sun Cluster 3.0 software transport infrastructure

Sun Cluster 3.0 software places far greater demands on the private interconnects than Sun Cluster 2.2 software. In addition to the heartbeat messages flowing back and forth, the interconnects are also used to transfer data and requests for global devices, global file systems, and global networking components described in the new features in Sun Cluster 3.0 software.

Although the physical inter-node connectivity is achieved in the same way, the transport software infrastructure is provided by a number of kernel agents rather than the user level process in Sun Cluster 2.2 software. The implementation brings a number of benefits:

- Support for up to six interconnects (although there is no inherent upper limit).
- All interconnects are used simultaneously by means of dynamic load balancing.
- Interconnects can be enabled / disabled or added / deleted dynamically.
- Ease of administration through a GUI or CLI for control, status, or configuration.
- Kernel agents provide zero copy optimizations.

The Cluster Membership Monitor (CMM)

Sun Cluster 2.x and 3.0 products define a concept of *membership* as a group of nodes that can successfully communicate with every other node in the group via the private interconnect infrastructure. This concept is critical for the success of a cluster product that is effectively performing distributed computing operations. It is the task of the Cluster Membership Monitor (CMM) to ensure that only one cluster invocation is in progress at any one time.

To determine membership, and more importantly, ensure data integrity, the CMM must:

- Account for change in cluster membership, such as a node joining or leaving the cluster.
- Ensures that a "faulty" node leaves the cluster.
- Ensures that the "faulty" node stays out of the cluster until it is repaired.
- Prevents the cluster from partitioning itself into subsets of nodes.

The CMM therefore protects a cluster against:

- *Split brain* (both 2.2 and 3.0) - when all communication between nodes is lost and the cluster becomes partitioned into sub-clusters, each believing that it is the only partition.
- *Amnesia* (3.0 only) - when the cluster restarts after a shutdown with cluster configuration data older than at the time of the shutdown.

Changes in cluster membership drive the cluster reconfiguration sequence that may, in turn, result in services being migrated from failed or faulty nodes to healthy ones.

The Sun Cluster 2.2 CMM

The Sun Cluster 2.2 CMM is implemented as the `clustd` process. It ensures that any cluster membership transition should result in a majority (N/2+1) of the nodes from the previous stable incarnation being present in the surviving one. Note that there are exceptions to this rule; such as when all nodes lose communication and it would be undesirable for all of them to shut down, such as in a 1:1:1:1 split. For cluster configurations containing more than two nodes, the algorithm is quite complex and beyond the scope of this paper. For a complete description of the process see: *Failure Fencing for Sun Cluster 2.2*; whitepaper by Geoff Carrier and Paul Mitchell; 2000, Sun Microsystems, Inc.in the References section.

During the establishment of the new, VxVM based cluster, the CMM may seek to reserve a *quorum* disk to break the tie in cases of a one-one split. A quorum disk is a nominated device in the shared storage connected to the relevant nodes. The reservation is enacted as a SCSI-2 ioctl. The node that is granted the reservation causes the second attempt to fail. A coin being tossed serves as a good analogy.

The SCSI-2 reservation ioctl used is part of the SCSI-2 command set. This is commonly implemented in most modern disk firmware. However, the reservation call is neither persistent, capable of surviving reboots, nor able to cope with multiple paths to the same device. This precludes the use of either alternate pathing (AP) or dynamic multi-pathing (DMP) software.

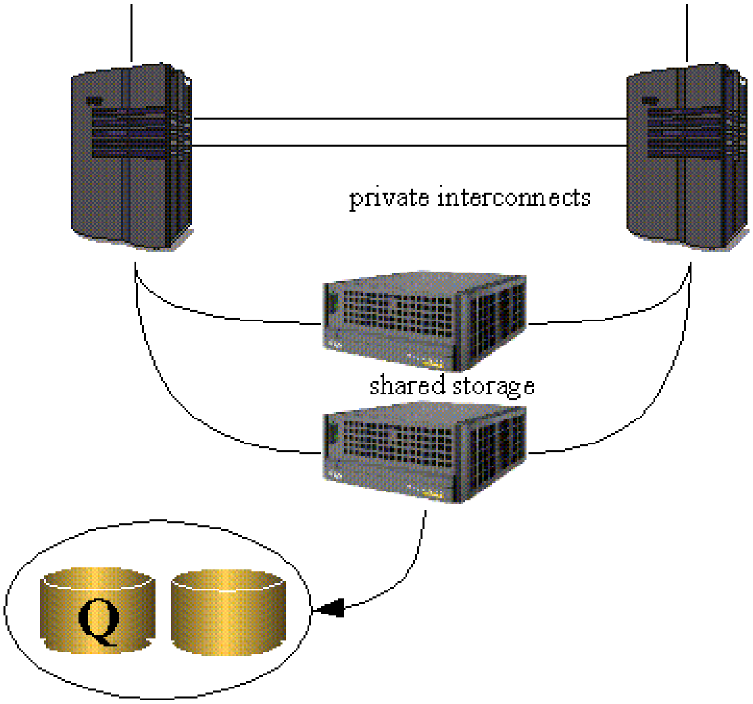

FIGURE 1 Example quorum disk allocation in a cluster pair

The node losing the reservation race then drops out of the cluster, and the surviving node placing a SCSI reservation on all the disks in the diskgroups it now owns to *fence off* the faulty node and ensure that it cannot corrupt the data. In the event that the cluster was running Oracle Parallel Server, under CVM, and an I/O was pending on the faulty node, then the node will be panicked by the failfast driver.

