



Very Many Database (VMDB) Clusters with IBM eServer BladeCenter and Oracle10g

A Comprehensive Analysis of an Implementation of the Flexible Database Cluster Architecture

Phil Horwitz and Martha Centeno

IBM eServer xSeries Performance Development & Analysis
Research Triangle Park, NC USA

Kevin Closson

PolyServe, Inc.
Beaverton, OR USA

Abstract

The datacenter trend towards server consolidation is occurring at a rapid pace. A recent IT industry survey reported that 89% of respondents are in the process of consolidating servers or are planning to do so. The motivating factors behind server consolidation include agility, availability, security and regulatory compliance. Moreover, TCO is improved by server consolidation since IT personnel can manage consolidated servers more easily and effectively than a large collection of dedicated-purpose systems. But risk avoidance is important since applications don't always coexist. Choosing an architecture for consolidation that offers manageability, flexibility and application isolation is essential. IBM eServer™ BladeCenter® clustered systems running Linux® with PolyServe Matrix Server is a powerful and effective architecture for consolidation. This paper presents a Proof of Concept that demonstrates the consolidation of 60 Oracle10g databases into a 14-node cluster focusing on manageability, performance and availability.

1. Flexible Database Clusters for Large Scale Consolidation	3
Consolidation at a Glance	3
The Flexible Database Cluster Advantage	4
Flexible Database Clusters with Oracle10g – At a Glance.....	4
2. Infrastructure for a Flexible Database Cluster	8
Matrix Server	8
Matrix Server Cluster Filesystem.....	10
Matrix Server Cluster Volume Manager.....	11
Matrix Server Oracle Disk Manager	13
PolyServe MxDB-Oracle-HiAv Solution Pack.....	16
3. Proof of Concept	23
Overview of the IBM eServer BladeCenter.....	23
Managing the BladeCenter.....	25
Storage Subsystem	34
Test Description	37
4. Measurement Results	40
Microbenchmark Analysis.....	40
Oracle Parallel Query Tests	44
High Availability and Manageability	47
5. Summary	59
6. Appendix	60
ioDSS.sql.....	60

Flexible Database Clusters for Large Scale Consolidation

The concept of a Flexible Database Cluster is based on building a cluster that is large enough to support several databases. The goal of this proof of concept was to demonstrate that it makes sense to consolidate large numbers of “pair-wise” failover clusters into large, manageable Flexible Database Clusters.

Consolidation at a Glance

For many years, large SMP systems have supported more than one Oracle database in what is routinely referred to as *Server Consolidation*—concentrating the workloads from several small SMP servers into one large SMP system. Server Consolidation, a trend that started in the late 1990s, was seen as a way to increase efficiency in the datacenter by reducing administrative overhead.

If consolidation adds value in an SMP environment, why wouldn't it add value in a cluster environment? The answer is not simply that it does add value, but, in fact, that consolidation delivers more value in clustered environments than in SMP environments, particularly when supporting Oracle10g fail-over clusters.

A report provided by Meta Group¹ in 2002 makes the point that the “Hard Partitions” technology has been recognized as valuable in consolidation. Hard Partitions offer fixed system provisioning to given applications within a single SMP system. With these systems, administrators can dedicate, for example, 16 processors to application A and another 16 processors to application B.

The report also focuses on the value of dynamic partitioning (e.g., LPAR, VPAR) and suggests that it is critical for supporting consolidation; that is, flexibility is key. These SMP-oriented Server Consolidation systems support the ability to dedicate CPUs to important workloads. Likewise, less important workloads (e.g., untested applications still under development) are cordoned off to other CPUs to limit any disturbance they might cause to other applications. Arguably, clusters do this even better than SMPs with domains or partitions. After all, with SMP systems, there are usually many system components such as memory controllers and I/O adapters that are shared among partitions or domains. With clusters, however, an application running on nodes 1 and 2 does not share server-level resources with applications on nodes 3 and 4. The SAN is shared, but partitionable.

Server Consolidation is a good thing and, because of architectural differences, clusters perform even better in some ways than do single SMP systems. Why then are so many IT organizations supporting a small, dedicated cluster for each application? If several different clusters all support Oracle10g-based applications, why not consolidate?

The Meta Group report of 2002, along with many others, further credit Server Consolidation as enabling IT savings through reductions in the cost of support and skills, capital equipment, and system management. Fewer systems require less management.

If consolidating simple, single servers into one large server yields IT savings, how much more so does consolidating complex clusters into one large cluster?

¹ See the Meta Group report on ZDNet at: <http://techupdate.zdnet.com/techupdate/stories/main/0,14179,2878371-1,00.html>

The Flexible Database Cluster Advantage

The Flexible Database Cluster concept lowers administrative overhead and offers higher availability and on-demand scalability beyond that of several small clusters. It is an architecture well worth considering. A Flexible Database Cluster is more than just a big cluster. The prime ingredients include:

- **Manageable Hardware.** IBM® eServer® BladeCenter® and IBM TotalStorage® DS4500 intelligent storage array
- **Systems Software.** PolyServe Matrix Server Cluster Volume Management, Cluster Filesystem and Integrated High Availability Engine
- **Deployment Methodology.** PolyServe MxDB-Oracle-HiAv Solution Pack

The synergy of IBM® eServer® BladeCenter®, PolyServe Matrix Server, and Oracle10g makes the Flexible Database Cluster (FDC) a powerful platform for supporting multiple applications. This white paper describes a proof-of-concept that validates the architecture and technology of the Flexible Database Cluster for consolidation and confirms that:

- PolyServe Matrix Server and Oracle10g perform extremely well on the IBM eServer BladeCenter platform.
- The BladeCenter architecture and technology help provide an unparalleled high-availability platform for implementing Flexible Database Clusters.
- IBM and PolyServe are leading the development of clusters consolidation.
- The architecture and technology of the Flexible Database Cluster help enable on-demand computing. Cluster nodes provide a pool of flexible resources for use among applications. The availability of Oracle10g is enhanced because nodes can be dynamically provisioned using Matrix Server to cover the loss of another node.
- The Flexible Database Cluster provides strong management tools such as Matrix Server for performance and availability. A single large cluster is now easier to manage than many small clusters.
- A general-purpose cluster filesystem such as the one included with Matrix Server provides a single-system feel and greatly enhances manageability. A shared Oracle home used by all nodes also simplifies management. Support is available for all database operations that require a filesystem.
- A specialized Cluster Volume Manager such as the one included with Matrix Server offers improved manageability and performance.
- Improved manageability, scalability, expandability, availability and asset utilization in an FDC configuration also can help dramatically lower total cost of ownership (TCO) relative to a UNIX®-based IT environment.

Flexible Database Clusters with Oracle10g—At a Glance

The thought of assembling a 14-node cluster, much less managing one, conjures up visions of cabling and OS configuration nightmares. This image is likely rooted in the UNIX-based clustered systems of the 1990s. Clusters of that era were configured with very few nodes primarily due to limitations in Oracle Parallel Server. The economic benefit of combining powerful, commodity processor-based servers running Linux in a large cluster is well known. The question is "Why make a large database cluster, and what are the management considerations?"

The answer hinges on the technology that is being assembled into the cluster. The base requirement for an Oracle10g High Availability (failover) cluster is a set of servers with shared disk (e.g., JBOD) access and LAN/interconnects connectivity. Strictly speaking, nothing more is required. For reasons explored throughout this paper, it is unlikely that any such "bare-bones" clusters will be manageable when there are many Oracle10g databases deployed. Nor are such clusters configured to offer the management and flexibility attributes of the Flexible Database

Cluster (FDC) model studied during this project. Finally, simple clusters are not integrated sufficiently to perform the required service monitor probing that is needed to ensure availability.

Concerns over building and maintaining large clusters for Oracle10g High Availability generally fall into six categories, although not necessarily in this order:

- Storage configuration and management
- Storage connectivity and configuration
- OS configuration and management
- Oracle product installation, configuration and maintenance
- Database file locations and space management
- Operational methodology

Storage Configuration and Management

Today, building large clusters such as the Flexible Database Cluster in this proof of concept is actually quite simple, due in part to Fibre Channel SAN technology. Even simpler are large clusters with advanced commodity processor-based clusters such as the IBM eServer BladeCenter.

Storage management is also much simpler and more powerful with today's technology. A great example is the IBM TotalStorage® DS4500 intelligent storage array.

Storage Connectivity and Configuration

Modern technology is also making it easier to connect and configure SAN switches. In the FDC proof-of-concept system, the switch was easily configured using the IBM TotalStorage SAN Fibre Channel Switch Specialist management tool. The switch features an embedded Web browser interface for configuration, management, and troubleshooting. The BladeCenter integrates Fibre Channel switch hardware packaged to nearly eliminate cabling.

OS Configuration and Management

Configuring Linux to support a large cluster is much simpler than legacy UNIX. Enhanced configuration tool kits are available with Linux distributions such as SUSE Linux Enterprise Server (SLES9), which provides the KDE Konsole² utility. This utility feeds key input to all or some nodes of a cluster, which is useful for redundant administrative tasks. This is but one example of the added value that SUSE SLES9 offers. Every small improvement that makes a cluster feel more like a single system is critically important, even more so in the case of the Flexible Database Cluster.

Oracle Product Installation, Configuration and Maintenance

On traditional failover clusters, the Oracle10g software must be installed on each node where instances will be executed. Installing Oracle is not difficult. However, difficulty arises when Oracle is installed, for example, on 10 sets of pair-wise clusters in support of 10 failover databases.

With a general-purpose cluster filesystem³, it is quite simple to set up a "Shared Oracle Home." The Oracle Universal Installer is instructed to install on only one node (a single-node cluster install). With a shared Oracle Home, all nodes in the cluster can use the same executables. Also, configuration files are located in the cluster filesystem and can be edited from any node in the cluster. In fact, all configuration files for all nodes can be edited from any node in the cluster.

² For information about using KDE Konsole when configuring large clusters for Oracle9i RAC, see this paper on the SUSE website: http://www.SUSE.com/en/business/certifications/certified_software/oracle/docs/9iR2_RAC_sles8_polyserve.pdf

³ Such as the PolyServe Matrix Server CFS used in this test. For more information, visit www.polyserve.com.

\$ORACLE_HOME points to the same directory on all nodes, and that directory is in the cluster filesystem.

Since the PolyServe MxDB-Oracle-HiAv Solution Pack for Oracle10g uses Virtual Hosts as the TNS hostname for services, there is no worry over shared files such as listener.ora. The PolyServe cluster filesystem has always supported Context Dependent Symbolic links, a feature that guarantees the ability to have node-local files.⁴

Database File Locations and Space Management

Managing a large number of Oracle datafiles can become difficult when several databases are housed in one large cluster, such as those in this proof of concept. Since the PolyServe cluster filesystem is optimized for Oracle by supporting both direct and asynchronous I/O, locating many databases in a simple, yet optimized, cluster filesystem directory greatly simplifies large scale consolidation. After all, filesystems have been established as the most sensible place to store Oracle datafiles for more than 20 years, so why switch to more complex, non-filesystem approaches just to support failover? Creating databases in the cluster filesystem is simple. In fact, cluster filesystem is the first option available in Oracle10g Database Configuration Assistant, as shown in Figure 1.1.

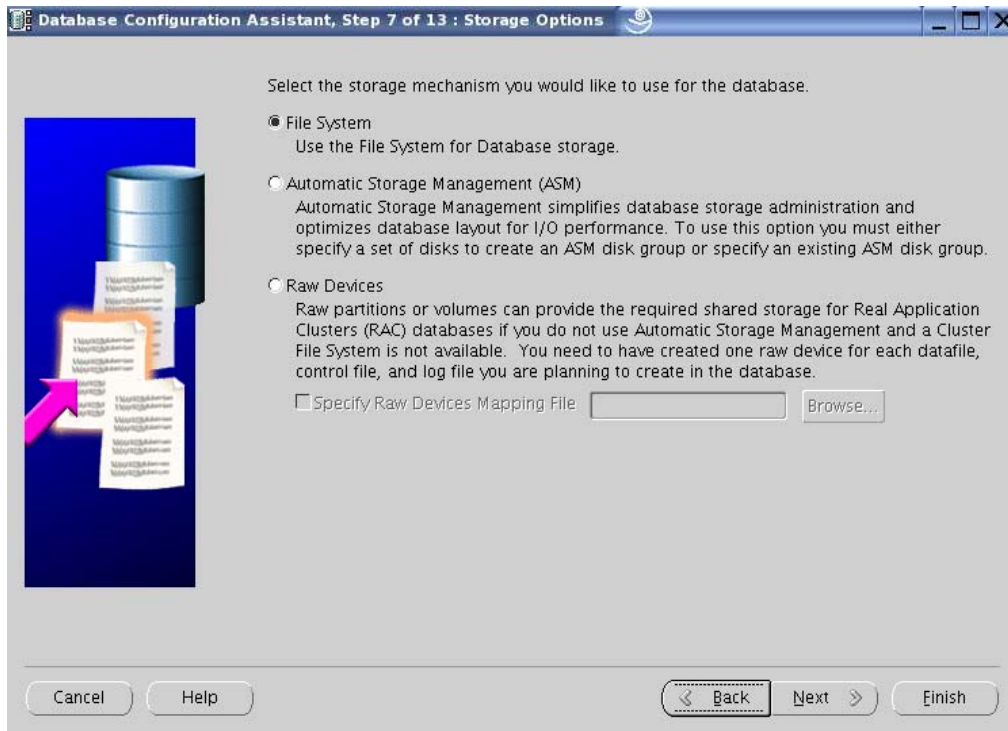


Figure 1.1: Oracle Database Configuration Assistant Storage Options Page

⁴ For more indepth information about context-dependent symbolic links, please see the following IBM Redbook: <http://www.redbooks.ibm.com/Redbooks.nsf/RedbookAbstracts/redp9123.html?Open>

Another significant technology that assists in large-scale, many database clusters consolidation is an Oracle feature, introduced in Oracle9i, known as Oracle Managed Files (OMF). Although commonly misunderstood, OMF has been an Oracle feature since Oracle9i. In fact, OMF files are selectable from the Database Configuration Assistant as Figure 1.2 shows.