Clusters that use Solstice DiskSuite software employ a different mechanism for failure fencing. They also do not use the concept of a quorum disk. The mechanism is very briefly outlined here.

Under normal circumstances, each node will have a SCSI reservation on the metasets they own. In the event of a loss of communication, nodes will release the reservations on the metasets they own, and attempt to reserve the sets belonging to the other nodes. A some point a cluster node will detect a reservation conflict and panic. The remaining node will then take over control of this diskset and re-establish the reservation on the all the sets they own.

Once the new cluster membership has been established, either through a fault or an administrative operation, the reconf_ener process will continue to execute the reconfiguration steps. There are twelve steps, with the last four being concerned with taking control of disksets and restarting logical hosts (data services) on the cluster nodes. All of the logical hosts that are owned by the physical node are restarted serially. Therefore, the failover time will be the sum total of all the diskset import and application restart times.

Sun Cluster 3.0 CMM

Sun Cluster 3.0 software implements its cluster membership monitor as a kernel module. This ensures that it will not be as heavily affected by resource starvation as user level daemons are, and also means that time-outs can be more tightly controlled to allow faster failure detection. The CMM receives information about connectivity to other nodes via the cluster transport mechanism.

The algorithm used to determine cluster membership differs from Sun Cluster 2.2 implementation, and is independent of the volume management product used. Each node within the cluster has a vote; there are V_n in total. The cluster can also be configured with up to V_n-1 nominated quorum disks from the shared storage, V_q in total. The cluster will therefore have $V_t=V_n+V_q$ votes present when all members are operational. For any cluster partition to survive, a majority ($V_t/2+1$) of these V_t vote must be present. If nodes need to compete for quorum disk votes to gain a majority, then the nodes attached to the shared quorum disks race to reserve them in the same order. The node, and consequently the partition, that wins the race for the first quorum disk stays up, while the other partition node will panic out of the cluster. Nodes in the winning partition then place a persistent SCSI reservation on the shared storage to prevent errant nodes from corrupting the data.

If a multi-node cluster with n node does not have n-1 quorum disk assigned, then a point will be reached when the number of the remaining nodes m, and the number of quorum disks q, falls below majority, i.e. $m+q < (n+q)/2+1$. Therefore, in some circumstances the remaining nodes of a cluster will panic due to loss of majority.

The initial version of Sun Cluster 3.0 software only supports dual hosted storage. In order for the CMM to work correctly, and be able to successfully protect shared storage, SCSI-3 Persistent Group Reservation (PGR) functionality is needed within the disk firmware. This allows persistent reservations to be placed on multi-hosted disks that grant access to a specified set of nodes while denying others.

The mechanism described has an added benefit; when rebooting cluster nodes, a partition will not start until it has successfully obtained the majority of the V_t votes. If the node is connected to shared storage containing one or more quorum disks, and yet was not part of the previous cluster partition, then the persistent reservation on the disks, held by a different node, will prevent the other nodes from obtaining their vote. This ensures that configuration information in the cluster configuration repository will always be the most current version.

Once a new membership has been established, the CMM will signal the user land Resource Group Manager daemon, `rgmd`, to start the reconfiguration process. This is a major differentiating factor between the Sun Cluster 2.2 and 3.0 products. The Sun Cluster 3.0 product parallels the import of diskgroups and restarting of data services wherever there is no explicit service dependency, whereas Sun Cluster 2.2 software runs them sequentially. This can lead to Sun Cluster 3.0 software having a considerably short failover time.