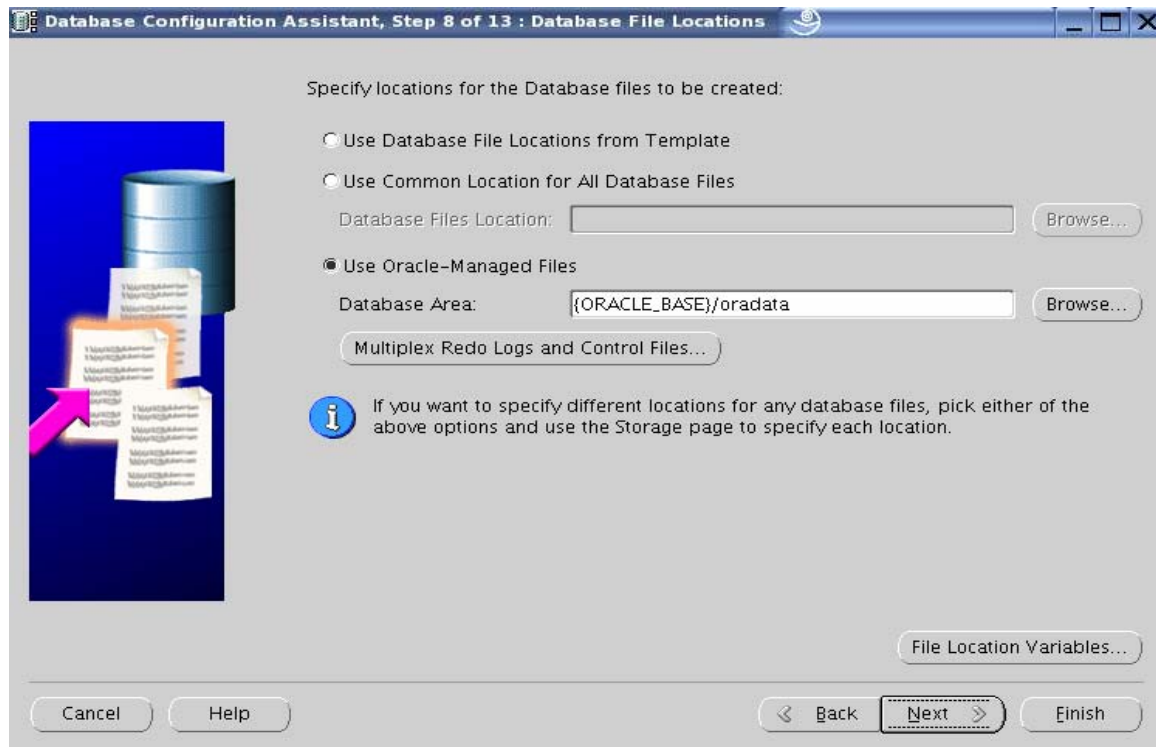


Figure 1.2: Oracle Database Configuration Assistant File Locations Page

Using Oracle Managed Files is a good option in a large FDC because it simplifies administration. OMF⁵ provides the following benefits:

- Simplifies deployment of databases. OMF minimizes the time spent making decisions regarding file structure and naming and reduces file management tasks overall.
- Reduces corruption that can be caused by administrators specifying the wrong file. All OMF files are provided a unique name by Oracle.
- Reduces wasted disk space consumed by obsolete files. OMF removes unused files automatically.

Forcing a Database Administrator (DBA) to think about tablespaces as a collection of files takes time away from actual database administration. The “contents” of the files are the actual database. Databases consist of rows in blocks grouped into extents and tracked as segments. Those objects reside in datafiles. Deploying with OMF reduces the need for physical administration and allows concentration on logical administration—providing the biggest return on database administration efforts. For instance, why be concerned about what sectors of a disk are used by a filesystem? That particular detail is the responsibility of the filesystem. Likewise, DBAs can leave datafile management to OMF. More time can be dedicated to query optimization and other such tasks that may actually improve performance.

Examples showing how the FDC case study used OMF are provided later in this paper.

⁵ For in-depth information about Oracle Managed Files, see the Oracle documentation.

Operational Methodology

Monitoring is a major concern in a large cluster environment. If administrators must execute numerous commands to get the status of many nodes, the value proposition quickly dwindles.

Monitoring cluster operations with the type of technology used in the Flexible Database Cluster proof of concept is vastly superior to the technology of the recent past. Comprehensive I/O monitoring at the storage level is possible through modern storage management software. Oracle10g and Grid Control offer a great deal of instance and global database monitoring. PolyServe Matrix Server offers monitoring of the entire cluster from a central console that can be executed either remotely or on any system in the datacenter. Finally, PolyServe's implementation of the Oracle Disk Manager offers unprecedented Oracle-centric, cluster-aware I/O monitoring.

Beyond monitoring, other issues have traditionally affected cluster operations. For instance, administrators have had to connect to specific nodes in the cluster to perform operations such as configuring files in Oracle Home. Also, requiring administrators to remember what node they used to perform an export or what node has a SQL*Plus report on it can be bothersome in today's hectic IT environment. Administrators can simply change directories to the same filesystem location from any node to perform administrative functions and run scripts.

Concerns such as these have served as roadblocks to deploying the sort of large cluster that can be used as a Flexible Database Cluster.

Infrastructure for a Flexible Database Cluster

This section describes how the Flexible Database Cluster was set up for testing. The core platform software technology for this aspect of the proof of concept was the PolyServe Matrix Server product with the MxDB-Oracle-HiAv Solution Pack. Four essential Matrix Server products were germane to this testing:

- PolyServe Cluster Filesystem
- PolyServe Cluster Volume Manager
- PolyServe MxODM Library
- MxDB-Oracle-HiAv Solution Pack

Matrix Server

Matrix Server is more than a cluster filesystem; it also offers a scalable cluster volume manager, high-availability framework and SAN management. Furthermore, Matrix Server offers multi-path I/O, which is a pivotal component in reducing and eliminating single points of failure.

Figure 2.1 shows the Matrix Server Management Console, which provides a bird's-eye view of cluster status. The console can be used with simple point and click to set up volumes, filesystems, mount options and advanced Hi-Av process and service monitoring for failover. A command-line interface is also available for cluster configuration.

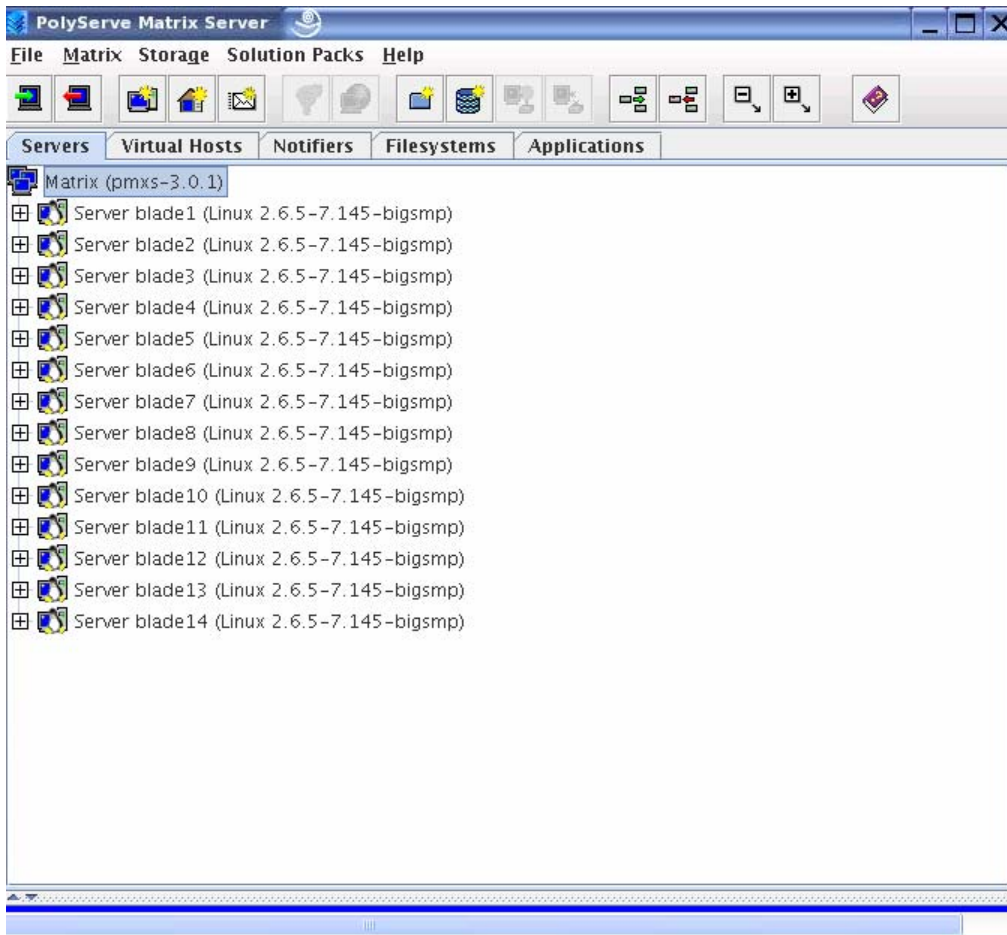


Figure 2.1: PolyServe Management Console

Figure 2.2 provides another view of the Management Console, showing its drill-down capability for obtaining the status of MxDB-Oracle-HiAv Solution Pack Virtual Servers for Oracle10g.

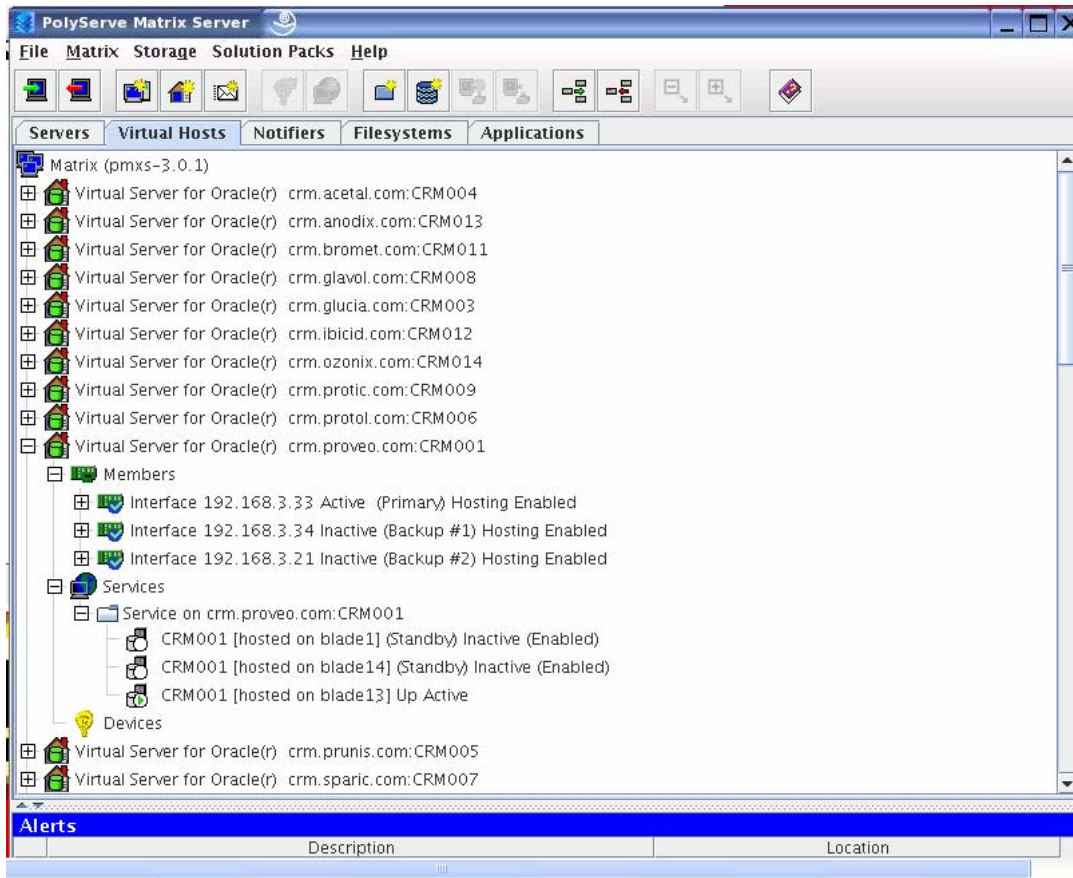


Figure 2.2: PolyServe Matrix Server Console with Virtual Services for Oracle10g

Matrix Server Cluster Filesystem

The Matrix Server cluster filesystem is both general-purpose and optimized for Oracle. These attributes proved quite valuable in the following ways.

General-Purpose

- A single Shared Oracle Home was configured for all 14 nodes.
As described earlier in this paper, a general-purpose cluster filesystem such as the Matrix Server CFS supports setting up a single directory for Oracle Home. This functionality is key to the Flexible Database Cluster architecture.
- Archived redo logging was performed in a cluster filesystem location and compressed.
- Some of the data was loaded with External Tables which were located in the CFS.

Optimized for Oracle

- All datafiles, control files, online logs, and so on were located in filesystems mounted with the Matrix Server "DBOPTIMIZED" mount option, which implements Direct I/O.

- Oracle Disk Manager⁶ (ODM) was used for asynchronous I/O and improved clusterwide I/O monitoring.

Matrix Server Cluster Volume Manager

The filesystem created to hold all of the database files was placed on a PolyServe Matrix Server cluster volume. The volume was a striped concatenation of four LUNs presented by the IBM DS4500 array with total capacity of 657GB. Each LUN was comprised of 20 physical disks so the total physical disk count for the database files was 80. The LUNs were configured as RAID 1+0 within the array. The PolyServe software then concatenated the LUNs and striped them on a 256KB stripe width (stripe width is fully configurable). Figure 2.3 shows a `df(1)` listing from the FDC. Note the large filesystem contained in `/dev/psv/psv1` and mounted on `/u02`—a PolyServe Logical Volume.

```

$ df
Filesystem          1K-blocks      Used Available  Use% Mounted on
/dev/hda2           36918868    7037020  29881848   20% /
tmpfs               2075476         4   2075472    1% /dev/shm
/dev/hda1            99043        10716   83213    12% /boot
/dev/psd/psd1p2     34413436    9800296  24613140   29% /u01
/dev/psv/psv1      689915696  252145192 437770504   37% /u02
$

```

Figure 2.3: PolyServe Cluster Filesystems space listing with `df(1)`.

⁶ For in-depth information regarding Oracle Disk Manager, see the white paper on the Oracle Technology Network: http://otn.oracle.com/deploy/availability/pdf/odm_wp.pdf.

Configuring cluster volumes is very simple using the PolyServe Management Console. The available disks are presented as shown in Figure 2.4. The administrator simply chooses which disks the volume will be comprised of and specifies the internal label and stripe width.

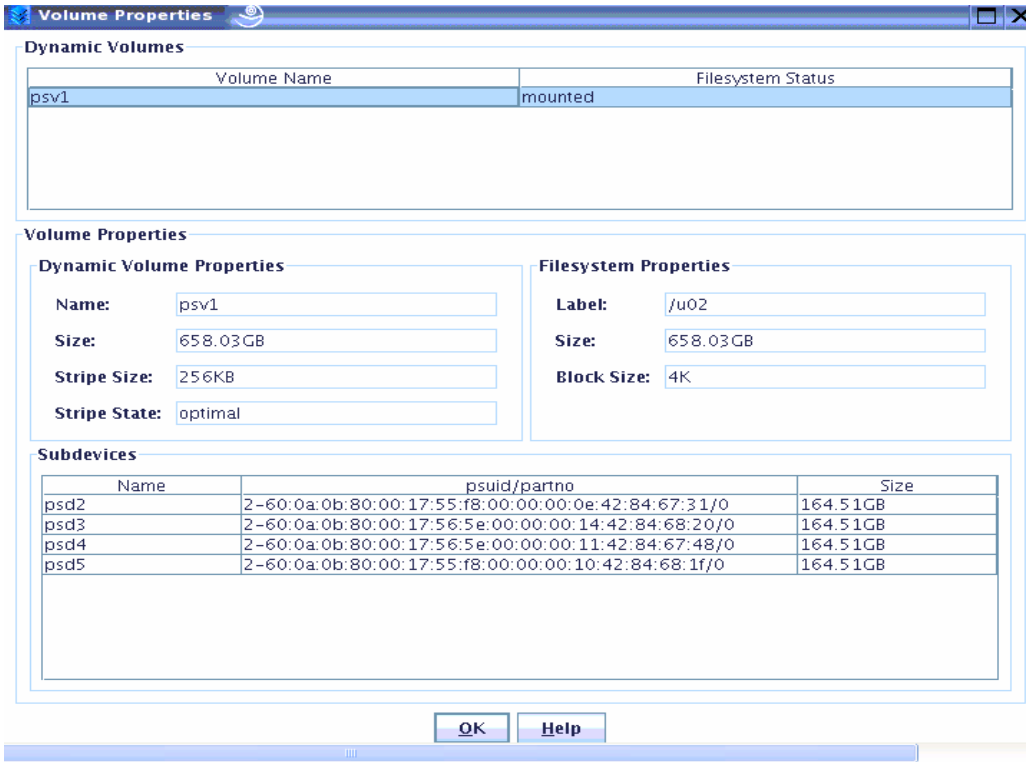


Figure 2.4: PolyServe Cluster Volume Manager Properties Dialog

Once the volume is created, the Management Console is used to create and mount a filesystem. The Management Console Filesystems tab provides drill-down information for all filesystems in the cluster as Figure 2.5 shows.

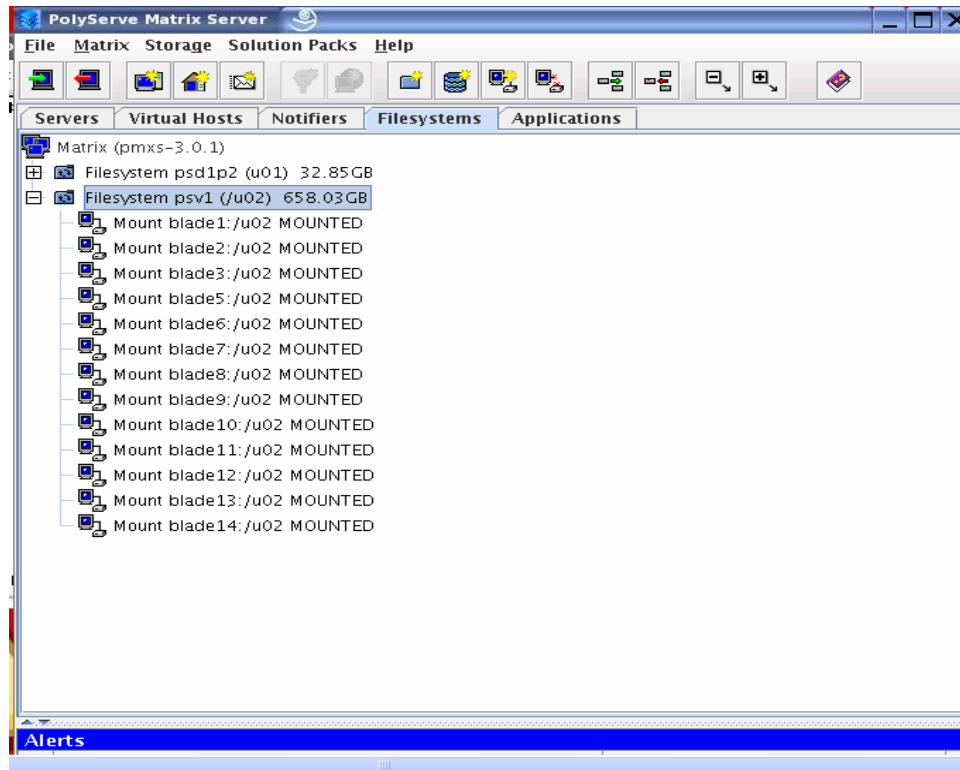


Figure 2.5: PolyServe Cluster File System mount details

Matrix Server Oracle Disk Manager

PolyServe Matrix Server provides an ODM library implementation called MxODM to support the Oracle Disk Manager interface. Although MxODM offers improved datafile integrity through clusterwide file keys for access, its main benefit in the FDC architecture is improved manageability through enhanced performance monitoring. MxODM also enables Oracle10g with asynchronous I/O.

MxODM for Performance Monitoring

The MxODM I/O statistics package provides the following basic I/O performance information. These reported items are referred to as the Core Reporting Elements.

- Number of file Read and Write operations
- Read and Write throughput per second in Kilobytes
- Count of synchronous and asynchronous I/O operations
- I/O service times
- Percentages

The Core Reporting Elements can be provided at the following levels:

- Clusterwide Level. Provides aggregate information for all database instances on all nodes.
- Database Global Level. Limits information to a named database (e.g., OE001,FIN008,MRP010).

- Instance Level. Limits information to a named instance—a feature that is more relevant in a RAC configuration.
- Node Level. Limits information to a named node (e.g., blade1.ibm.com, blade8.ibm.com). This information is the aggregate of all instance activity on the named node. If a node hosts instances accessing different databases (e.g., \$ORACLE_SID=OE002, \$ORACLE_SID=MRP002), the Core Reporting Elements will reflect the combined information for all instances on the named node.

Since MxODM is linked in with the Oracle Server and understands Oracle file, process, and I/O types, the **mxodmstat(8)** command offers very specialized reporting capabilities. On complex clustered systems, it is nearly impossible to take a quick look at the clusterwide or per-instance activity for a given subsystem of the Oracle Server. For example, on a 14-node cluster with 15 Order Entry instances and Parallel Query Slaves (PQO) active on nodes 7 through 10 on the CRM databases, a DBA will find it extremely difficult to associate clusterwide impact to the PQO activity. Likewise, quickly determining the DBWR activity for only the FIN instances on nodes 1 through 6 is nearly impossible—without MxODM. These are just the sorts of puzzles that are easily solved with the FDC architecture. Similar examples are provided later in this paper.

MxODM offers “canned” reporting that focuses on the following key Oracle “subsystems.”

Parallel Query Option (PQO). This query returns the Core Reporting Elements for only the Parallel Query Option slaves (e.g., ora_p000_OE001, ora_p001_MRP001, etc.). This is an extremely beneficial set of information because it allows DBAs to get a top-level view of the impact PQO is having on the cluster, either as a whole or at the node level.

Log Writer. This query focuses on only the **lgwr** processes and their activity at the cluster level, database level, or node level. Because all of the Core Reporting Elements can be returned in this query, it is beneficial for DBAs to maintain streaming output of this query showing **lgwr** activity at either the cluster level or broken down by database, instance, or node.

Database Writer. This query is of the utmost value. It too can return all Core Reporting Elements at all Reporting Levels; however, it can also limit reporting to only **dbwr** process activity. DBAs can glance at **mxodmstat(8)** output and easily determine the average **dbwr** I/O service times for all databases clusterwide, or can focus on specific databases, nodes, or instances.

MxODM for Diagnosis

Time for a parlor game. You have to think on your feet; you are the System Administrator. Quick, identify all the files open by Oracle Instances throughout the entire cluster. Ok, go!