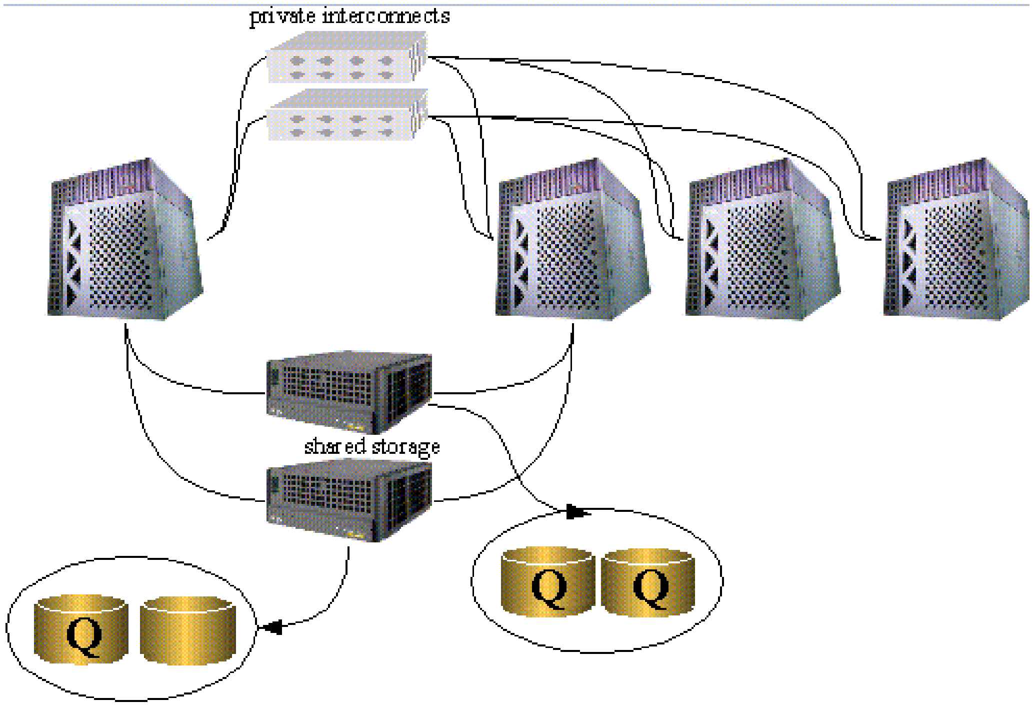

FIGURE 2 Example quorum disk allocation in a Pair + 2 configuration

Cluster configuration control

Managing configuration changes within a cluster is crucial to prevent data corruption or system outages through the use of stale or inaccurate data. Sun Cluster 2.2 and 3.0 software both implement cluster configuration databases, or repositories, to hold information about the current configuration, the state and location of the services they provide, and the ownership of attached storage among other things. The primary concern of any implementation is consistency of data and the prevention of temporal inconsistency (also known as amnesia).

The Sun Cluster 2.2 implementation calls the dynamic portion of this component the Cluster Configuration Database (CCD), and it is implemented as a user level process (the CCD daemon or `ccdd`). Sun Cluster 3.0 software takes its control, along with many other features, into the kernel, where the feature is known as the Cluster Configuration Repository (CCR). The two implementations are outlined and how the CCR, in Sun Cluster 3.0 software, provides a more robust solution is shown.

The Sun Cluster 2.2 configuration framework

The CCD framework is used to maintain a consistent, valid, cluster-wide database of configuration information.

Unlike Sun Cluster 3.0 software, nodes in a Sun Cluster 2.2 system do not boot directly into a cluster configuration. Instead, a cluster must be initiated, or a node joined, by the system administrator issuing a `scadmin` command. Each node interrogates their copy of the cluster database file (`/etc/opt/SUNWcluster/conf/<cluster_name>.cdb`) to learn about the cluster members and its topology. Without this information, the CMM would be unable to function. This information is not replicated between nodes, so any changes have to be manually replicated by an administrator. Note that inconsistencies in the `cdb` files can prevent a cluster node from starting. Once the cluster framework (`clustd`) has started, and the reconfiguration steps are being executed, the CCD is queried to assess which services (logical hosts) need to be started on various cluster nodes.

Each node in the cluster holds a copy of the CCD information. Updates and changes can only be made while there is a majority (>50%) of the potential copies available; the cluster nodes are online and part of a stable cluster. For configurations with more than two nodes, this usually does not present any problems. However, for two node clusters, both nodes must be present for a majority to be achieved, and this is a highly undesirable and restrictive requirement.

To overcome this problem and still allow consistent changes to be made while only one node was left in a two node (VxVM based) cluster, a shared CCD is implemented. The shared CCD requires two dedicated disks be placed into their own diskgroup, within which there is a single 10MB file system which holds copies of the CCD data files. Note that these disks cannot be used for any other purpose or data storage; this is one of the disadvantages of this implementation. The shared CCD becomes active whenever only one node remains in the cluster.

The ccdd ensures that any node starting a new cluster always receives the latest configuration information by checking the timestamps and checksums on the local and shared CCD (if present). In this fashion, the cluster ensures that information is both valid and up-to-date. Any subsequent changes by the ccdd continue to validate the files via their checksum, and rollback any changes that do not complete on every node.

In clusters with three or more nodes, it is possible to suffer from *amnesia*. In circumstances where the first node into the new cluster was not part of the previous majority sub-cluster, where changes had been made, its CCD may, potentially, be invalid. Best practice would demand that measures were taken to ensure this did not happen.

The Sun Cluster 3.0 configuration framework

The configuration framework used by Sun Cluster 3.0 software is known as the cluster configuration repository (CCR). As with Sun Cluster 2.2 software, the information is stored in flat ASCII files on the root file system. However, the mechanism for ensuring the validity of the CCR is completely different, and ensures that amnesia can not occur.

A Sun Cluster 3.0 system relies on a simple majority voting mechanism to decide whether a node or set of nodes can form a cluster. Unlike Sun Cluster 2.2 software, nodes boot directly into a cluster, unless otherwise directed by the system administrator, though use of the '-x' flag to the boot command. In order to complete the boot process, a majority of cluster votes must be obtained. Every cluster node and nominated quorum disk have a vote. As long as a simple majority can be achieved, then the cluster completes the boot process.

Once booted, kernel drivers keep copies of the CCR on each node in step using a two-phase commit protocol. Changes can continue to be made even on a cluster with only one remaining member. Under these circumstances, the votes of locally attached quorum disks will have given the node a majority. If this node is subsequently shut down, persistent reservation keys placed on the quorum disks will prevent other nodes from obtaining their votes and forming a new cluster. This ensures that until a node of the last cluster incarnation is booted, no new cluster can form that might otherwise pick up stale CCR data.

The CCR implementation has a number of advantages over its Sun Cluster 2.2 software equivalent:

- The CCR is updated via kernel drivers that are not subject to the same level of potential resource (CPU, memory) starvation as user level implementations.
- The CCR is a highly available kernel service.
- Amnesia is strictly prevented from occurring using the cluster membership established via the kernel based CMM. When the cluster is up, the CCR can be updated.
- Additional dedicated disk storage is not required.
- The CCR consists of multiple tables. The invalidity of one or more tables does not preclude updating of consistent tables. The active portion of the CCD is a single table in a flat file.

New Features in Sun Cluster 3.0 software

Sun Cluster 3.0 software offers several new features that enhance the functionality provided by Sun Cluster 2.2 software. These features and benefites are outlined.

Global devices

Every disk, tape, CD-ROM device, and VxVM volume and SVM meta-device, becomes a *global* device in Sun Cluster 3.0 software. These global devices are transparently accessible from any cluster node, regardless of their physical connection to a node or nodes. Each device is assigned a unique name and major/ minor number pair by the device ID (DID) pseudo driver. These devices then appear in the cluster namespace as part of the `/dev/global` hierarchy, allowing an administrator to manage and use these devices as if they were local. Operations on remote devices are mediated by the cluster transport infrastructure.

```
Phys-hardy# scdidadm -L | more
1         phys-hardy:/dev/rdsk/c0t0d0    /dev/did/rdsk/d1
2         phys-hardy:/dev/rdsk/c0t1d0    /dev/did/rdsk/d2
3         phys-floppy:/dev/rdsk/c1t9d0   /dev/did/rdsk/d3
3         phys-hardy:/dev/rdsk/c1t9d0    /dev/did/rdsk/d3
4         phys-floppy:/dev/rdsk/c1t10d0  /dev/did/rdsk/d4
4         phys-hardy:/dev/rdsk/c1t10d0   /dev/did/rdsk/d4
                                      .
                                      .
13        phys-floppy:/dev/rdsk/c2t13d0  /dev/did/rdsk/d13
13        phys-hardy:/dev/rdsk/c2t13d0   /dev/did/rdsk/d13
14        phys-floppy:/dev/rdsk/c2t14d0  /dev/did/rdsk/d14
14        phys-hardy:/dev/rdsk/c2t14d0   /dev/did/rdsk/d14
15        phys-floppy:/dev/rdsk/c0t0d0   /dev/did/rdsk/d15
16        phys-floppy:/dev/rdsk/c0t1d0   /dev/did/rdsk/d16
phys-hardy# newfs /dev/global/rdsk/d16s2
newfs: /dev/global/rdsk/d16s2 last mounted as /mnt
newfs: construct a new file system /dev/global/rdsk/d16s2: (y/n)? y
/dev/global/rdsk/d16s2: 4154160 sectors in 2733 cylinders of 19 tracks, 80
        sectors 2028.4MB in 86 cyl groups (32 c/g, 23.75MB/g, 5888 i/g)
super-block backups (for fsck -F ufs -o b=#) at:
 32, 48752, 97472, 146192, 194912, 243632, 292352, 341072, 389792, 438512,
 487232, 535952, 584672, 633392, 682112, 730832, 779552, 828272, 876992,
 925712, 974432, 1023152, 1071872, 1120592, 1169312, 1218032, 1266752,
                                     .
                                     .
 3697632, 3746352, 3795072, 3843792, 3892512, 3941232, 3989952, 4038672,
 4087392, 4136112,
phys-hardy#
```

TABLE 1 `newfs` on a non-local global device

Where a global device is only connected to a single node (or the device is a tape or CD-ROM), then there is only a single access path to that device. Failure of this path or its controlling node will render that device unavailable. Devices that are dual hosted however, have two possible paths through which physical I/Os requests can be serviced. One of these paths is active and is known as the primary path, while the subsequent paths are passive and known as secondaries.

The status of any I/O is synchronized (in the respective kernels) between the primary and secondary path, so the secondary path can take over transparently in the event of a primary path failure or a manual switch over. The process of fail/switch-over will introduce a delay into the I/O, but the I/O will be completed exactly as it would have done in the absence of the fail/switch-over. No application changes are needed to benefit from this new functionality.

Global devices provide a homogeneous device namespace across all the cluster nodes, removing any requirements for application configuration changes if the applications are moved from node to node. Global devices also have the added advantage that they do not require a systems administrator to be aware of, or care, where a device is connected in order to perform a command like `newfs`, `tar`, or `ufsdump`.

The nearest equivalent functionality within Sun Cluster 2.2 software allows a uniform device namespace to be achieved through the use of VxVM disk-groups or SVM metasets. Unlike Sun Cluster 3.0 software, the devices they contain are only visible on the node on which they are mastered, and only appear on another node after they are manually moved or have migrated due to a node failure. The Cluster Volume Manager (CVM) functionality of VxVM is a special case, as it is only used for Oracle Parallel Server. They lack any high availability features described previously (indeed, the proceeding examples simply have no analogue). Administrators are also required to be cognizant of where the disk-group / metaset is currently hosted in order to perform a command on their volumes or meta-devices.