The **mxodmstat(8)** command offers system administrators significant information about Oracle file usage throughout the entire cluster—from one central command execution. Consider 60 databases spread out across 14 nodes (as was the case in this Very Many Databases proof of concept). How many instances does the Oracle DBA have to connect to, or how many Grid Control tabs must be navigated to locate all of the file paths? Such information is available with a simple, single command using PolyServe. Figure 2.6 shows an active file dump issued from blade1. The **grep(1)** command is used to focus files used by Oracle SID OE001.

```

192.168.3.21 - PuTTY
$ mxodmstat -f | grep OE001
3637 CTL /u01/app/oracle/product/oradata/OE001/controlfile/o1_mf_ZOeae10yT1pK0_.ctl
3638 CTL /u01/app/oracle/product/oradata/flash_recovery_area/OE001/controlfile/o1_mf_ZOeae1w452pK1_.ctl
3640 DATA /u01/app/oracle/product/oradata/OE001/datafile/o1_mf_system_gOeae1OpA3hr0_.dbf
3641 DATA /u01/app/oracle/product/oradata/OE001/datafile/o1_mf_undotbs1_gOeae1UjP3hr2_.dbf
3642 DATA /u01/app/oracle/product/oradata/OE001/datafile/o1_mf_sysaux_gOeae1U0K3hr1_.dbf
3643 DATA /u01/app/oracle/product/oradata/OE001/datafile/o1_mf_users_gOeae1GXR3hr3_.dbf
3644 DATA /u01/app/oracle/product/oradata/OE001/datafile/o1_mf_indx_mFfae1TIp1tP20_.dbf
3645 DATA /u01/app/oracle/product/oradata/OE001/datafile/o1_mf_data_wFfae1Rh52tP21_.dbf
3646 DATA /u01/app/oracle/product/oradata/OE001/datafile/o1_mf_users_1kbf111g_.dbf
3647 DATA /u02/oradata/OE001_test_ts/test_ts.dbf
3648 DATA /u02/oradata/OE001_test_ts/datafile_4.dbf
3649 DATA /u02/oradata/OE001_test_ts/datafile_1.dbf
3650 DATA /u02/oradata/OE001_test_ts/datafile_2.dbf
3651 DATA /u02/oradata/OE001_test_ts/datafile_3.dbf
3652 DATA /u01/app/oracle/product/oradata/OE001/datafile/o1_mf_temp_2Peae1MHp1pK8_.tmp
3667 OLG /u01/app/oracle/product/oradata/OE001/onlinelog/o1_mf_1_ZOeae1RRC2pK2_.log
3668 OLG /u01/app/oracle/product/oradata/flash_recovery_area/OE001/onlinelog/o1_mf_1_ZOeae16ec3pK3_.log
3671 OLG /u01/app/oracle/product/oradata/OE001/onlinelog/o1_mf_2_ZOeae1lhM3pK4_.log
3672 OLG /u01/app/oracle/product/oradata/flash_recovery_area/OE001/onlinelog/o1_mf_2_OPeae1zN8pK5_.log
3675 OLG /u01/app/oracle/product/oradata/OE001/onlinelog/o1_mf_3_OPeae1wpp1pK6_.log
3676 OLG /u01/app/oracle/product/oradata/flash_recovery_area/OE001/onlinelog/o1_mf_3_OPeae1juY1pK7_.log
$ ls -l /u02/oradata/OE001_test_ts/datafile_4.dbf
-rw-r--r-- 1 oracle dba 1073750016 2005-09-05 16:51 /u02/oradata/OE001_test_ts/datafile_4.dbf
$

```

Figure 2.6: mxodmstat(8) file usage output for one database

Also available are the PIDs, SIDs and active node of Oracle processes throughout the entire cluster and whether they are background (e.g., dbwr, lgwr, pmon) or foreground processes (e.g., shadow or MTS server processes). This information is available from one central command executed on any node in the cluster as depicted in Figure 2.7.

```

192.168.3.21 - PuTTY
$ mxodmstat -l | grep OE001 | more
blade1 OE001 OE001 -- +Background 2502
blade1 OE001 OE001 -- +Background 6744
blade1 OE001 OE001 -- +Background 26570
blade1 OE001 OE001 -- +Background 26593
blade1 OE001 OE001 -- +Background 26603
blade1 OE001 OE001 -- +Background 26606
blade1 OE001 OE001 -- +Background 27296
blade1 OE001 OE001 -- +DB Writer 26539
blade1 OE001 OE001 -- +Log Writer 26557
blade1 OE001 OE001 -- +PQO 26612
blade1 OE001 OE001 -- +PQO 26614
blade1 OE001 OE001 -- +PQO 26616
$ ps -fp 26557
UID PID PPID C STIME TTY TIME CMD
oracle 26557 1 0 17:38 ? 00:00:00 ora_lgwr_OE001
$

```

Figure 2.7: Detailed Oracle file information from mxodmstat(8)

PolyServe MxDB-Oracle-HiAv Solution Pack

The Database Serving Solution Pack for Oracle HiAv on Linux (MxDB-Oracle-HiAv) provides high availability for database instances and Net Services listener processes.

MxDB-Oracle-HiAv uses a Virtual Oracle Service to provide connectivity to the database. A service monitor is associated with each Virtual Oracle Service. The monitor periodically checks the health of the database instances and listeners via a probe action. The health of the cluster in general is monitored by the core High Availability engine embedded in the Matrix Server product. The Virtual Oracle Service and service monitor are created through the Graphical User Interface supplied with the Solution Pack.

MxDB-Oracle-HiAv takes the following actions in the event that service for a database is lost:

- **Server Failure.** PolyServe Matrix Server monitors the health of the servers in the matrix and provides for application failover if a server becomes unavailable. In the case of MxDB-Oracle-HiAv, if a server executing a monitored database goes down, the database and associated listener will be started on the node configured as the first backup.
- **Database/Listener Service Loss.** If a database or listener under MxDB-Oracle-HiAv control should fail, it will be restarted. Failure is detected by the MxDB-Oracle-HiAv monitor probe.

Configuration

Configuring an existing database with MxDB-Oracle-HiAv is very simple. Any database created with the Database Configuration Assistant (DBCA) is configurable into the HA framework with no more trouble than filling out the simple GUI shown in Figure 2.8. Databases created with scripts can be encapsulated just as easily, provided they comply with a few simple restrictions such as there must be an spfile or init.ora for the database in the `$ORACLE_HOME/dbs` directory.

The Solution Pack will also configure Net Services if desired, or Net Services tools (e.g., netca, metmgr) can certainly be used.

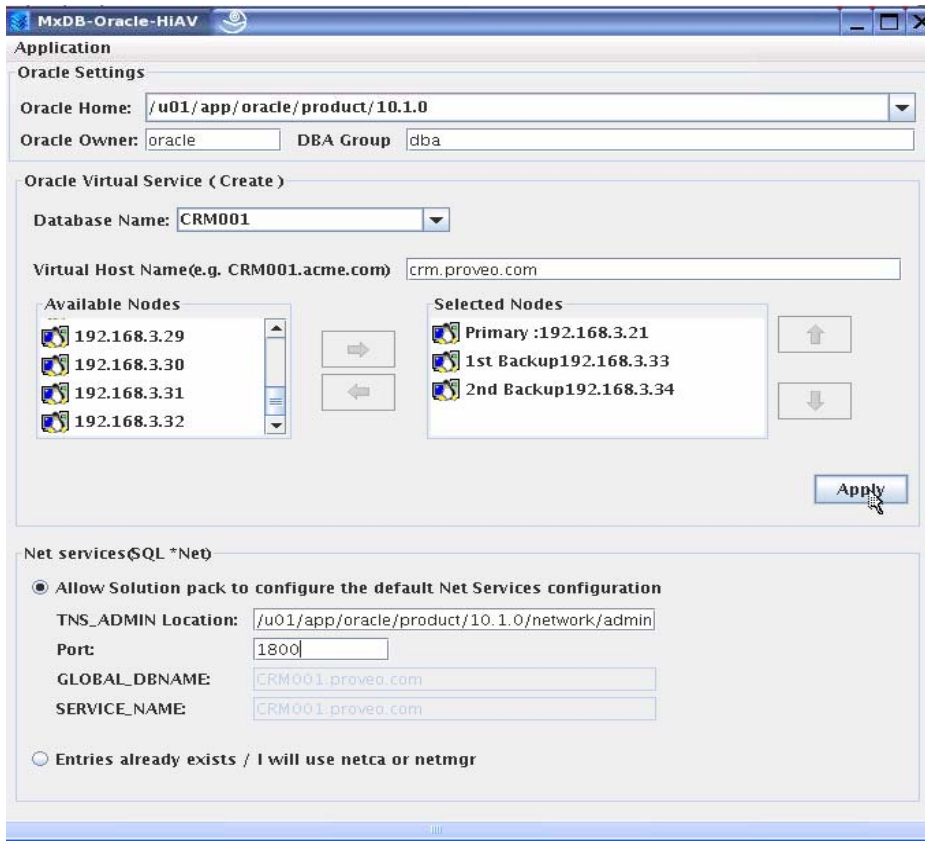


Figure 2.8: MxDB-Oracle-HiAv Configuration and Operations GUI

Operation

Once the database is placed under Solution Pack control, it must now be thought of as a Solution Pack service. Operations such as starting and stopping the database are Service Monitor operations. Indeed, connecting directly to a database being monitored by the Solution Pack and issuing the shutdown command will appear to the service monitor as a database crash. In that case, the Solution Pack will restart the database.

Operations are performed in either the Solution Pack GUI or the CLI. The buttons seen in Figure 2.9 show that a simple mouse click can start and stop a database. A set of mouse clicks can rehost the database from one node to another in the cluster. Notice the drop-down box for Oracle Home. The Solution Pack can manage any number of databases from any number of Oracle Homes of any combination of Oracle9i, Oracle10g Release 1 and Oracle10g Release 2.

Additionally, there is a built-in demo database called PILOT. Creating PILOT before encapsulating any production databases is very helpful for training purposes.

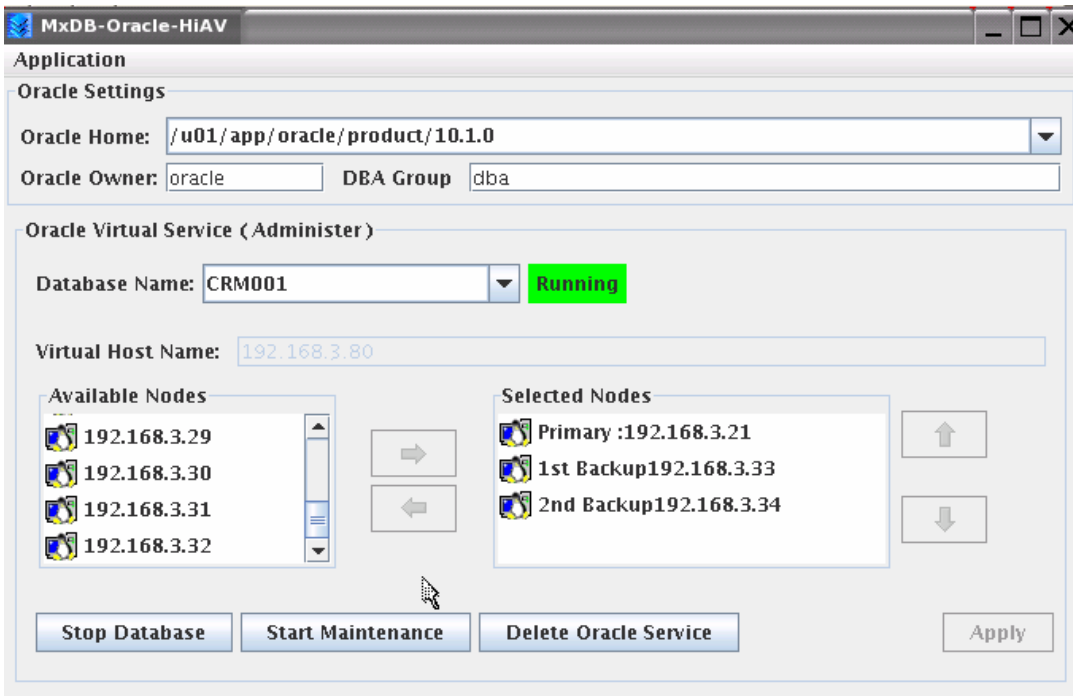


Figure 2.9: MxDB-Oracle-HiAv Virtual Oracle Service Status

Figure 2.10 shows how the DBA can perform operations on any of the databases from one central GUI without disconnecting and reconnecting.