Global network service

Sun Cluster 3.0 software provides a new global networking feature. A *global IP* address, or Global InterFace (GIF), is installed on a network interface for a particular subnet on a cluster node, known as the GIF node, or GIN. This IP address also appears on the loopback interfaces (lo0:1, lo0:2, etc.) of cluster nodes that host scalable services with a dependency on this IP address. Scalable services can be brought up on these nodes, and yet still bind to the global IP address. Incoming IP packets destined for a scalable service are then accepted via a GIF. Before they pass up the IP stack, they are first examined by a Packet Despatch Table (PDT) driver, which has the option of three policies for packet distribution: *weighted, ordinary sticky,* and *wildcard sticky.*

A *weighted* policy hashes the packets, based on the source IP and port number, into "buckets" associated with the subscribing node. The packets are then sent to the relevant node over the cluster transport infrastructure, appear on the loopback interface, and can be accepted by a receiving application. When an application responds, its outgoing packets are sent via its local network interface card, thus providing scalable outbound IP traffic.

The alternative *sticky* policies allow concurrent application-level sessions over multiple TCP connections to share in-state memory (application session state). An e-commerce site that fills a shopping cart via HTTP on port 80 and receives payments using SSL on port 443 would use *ordinary sticky.* A passive mode FTP initially connecting port 21, and then reconnecting to a dynamically allocated port, would use *wildcard sticky.*

```
Phys-floppy# ifconfig -a
lo0: flags=10008c9<UP,LOOPBACK,RUNNING,NOARP,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
lo0:1: flags=10088c9<UP,LOOPBACK,RUNNING,NOARP,MULTICAST,PRIVATE,IPv4> mtu
8232 index 1
        inet 172.16.193.2 netmask ffffffff
lo0:2: flags=10088c9<UP,LOOPBACK,RUNNING,NOARP,MULTICAST,PRIVATE,IPv4> mtu
8232 index 1
        inet 129.159.54.181 netmask ffffffff
hme0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 129.159.54.160 netmask ffffff00 broadcast 129.159.54.255
        ether 8:0:20:7d:8a:70
hme2: flags=1008843<UP,BROADCAST,RUNNING,MULTICAST,PRIVATE,IPv4> mtu 1500
index 3
        inet 172.16.0.130 netmask ffffff80 broadcast 172.16.0.255
        ether 8:0:20:7d:8a:70
hme2:1: flags=1008843<UP,BROADCAST,RUNNING,MULTICAST,PRIVATE,IPv4> mtu 1500
index 3
        inet 172.16.194.5 netmask fffffffc broadcast 172.16.194.7
hme1: flags=1008843<UP,BROADCAST,RUNNING,MULTICAST,PRIVATE,IPv4> mtu 1500
index 4
        inet 172.16.1.2 netmask ffffff80 broadcast 172.16.1.127
        ether 8:0:20:7d:8a:70
phys-floppy# rlogin phys-hardy
Password:
Last login: Wed May 23 18:20:31 from phys-floppy
Sun Microsystems Inc.   SunOS 5.8       Generic February 2000
phys-hardy# ifconfig -a
lo0: flags=10008c9<UP,LOOPBACK,RUNNING,NOARP,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
lo0:1: flags=10088c9<UP,LOOPBACK,RUNNING,NOARP,MULTICAST,PRIVATE,IPv4> mtu
8232 index 1
        inet 172.16.193.1 netmask ffffffff
hme0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 129.159.54.161 netmask ffffff00 broadcast 129.159.54.255
        ether 8:0:20:7d:78:1a
hme0:1: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 129.159.54.182 netmask ffffff00 broadcast 129.159.54.255
hme0:2: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 129.159.54.181 netmask ffffff00 broadcast 129.159.54.255
hme2: flags=1008843<UP,BROADCAST,RUNNING,MULTICAST,PRIVATE,IPv4> mtu 1500
index 3
        inet 172.16.0.129 netmask ffffff80 broadcast 172.16.0.255
        ether 8:0:20:7d:78:1a
hme2:2: flags=1008843<UP,BROADCAST,RUNNING,MULTICAST,PRIVATE,IPv4> mtu 1500
index 3
        inet 172.16.194.6 netmask fffffffc broadcast 172.16.255.255
hme1: flags=1008843<UP,BROADCAST,RUNNING,MULTICAST,PRIVATE,IPv4> mtu 1500
index 4
        inet 172.16.1.1 netmask ffffff80 broadcast 172.16.1.127
         ether 8:0:20:7d:78:1a
```

TABLE 2 global IP address (129.159.54.181) plumb in on hme0:2 and lo0:2

The GIF is also made highly available through a combination of the Public Network Monitoring (PNM) facility within a single node and resource group failover across nodes. This ensures that in the event of a NIC failure on the GIN, its IP addresses are transferred to a standby card. In the event of a node failure, the IP addresses are transferred to equivalent NICs on another cluster node. Global IP addresses can also be migrated under administrative control. In each case, the global network service will be continuously available from the IP perspective. Packets that are dropped while the interface is being moved will be re-transmitted as part of the standard TCP-IP recovery mechanisms (UDP packet loss will be handled in the application layer itself as per normal). From the perspective of external applications, the global IP address behaves in an identical fashion to IP addresses on any single server, and no application changes are required.

Sun Cluster 3.0 software allows both many-to-many and one-to-many relationship between services and IP addresses. Multiple IP addresses can be used for a single service, or multiple services can use a single IP address.

Sun Cluster 2.2 software has no equivalent global networking functionality, and cannot be used to implement scalable services.

Global file service

The Sun Cluster 3.0 global file service is another feature that has no equivalent under Sun Cluster 2.2 software. The global file service provides cluster-wide file systems, allowing a uniform global directory namespace to be created and maintained. A global file system is always mounted on every node of the cluster on the same mount point. Nodes that subsequently join the cluster replay these mount commands to achieve consistency.

Cluster file systems (CFS) are built on top of the global devices and implement standard UNIX® file systems: UFS or HSFS. The implementation of the CFS is such that an application running on top of it will see behavior identical to that encountered on a standalone server with all POSIX semantics being honored. Again, applications do not need to be changed in order to run on top of a cluster file system.

Because cluster file systems are built on top of global devices, they inherit all of the transparent fail/switch-over characteristics that are built into the kernel drivers that implement them. Consequently, a cluster file system will remain mounted on all surviving cluster nodes until the last path to the underlying disk, meta-device, or volume fails. Applications will see only a delay in I/O requests being serviced, while the kernel recovers the pending I/O state and file system consistency under the covers. Likewise, I/O from nodes that are not connected to the primary path for the underlying device will have their I/O request and associated data proxied to and from the primary node via the cluster transport infrastructure.

```
Phys-hardy# df -k
Filesystem            kbytes    used   avail capacity  Mounted on
/dev/dsk/c0t0d0s0    1240727  783041  395650    67%    /
/proc                      0       0       0     0%    /proc
fd                         0       0       0     0%    /dev/fd
mnttab                     0       0       0     0%    /etc/mnttab
swap                  483160     184  482976     1%    /var/run
swap                  483016      40  482976     1%    /tmp
/dev/did/dsk/d1s5      96031    5537   80891     7%    /global/.devices/node@1
/dev/did/dsk/d15s5     96031    5541   80887     7%    /global/.devices/node@2
/dev/md/nfs-set/dsk/d70
                     2031359    2333 1968086     1%    /global/nfs-set
/dev/md/web-set/dsk/d80
                     2031359    2325 1968094     1%    /global/web-set
phys-hardy#
phys-hardy# rlogin phys-floppy
Password:
Last login: Fri May 18 18:20:19 on console
Sun Microsystems Inc.   SunOS 5.8       Generic February 2000
phys-floppy# df -k
Filesystem            kbytes    used   avail capacity  Mounted on
/dev/dsk/c0t0d0s0    1240727  830075  348616    71%    /
/proc                      0       0       0     0%    /proc
fd                         0       0       0     0%    /dev/fd
mnttab                     0       0       0     0%    /etc/mnttab
swap                  487024     128  486896     1%    /var/run
swap                  486928      32  486896     1%    /tmp
/dev/did/dsk/d1s5      96031    5537   80891     7%    /global/.devices/node@1
/dev/did/dsk/d15s5     96031    5541   80887     7%    /global/.devices/node@2
/dev/md/nfs-set/dsk/d70
                     2031359    2333 1968086     1%    /global/nfs-set
/dev/md/web-set/dsk/d80
                     2031359    2325 1968094     1%    /global/web-set
phys-floppy#
```

TABLE 3 Cluster file systems mounted concurrently on two nodes

From an administration perspective, using the global file system is very simple. File systems are created using `newfs` (1M) and mounted using the '`-g`' option to mount (1M).

The cluster file system functionality can be summarized:

- File access location becomes transparent. A process can open a file anywhere in the system, and processes on all nodes can use the same path name to locate that file.
- Coherency protocols are used to preserve the POSIX file access semantics, even if the file is accessed concurrently from multiple nodes.
- Extensive caching and zero-copy bulk I/O are provided to move large data objects efficiently.
- CFS is built on top of the existing Solaris file system at the vnode interface. This interface enables CFS to be implemented without extensive kernel or file system modifications.
- CFS is independent of underlying file system and volume management software. CFS makes any supported on-disk file system global.

The global file service provides the following user benefits compared to Sun Cluster 2.2 software:

- Uses existing UNIX command set, e.g. `mount`, `newfs`, and `ufsdump`, thus minimizing system administrator training.
- Provides continuous access to data, even when failures occur. Applications do not detect failures as long as a path to disks is still available. This guarantee is maintained for raw disk access and file system operations.
- Provides a new class of application implementation where the application and its storage are no longer collocated. This leads to additional recoverable failure scenarios that were inconceivable under Sun Cluster 2.2 software.

Data services

The term *data service* is used to denote an application which runs on a cluster and has been made highly available via a collection of scripts and programs providing start, stop, and monitoring capabilities. The terminology used between the Sun Cluster 2.2 and 3.0 products is quite different, due to the inherently different capabilities and design philosophies of the two products.

Both products are capable of providing basic failover capabilities for crash tolerant applications; however, Sun Cluster 3.0 software allows a new type of scalable service to be deployed. This relies on the new global networking and file system functionality that was introduced with Sun Cluster 3.0 software.