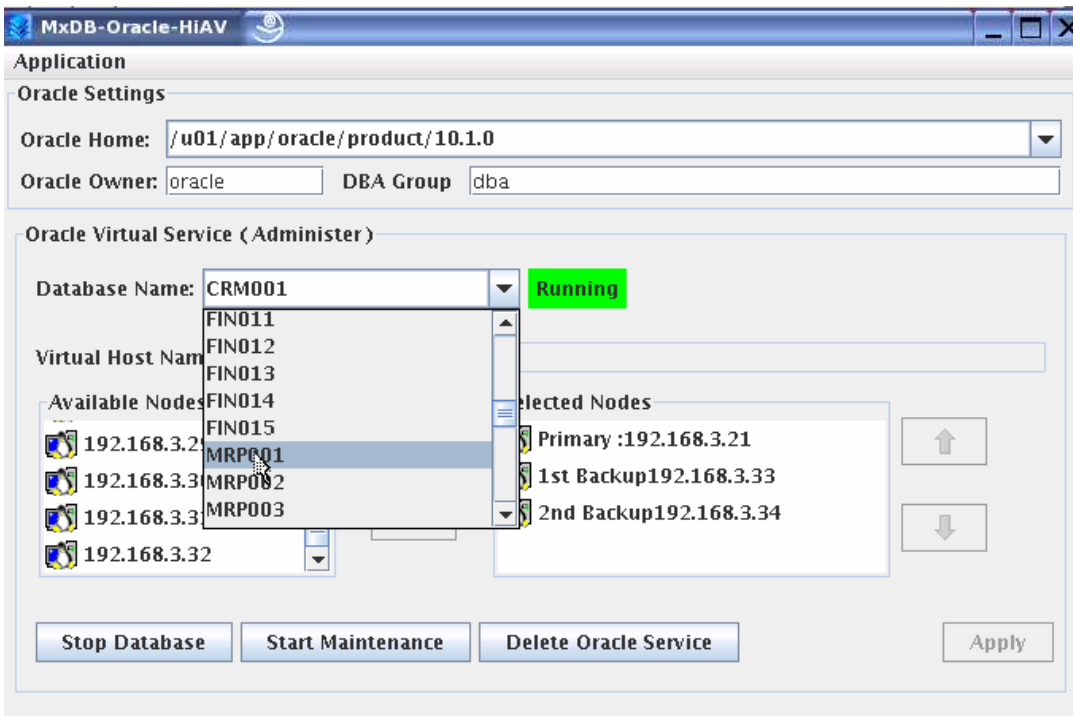


Figure 2.10: Selecting a database on the MxDB-Oracle-HiAv GUI

Figure 2.10 also shows a button for starting and stopping Maintenance Mode. In this mode, the database is still encapsulated by the Solution Pack, but is no longer being probed. This mode is a very convenient way to “free” the database up for multiple startup/shutdown operations when probing is not desirable. Maintenance Mode is also provided to ensure that the Solution Pack does not hinder any third-party operations such as backup.

Roles

With MxDB-Oracle-HiAv, the DBA does not need any superuser privilege for configuration or operation. With the Solution Pack GUI (or CLI), the DBA can configure, start, stop, rehost and transition the service monitor into maintenance mode. This is a significant feature, considering that some competitive products require superuser privileges for the DBA to perform database operations.

If the DBA has the Solution Pack GUI and CLI, what tools are there for the System Administrator? The System Administrator is, after all, responsible for the entire cluster. What if he or she has to perform maintenance on a node with an active MxDB-Oracle-HiAv Virtual Oracle Service running? The Matrix Server Management Console also exposes Solution Packs to the System Administrator. Figure 2.11 shows the Applications tab on the Management Console. Visible are such details as the database name and the nodes that the DBA has chosen for the primary and backup servers.

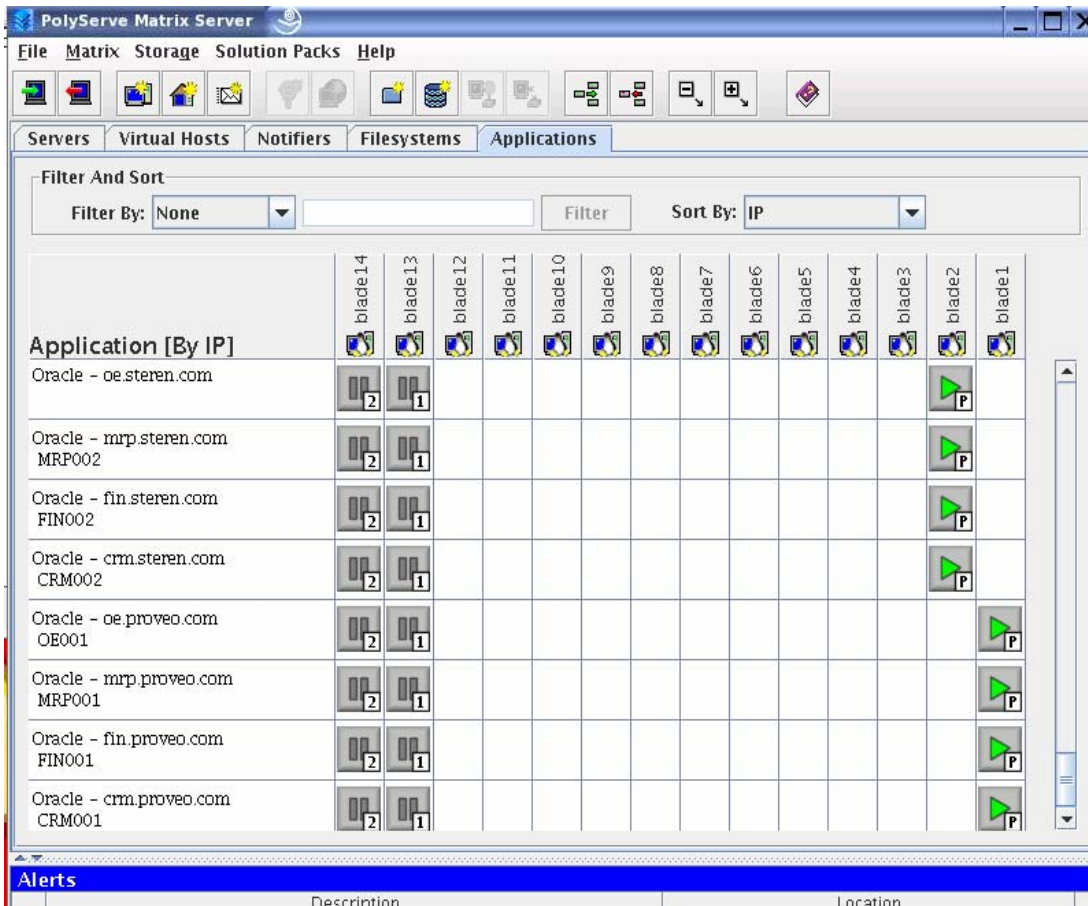


Figure 2.11: PolyServe Management Console Applications view of Virtual Oracle Services

The Management Console can be used for rehosting a database just as easily as the MxDB-Oracle-HiAv GUI and CLI. Figure 2.12 shows the dialog box used for rehosting a database. The example rehosts an MxDB-Oracle-HiAv Virtual Oracle Service called OE.steren.com.

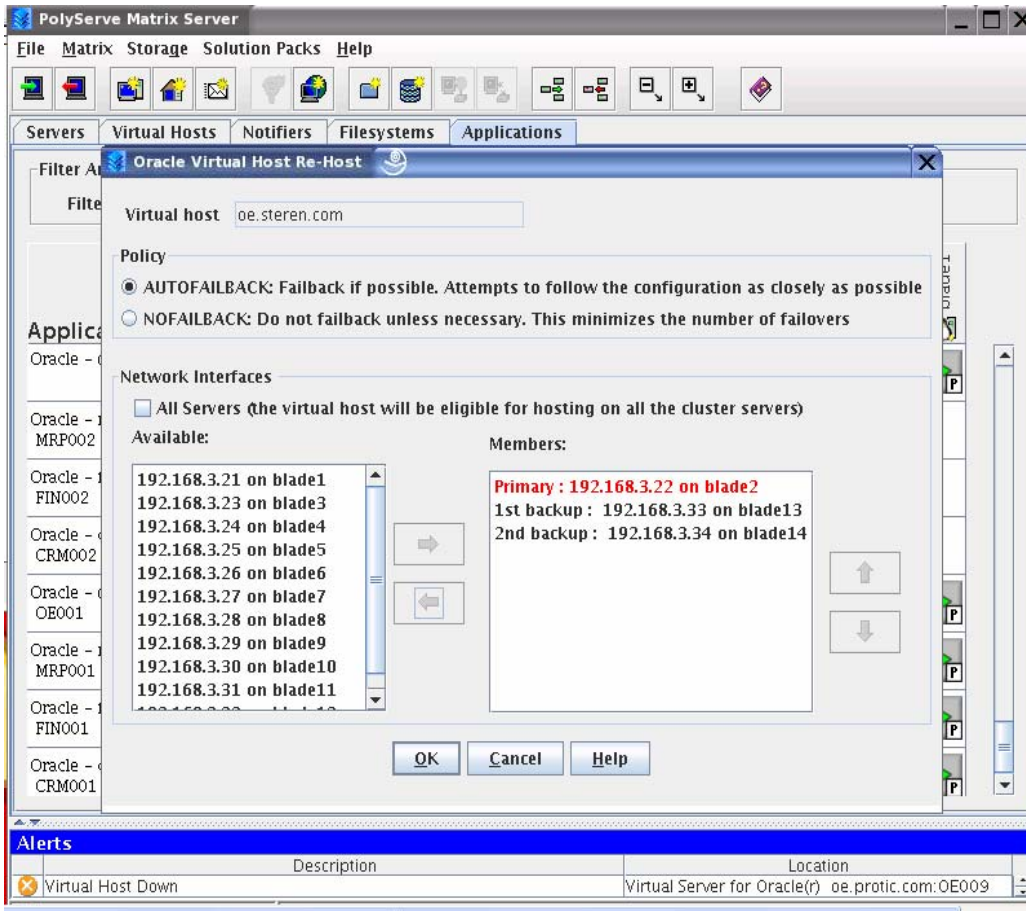
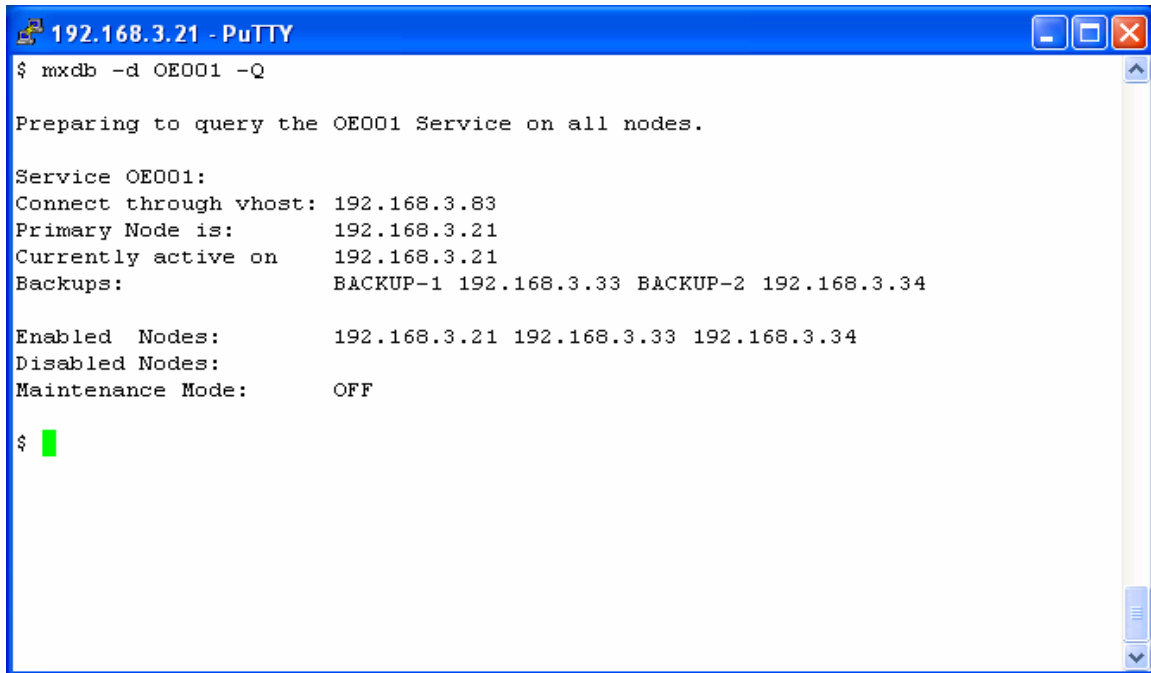


Figure 2.12: Management Console dialog for re-hosting a Virtual Oracle Service

Figure 2.13 shows an example of the CLI command `mxdb -d <DBNAME> -Q`, which prints information about the Virtual Oracle Service.



```
192.168.3.21 - PuTTY
$ mxdb -d OED01 -Q

Preparing to query the OED01 Service on all nodes.

Service OED01:
Connect through vhost: 192.168.3.83
Primary Node is:      192.168.3.21
Currently active on  192.168.3.21
Backups:             BACKUP-1 192.168.3.33 BACKUP-2 192.168.3.34

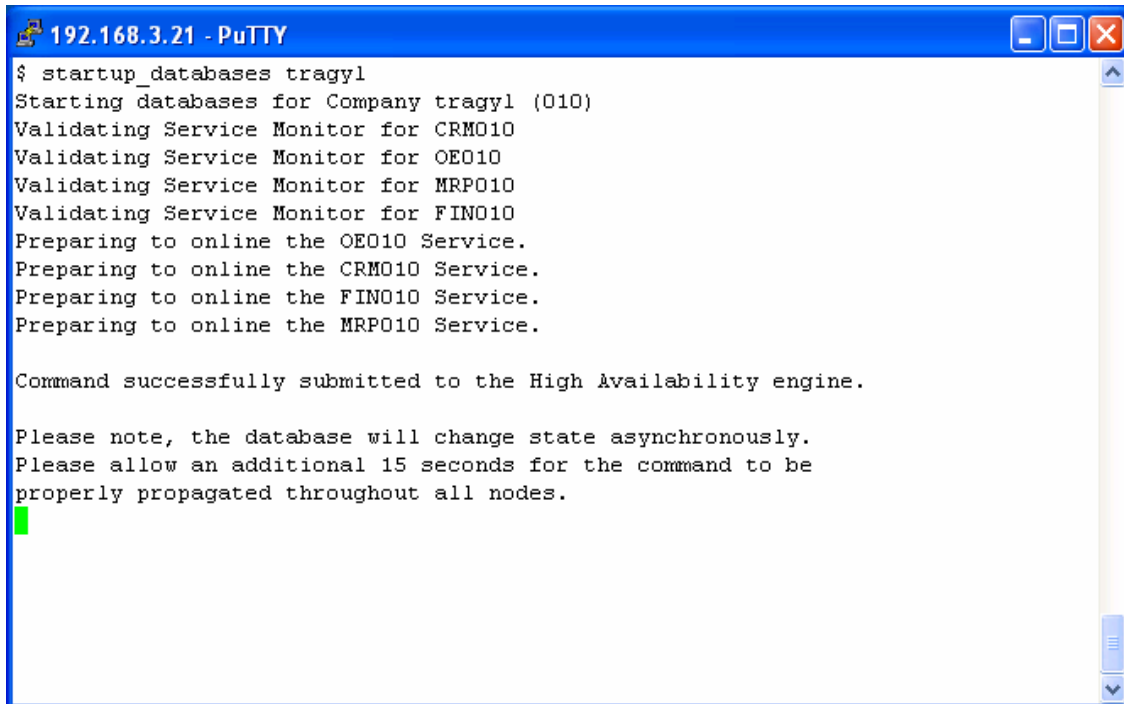
Enabled Nodes:       192.168.3.21 192.168.3.33 192.168.3.34
Disabled Nodes:
Maintenance Mode:   OFF

$ █
```

Figure 2.13: MxDB-Oracle-HiAv CLI command to query status

Extensible Management Interface

The MxDB-Oracle-HiAv CLI is fully scriptable—a quality much needed for tailoring the Solution Pack in complex environments. Here again, the DBA has full operational authority to perform operations on multiple databases in “groups.” Figure 2.14 shows the execution of the simple **startup_databases** script, which maps Virtual Oracle Services to a common theme such as Company Name and performs a startup operation on the entire set.



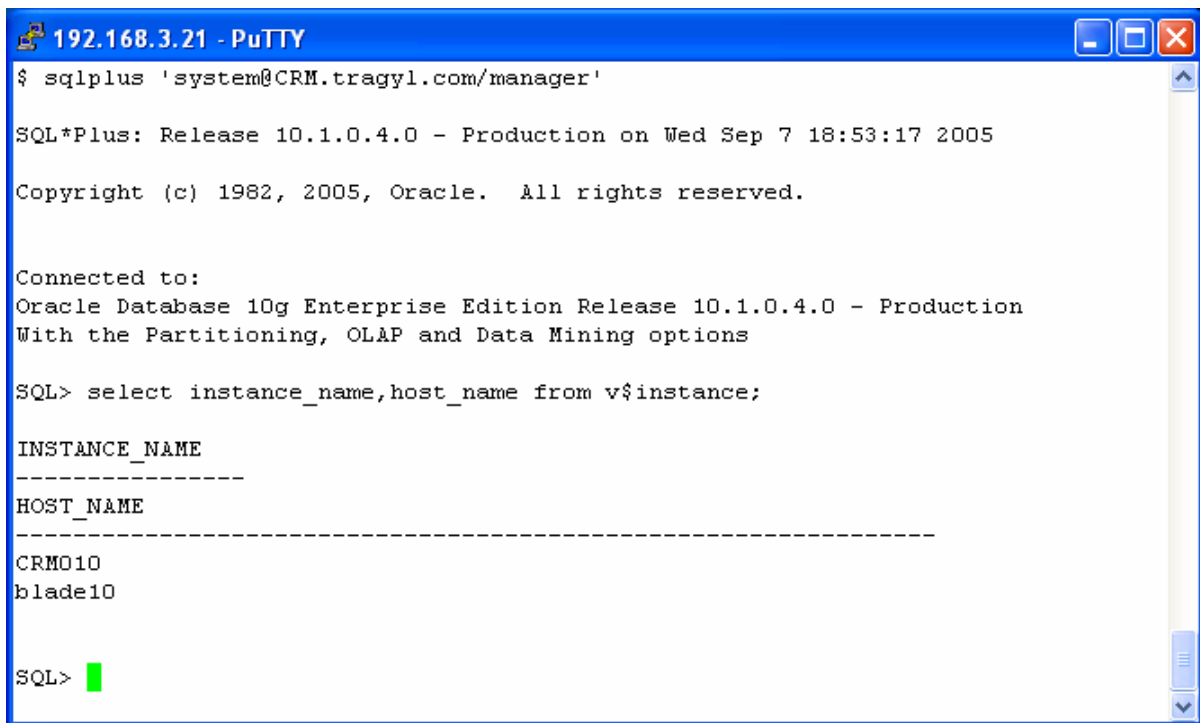
```
192.168.3.21 - PuTTY
$ startup_databases tragyl
Starting databases for Company tragyl (010)
Validating Service Monitor for CRM010
Validating Service Monitor for OE010
Validating Service Monitor for MRP010
Validating Service Monitor for FIN010
Preparing to online the OE010 Service.
Preparing to online the CRM010 Service.
Preparing to online the FIN010 Service.
Preparing to online the MRP010 Service.

Command successfully submitted to the High Availability engine.

Please note, the database will change state asynchronously.
Please allow an additional 15 seconds for the command to be
properly propagated throughout all nodes.
```

Figure 2.14: Performing Operations on a set of Virtual Oracle Services

Figure 2.15 shows an example taken from the proof of concept. In this case, sqlplus is used to connect to the CRM database for a hosted company called Tragyl. The operator of the sqlplus session doesn't need to know which server is hosting the CRM database, but a query from V\$INSTANCE reveals that the current host for the instance is blade10.



```
192.168.3.21 - PuTTY
$ sqlplus 'system@CRM.tragyl.com/manager'

SQL*Plus: Release 10.1.0.4.0 - Production on Wed Sep 7 18:53:17 2005

Copyright (c) 1982, 2005, Oracle. All rights reserved.

Connected to:
Oracle Database 10g Enterprise Edition Release 10.1.0.4.0 - Production
With the Partitioning, OLAP and Data Mining options

SQL> select instance_name,host_name from v$instance;

INSTANCE_NAME
-----
HOST_NAME
-----
CRM010
blade10

SQL>
```

Figure 2.15: Connecting to a Virtual Oracle Service

Proof of Concept

State-of-the-art and robust technologies were key to creating a suitable test system to prove the VMDB architecture. Figure 3.1 shows the cluster system components used for this proof of concept.

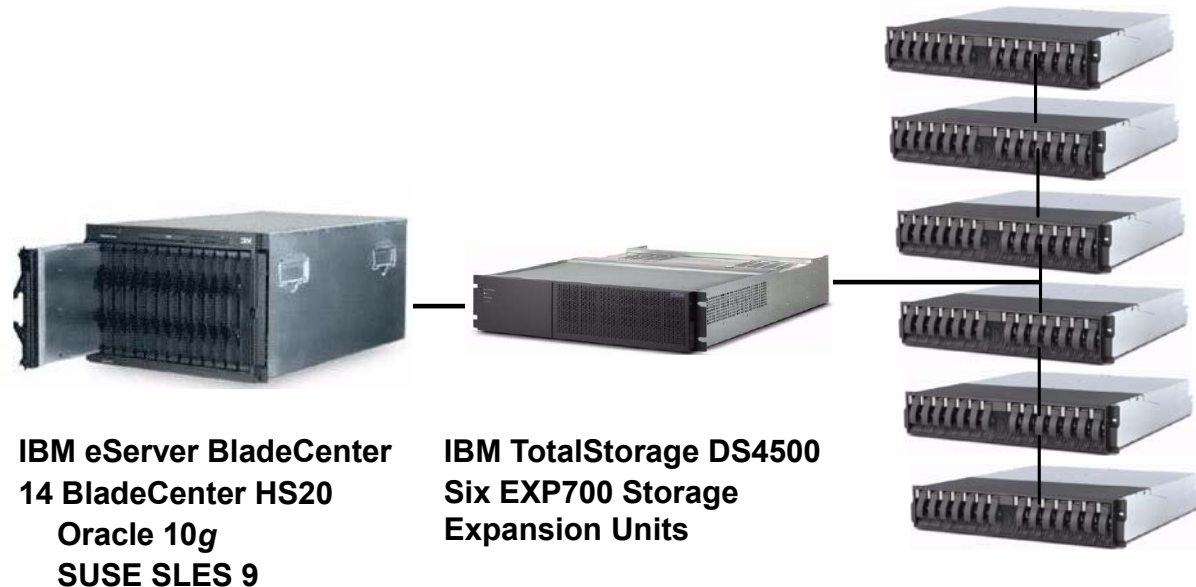


Figure 3.1: The proof-of-concept VMDB Database Cluster

Overview of the IBM eServer BladeCenter

To support the basic computing infrastructure needed by the VMDB proof of concept, it was important to choose a hardware platform that would showcase flexibility and manageability. The IBM eServer BladeCenter provides both of these attributes.

With emerging blade technologies enabling customers to reverse their “server sprawl” and collapse the complexity of their distributed IT infrastructure, we felt that this proof point would be an ideal opportunity to showcase this technology in a database cluster environment. Blades also deliver better management software, have less cable snarl, more expansion possibilities and smaller footprint requirements.

With those features in mind, it is easy to see why the IBM eServer BladeCenter, with its advantage of being rack-optimized and high-density in an innovative 7U form-factor design, was a perfect choice for deploying the Oracle10g RAC installation. The IBM eServer BladeCenter chassis accommodates up to 14 hot-swap 2-way Intel® Xeon™ processor-based blade servers and integrates within the chassis such key infrastructure components as Layer 2-7 Gigabit Ethernet Switching, SAN switching and centralized management tools.

With the need to connect large amounts of highly available disk storage to the BladeCenter, the storage subsystem was designed around the IBM TotalStorage DS4500⁷ Storage Server. The DS4500 is a RAID storage subsystem that contains Fibre Channel (FC) interfaces to connect both the host systems and the disk drive enclosures. With its dual 2Gbps controllers and high-availability design, the DS4500 delivers the necessary throughput to support this high-end proof point.

⁷ The IBM TotalStorage FASTT products were renamed in September 2004. The FASTT700 is now called DS4400; FASTT900 is called DS4500. The family is referred to as the DS4000 series.

IBM eServer BladeCenter Chassis

The IBM eServer Blade Center chassis, as shown in Figure 3.2, can accommodate up to 14 blade servers in its 7U form factor. Resources are shared among all of the blades and include power, switch, management and blower modules. The chassis provides high-speed I/O capabilities for all of the modules, thereby reducing the amount of cabling required in the datacenter. The Management Module, through remote access, allows the control of components in the enclosure.

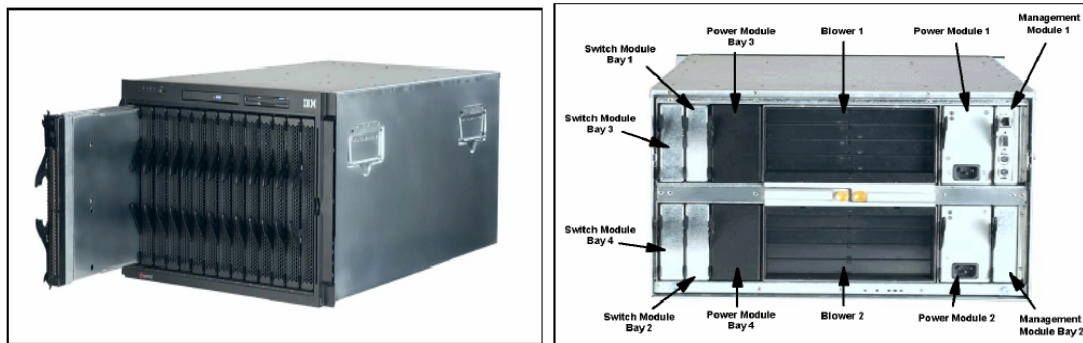


Figure 3.2: Front and rear views of the BladeCenter chassis

The IBM eServer BladeCenter chassis was configured as follows:

- Standard 48X CD-ROM and 1.44MB floppy accessible from all blades in the Media Tray.
- Management Module. The center for systems management on the BladeCenter, the Management Module is responsible for monitoring all components in the BladeCenter as well as each individual blade. It has the capability of detecting the condition and state of any of the installed components.
- Two additional 1200-watt hot-swap power modules (two are standard) were required to power blade slots 7-14. Installed as pairs, the power modules provide redundancy and power for robust configurations.
- Two 4-port Ethernet Switch Modules. Although not standard on the BladeCenter unit, the modules were necessary to provide the interconnectivity between the blades and Management Module and the external network. The module is a fully functional Ethernet switch with four external gigabit ports, two internal 10/100 links to the Management Module and 14 internal gigabit links to the blades. Two Ethernet Switch Modules were used in this proof point to support access to the external, public network (eth0) and the internal, private interconnect traffic (eth1).
- Two 6-port Fibre Channel Switch Modules. With the Fibre Channel Expansion Card in each blade server, the optional 6-port Fibre Channel Switch Module completed the required Fibre Channel connectivity to the SAN. Each port is capable of supporting transmission speeds of up to 2 Gbps after auto-negotiating with the DS4500 Storage Server.
- 14 IBM eServer BladeCenter HS20 blades.