The following table provides a mapping between the terminology used by each product:

Sun Cluster 2.2 Product	***Sun Cluster 3.0 Product***
Data service, such as Oracle or NFS	Resource type
Data service instance	Resource
Logical host	Resource group, but does not contain any disksets
Disksets, managed as part of the logical host	Device group, managed as a separate resource

Sun Cluster 2.2 logical hosts

Logical hosts are the basis for all highly available services with Sun Cluster 2.2 software, with the exception of Oracle Parallel Server. A logical host contains one or more disk-groups or disksets, an IP address per subnet, and one or more applications such as Oracle, NFS, or a web server, and is the smallest unit of service migration between nodes. Client users will access the service via the IP address configured with the service, as opposed to the physical address of the hosting node. This ensures transparency of service location to the user.

Sun Cluster 2.2 software does not provide an strict service dependency mechanism, although this can be achieve by registering the services in a specific order.

Data services are started, stopped, and monitored via a collection of programs and shell scripts registered with the cluster framework using the `hareg` (1M) command. These will be called using a call-back mechanism when the cluster needs to undergo a reconfiguration.

Management of specific instances of the data services rely on a number of disparate commands (`haoracle` (1M), `hasybase` (1M), `hadsconfig` (1M)), rather than the uniform `scrgadm` (1M) interface found in Sun Cluster 3.0 software.

Sun Cluster 3.0 resource group architecture

Sun Cluster 3.0 software takes a more object-oriented approach to the creation of the components needed to build highly available and scalable services. The three main constructs are *resource groups*, *resources*, and *resource types*.

When Sun or third party cluster agents, such as Oracle or NFS, are added to a cluster, the installer is actually adding one or more resource types for that application to the system. A resource type is a collection of programs, shell scripts, and configuration files that provide, as a minimum, the templates for starting, stopping, and monitoring that application. By default, Sun Cluster 3.0 software ships with three resource types:

- SUNW.logicalHostname for handling IP addresses for failover services.
- SUNW.SharedAddress for handling global IP addresses for scalable services.
- SUNW.HAStorage is used to synchronize the start-up of resources and disk device groups upon which the resources depend. It ensures that before a data service starts, the paths to the cluster file system mount points, global devices, and device group names are available.

Any additional resource types have to be registered with the cluster framework before use.

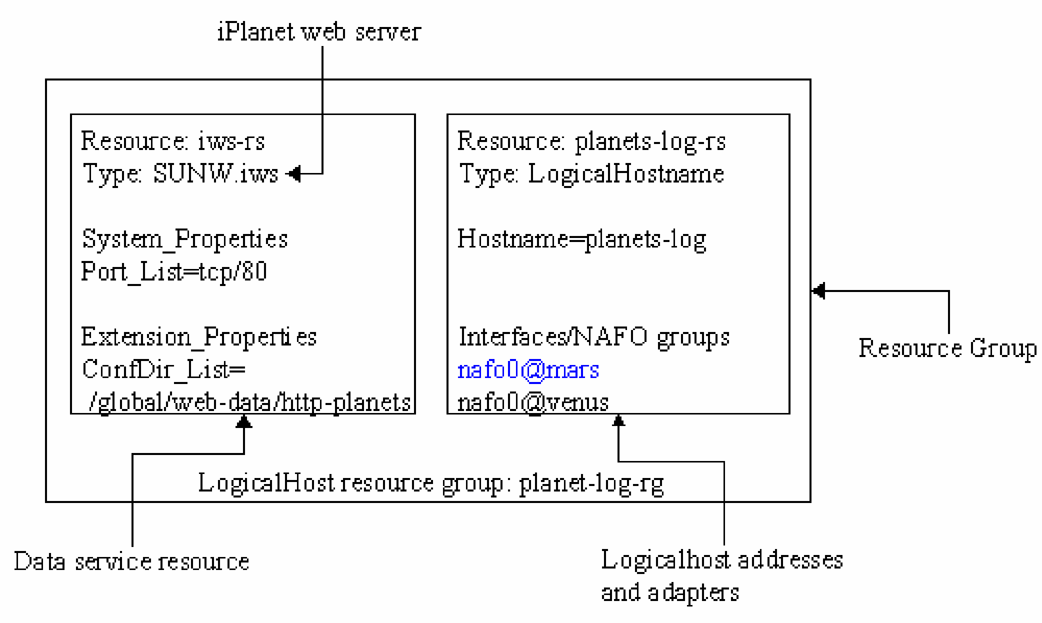

FIGURE 3 Example of a failover resource group configuration

Resources are instances of resource types and inherit all the methods of the resource type registered with the cluster framework. A resource will provide specific settings for various properties that the resource type requires, such as path names to application configuration files. There can be multiple resources of a particular resource type within the cluster without needing to modify the original shell scripts. This was not always true of Sun Cluster 2.2 software, especially where custom data services were concerned.

Resource groups form the logical container to hold one or more resources. The cluster then manages the location of applications by starting or stopping the relevant resource groups on one or more nodes, as appropriate. This task is performed by the resource group manager daemon (`rgmd`).

Sun Cluster 3.0 software provides a stronger resource and resource group dependency model. When a resource, say rsA, needs other resources, say rsB, rsC, and rsD, to be online before it can start successfully, the resource_dependencies property for rsA can be set to ensure these relationships are fulfilled. When the dependency is weaker, the resource_dependencies_weak property ensures that the start method of these resources are called before that of the dependent resource, i.e. rsB, rsC,rsD, and then rsA. However, in this case, there is no requirement for the start method to complete before the start method of rsA is called.