IBM eServer BladeCenter HS20 Blade Server

The BladeCenter HS20 blades are high-throughput, two-way SMP-capable Xeon processor-based blade servers. With support for up to 8GB SDRAM memory and processor speeds of 2.4GHz to 3.6GHz, these blade servers are highly scalable. An integrated service processor on each blade server enables communication with the BladeCenter Management Module for remote control of server tasks. Also integrated on the HS20 are two Ethernet controllers that can be configured for either fault-tolerance or increased throughput through adapter teaming. With blade

server expansion card options such as Myrinet Cluster, Gigabit Ethernet, Fibre Channel, 1X InfiniBand and support for both EIDE and SCSI drives, a blade can be tuned to create customized solutions that match application needs.



Figure 3.3: View of an HS20 Blade Server

For the proof point, each blade server was configured with:

- Two x 3.2GHz Intel Xeon processors
- 4GB PC2100 DDR ECC Chipkill® memory
- One 40GB IDE drive
- Integrated dual Broadcom Gigabit Ethernet controllers
- One Fibre Channel Expansion card
- Su-SE Linux Enterprise Server 9 (Service Pack 1)

Managing the BladeCenter

As IT environments become more complex to manage and support, it is important that there be tools and processes that allow the end user to track and maintain the infrastructure. To reduce the sometimes time-consuming task of managing high-density computing environments, the IBM eServer BladeCenter has a built-in, Web-based GUI⁸ that allows remote access to the BladeCenter to remotely power on/off blades and manage I/O modules. Access is gained through a standard Ethernet port and a standard Web browser. Figure 3.4 shows the IBM BladeCenter's main menu with section headings for System Monitors, Blade Tasks, I/O Module, and Management Module Control.

The Monitors menu allows the user to view status, settings and other information for each of the key components configured in the IBM eServer BladeCenter. The displayed information includes:

- System Status of blade servers, I/O modules, management modules and power modules
- Event Log
- Front Panel and Blade Server LEDs
- Vital Product Data for blades, I/O module and Management Module

⁸ The IBM eServer BladeCenter is also tightly integrated with the IBM Director V4 systems management tool and the Rapid Deployment Manager; however, these system management products are beyond the scope of this paper. For additional information about these products, visit www.ibm.com.

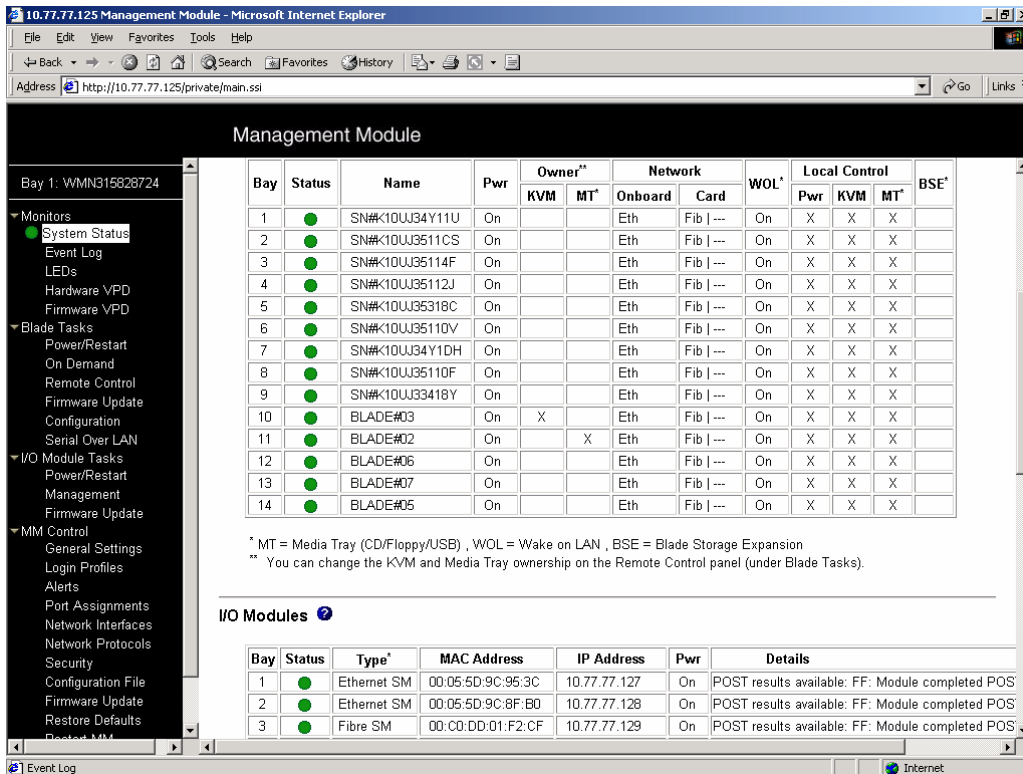


Figure 3.4: Portion of the System Status summary window

The settings for each blade server can be configured and controlled via the Blade Tasks section of the menu. The following tasks can be performed:

- Power/restart of individual or all blade servers (see Figure 3.5)
- Remote control of an individual blade to associate the media tray ownership
- Updates of firmware
- Configuration for each blade for KVM control, media tray control, Wake on LAN and boot sequence

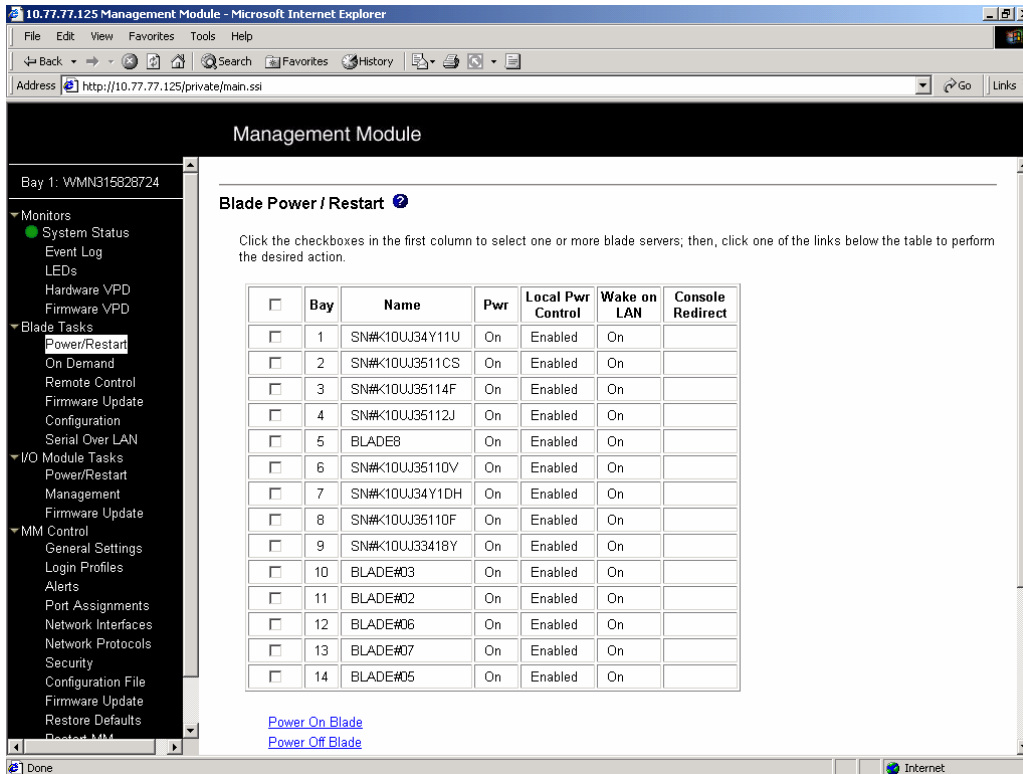


Figure 3.5: Portion of Blade Task menu selection

The heading of the I/O Module section of the menu depends on which optional switch modules are installed in the BladeCenter. In this particular installation there were two gigabit Ethernet switch modules and two Fibre Channel switch modules. The heading was displayed as "I/O Module Tasks" and the menu allowed the following:

- Power/restart of modules (see Figure 3.6)
- Individual switch management setting of IP network addresses and a drill-down for Advanced Management into each module to further configure the switch modules or to generate a telnet session/Web-based GUI to monitor and control the device
- Firmware upgrade capabilities

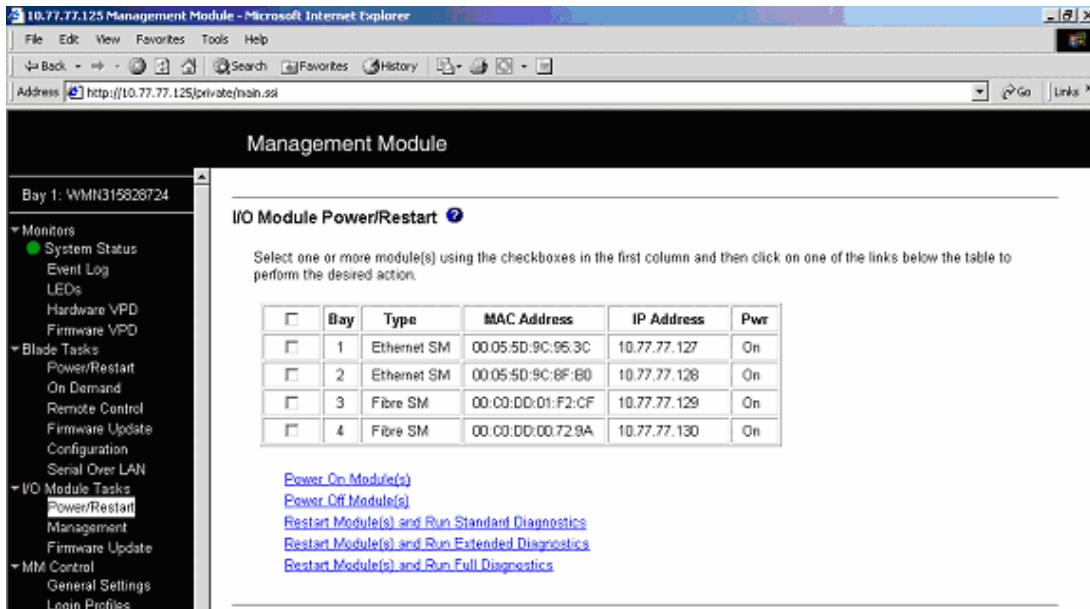


Figure 3.6: Partial display of I/O Module Tasks

The final module controlled through the Web-based system management GUI is the BladeCenter Management Module. The tasks include:

- General module settings such as name, date and time
- The ability to manage login profiles through ID and password control
- The ability to set alert levels and target users to receive notification (see Figure 3.7)
- Port assignments
- Network interfaces
- Network protocols
- Security
- Configuration file management with backup and restore options
- The ability to restore defaults
- The ability to restart the Management Module