Resource groups also have a dependency property, RG_dependency. This resource groups property indicates a preferred ordering for bringing other groups online or offline on the same node. It has no effect if the groups are brought online on different nodes.

Resource groups have a number of standard and extension properties that allow administrators fine grain control data services on a service-by-service basis. These can be changed while the cluster is running to enable customers to manage the load on the cluster nodes.

Failover data services

A failover data service is created from a resource group as previously described. Typically, a resource group will contain an IP address resource, in turn constructed from the SUNW.logicalHostname resource type, and one or more application components. An NFS service, for example, would include an NFS resource created from the SUNW.nfs resource type.

The distinguishing mark for a failover service is that the specific instance of the application defined can only be run on one node concurrently. This is enforced by two resource group properties: maximum_primaries and desired_primaries, both of which are set to one. This does not mean that more than one instance of this type of service can be run, rather they act on different data sets. Two Oracle databases can be used as an example.

Scalable data services

Scalable services are a new feature to Sun Cluster 3.0 software and have no equivalent in 2.2. Suitable resources in a resource group can be brought on-line on multiple nodes simultaneously, and communicate with the network via the global network service that hosts the particular IP addresses required. In this case, maximum_primaries and desired_primaries will both be greater than one.

Both iPlanet web server (SUNW.iws) and Apache (SUNW.apache) are capable of being defined as scalable services. This list is set to grow as Sun Cluster develops.

HA-API and the SunPlex Agent Builder

Both Sun Cluster 2.2 and 3.0 products have an API for developing application agents. Note that there is no compatibility between them, as the products are radically different. In the 3.0 release, the Resource Management API (RM-API or SUNWscdev package) provides low level C and callable shell script interfaces to basic data service operations. Accompanying this is the higher level Data Service development library (DSDL or SUNWscsdk package), which provides a library for accessing information about the cluster. This saves the developer a lot of repetitive and error prone coding.

The SunPlex Agent Builder is a new feature within the Sun Cluster 3.0 software that allows customers, professional service staff, and system integrators to build simple application agents quickly. Accessed via a GUI, the builder outputs either C or ksh routines for the resource type being constructed, making use of the DSDL. This offers a considerable advance over 2.2, where end users were forced to effectively hand-craft agents from scratch each time.

Summary

Sun Cluster 3.0 software provides significant enhancements in functionality, ease of use, and manageability over Sun Cluster 2.2 software.

The global file service provides a framework for a continuously available file system that is present on all cluster nodes concurrently. This simplifies application deployment, by allowing binaries and configuration files to be installed once and managed singly and centrally. Application data files also have the guarantee of having a consistent namespace on every cluster node. This is one of the key features for enabling service level management.

Global devices allow applications and system services to have a homogeneous namespace across the cluster. Management is simplified, because the uniformity of access to these devices frees an administrator from the chore of having to be logged into a particular node to perform operations such as creating a file system.

The global networking service coupled with the global file service creates a new class of scalable application that could not be achieved on Sun Cluster 2.2 software. Incoming IP packets for suitable applications can be load balanced according to one of three administrator selectable policies, without the need for additional hardware or software. Scalable services can not only scale dynamically to meet enterprise workloads, but also provide continuous availability, although at potentially reduced throughput, in the presence of application instance failures, so long as at least one instance remains.

The `scinstall` menu system substantially simplifies the initial installation and addition of subsequent cluster nodes. On-going GUI administration can be carried out through a secure web browser interface that not only performs the majority of cluster management tasks, but also generates the command line equivalent, facilitating scripting of repetitive administration tasks.

Finally, to enable the cluster software and the nodes to be brought into an enterprise management framework, the Sun Cluster 3.0 software is provided with a Sun Management Center 3.0 agent. This allows all the relevant cluster and node information to be accessed from a central administrative point, under a single administrative framework, that integrates with standard enterprise management products.

Sun Cluster 3.0 software is a key strategic technology in Sun's Service Point Architecture, enabling customers to finally "manage the service, not the server."

References

Writing Scalable Services with Sun Cluster 3.0; whitepaper by Peter Lees; 2001, Sun Microsystems, Inc.

Failure Fencing for Sun Cluster 2.2; whitepaper by Geoff Carrier and Paul Mitchell; 2000, Sun Microsystems, Inc.

Sun™ Cluster Environment: Sun Cluster 2.2; Enrique Vargas, Joeseph Bianco, David Deeths; April 2001, Prentice-Hall, ISBN 0130418706

Author’s Bio: Tim Read

Tim Read is a Lead Consultant for the High End Systems Group in Sun UK Joint Technology Organization. Since 1985, he has worked in the UK computer industry, joining Sun in 1990. He holds a BSc in Physics with Astrophysics from Birmingham University. As part of his undergraduate studies, Tim studied clusters of suns; now he teaches and writes about Sun clusters.

Author’s Bio: Don Vance

Don Vance has worked in the IT industry for 9 years. During that period, he has worked in Telecoms for 5 years, and has worked in the Sun Reseller area for the past 4 years. Don has a BSc in Electrical and Elecronic Engineering from Napier University. While writing this paper, he worked for Horizon Open Systems, and has recently joined Compelsolve as a Sun Pre-Sales Consultant.