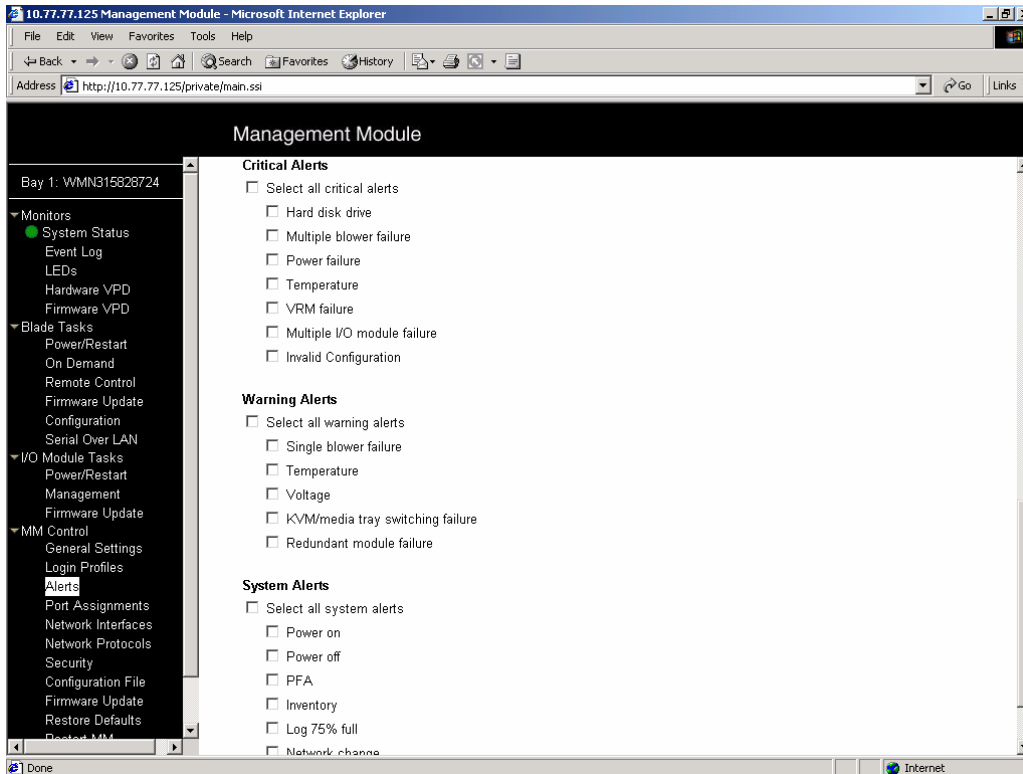


Figure 3.7: Sampling of alerts that can be monitored

Customizing BladeCenter I/O Modules to Match Application Needs

One of the strengths of a BladeCenter deployment is the flexibility to add modules to enable customized solutions that match application needs. From a physical perspective, adding these modules inside the chassis reduces the complexity of the external infrastructure by reducing cabling, power and space requirements. From a management perspective, complexity is reduced by having a centrally administered installation. Current I/O module options for the IBM eServer BladeCenter include:

- **Ethernet Network Switches:**
Ethernet switches designed by industry leading vendors like Cisco and Nortel provide a high-speed Ethernet connection between each blade server and the external network environment and integration of advanced Ethernet functionality such as Layer 2-7 Gigabit into the chassis.
- **FibreChannel Switches:**
FC switches in 2- and 6-port models from vendors QLogic, Brocade and McDATA support the integration of the BladeCenter architecture into enterprise fabric SAN with FC uplink transmission rates up to 2Gbps.
- **InfiniBand Switches:**
The InfiniBand switch module with its high bandwidth and low latency characteristics, consolidates I/O and storage across the entire BladeCenter or collection of BladeCenters to a centralized location for a cost savings and high-availability infrastructure.
- **Optical Pass-thru Module:**
The OPM allows unswitched, unblocked network connections to each blade server. This provides customers with the flexibility of integrating a BladeCenter into existing network infrastructures that have already standardized on a specific SAN fabric.

The requirements for standard Gigabit Ethernet connectivity and connection to a fault-resilient Fibre Channel SAN for the proof-point were easily met by literally sliding in the 4-port Gigabit Ethernet Switch Module and the 6-port Fibre Channel Switch Module (see Figure 3.8). These modules were immediately recognized by the BladeCenter Management Module and were available for configuration.



Figure 3.8: View of 4-port Ethernet Switch Module and 6-port Fibre Channel Switch Module

Managing the BladeCenter 4-port Gigabit Ethernet Switch Module

The Ethernet Switch Module (ESM) can be managed through a telnet session or a Web interface, which is the preferred method (see Figure 3.9). Each module on the BladeCenter is assigned a TCP/IP address that corresponds to the module slot in which the switch is installed. This makes it very easy to start a telnet session directly or to access the Web interface at the assigned address. Through the Web interface, the ability to manage and monitor performance of the Ethernet switch module is only several mouse clicks away.

From the main BladeCenter Management menu, selecting I/O Module Tasks -> Management and then choosing the Advanced Management button for the module of choice provides the end user the capability to drill down into a Web-based, graphical tool for monitoring the 4-port Gigabit switch. For ease of navigation, the menu is broken into four distinct categories:

- Switch configuration for port settings, VLANs, link aggregation, and so on
- Remote management setup
- Network monitoring to display port utilization statistics
- Maintenance tasks such as upgrading firmware, downloading configuration, and restarting modules

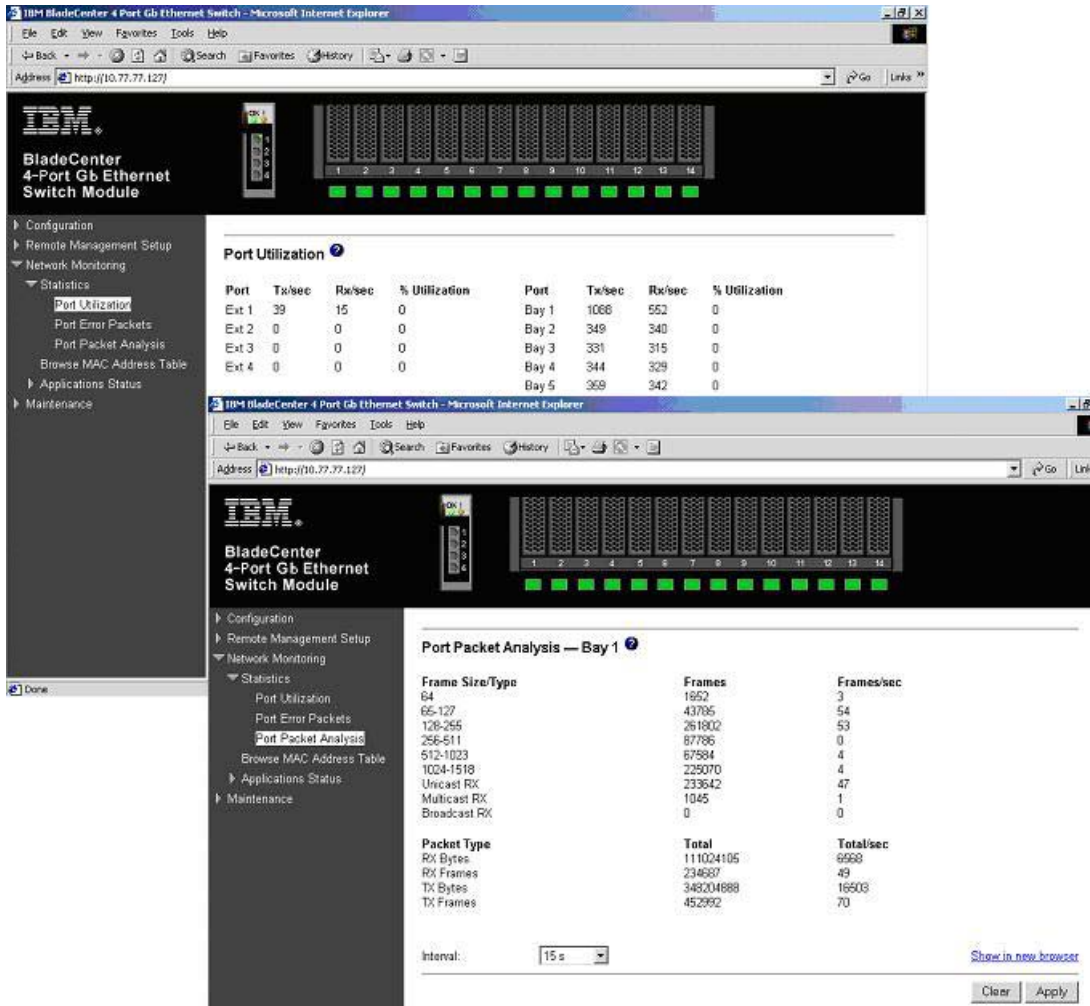


Figure 3.9: Multiple displays of performance information for the 4-port GB Ethernet Switch Module

Managing the BladeCenter Fibre Channel Switch Module

The configuration and management of the BladeCenter Fibre Channel Switch Module is facilitated with the SAN utility tool. This tool is a Java-based graphical user interface that allows viewing and configuration of ports, zoning and network setup, and troubleshooting through diagnostic functionalities. The SAN utility can be installed and executed from one of the blade servers or from an external monitoring system. It is supported in both Windows and Linux environments. The SAN utility provides a graphical view of the Fibre Channel subsystem and can be used to view, monitor or change network, switch module, and port configuration for one or more fabrics concurrently. There are two basic views available with the SAN utility: Topology Display and Faceplate Display.

The Topology Display (Figure 3.10) shows all switches that are able to communicate and all connections between switches. (A *fabric* is defined as one or more connected switches.) The graphic window of the topology display provides status information for switches, interswitch links, and the Ethernet connection.

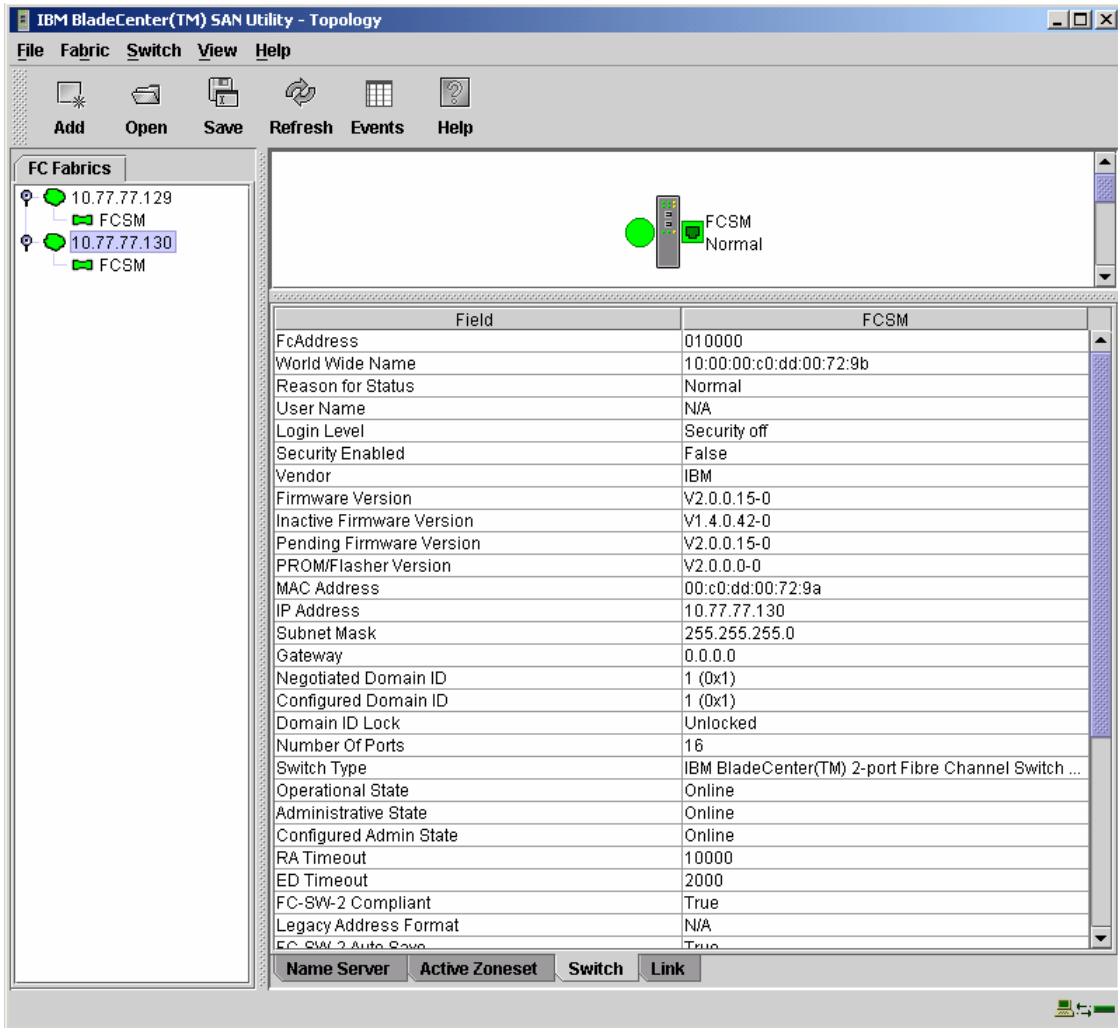


Figure 3.10: SAN Utility Topology Display

The BladeCenter Fibre Channel Switch Module information can also be viewed through the Faceplate Display (Figure 3.11), which shows the front of the switch and its related information: switch name, switch operational state, and port status.

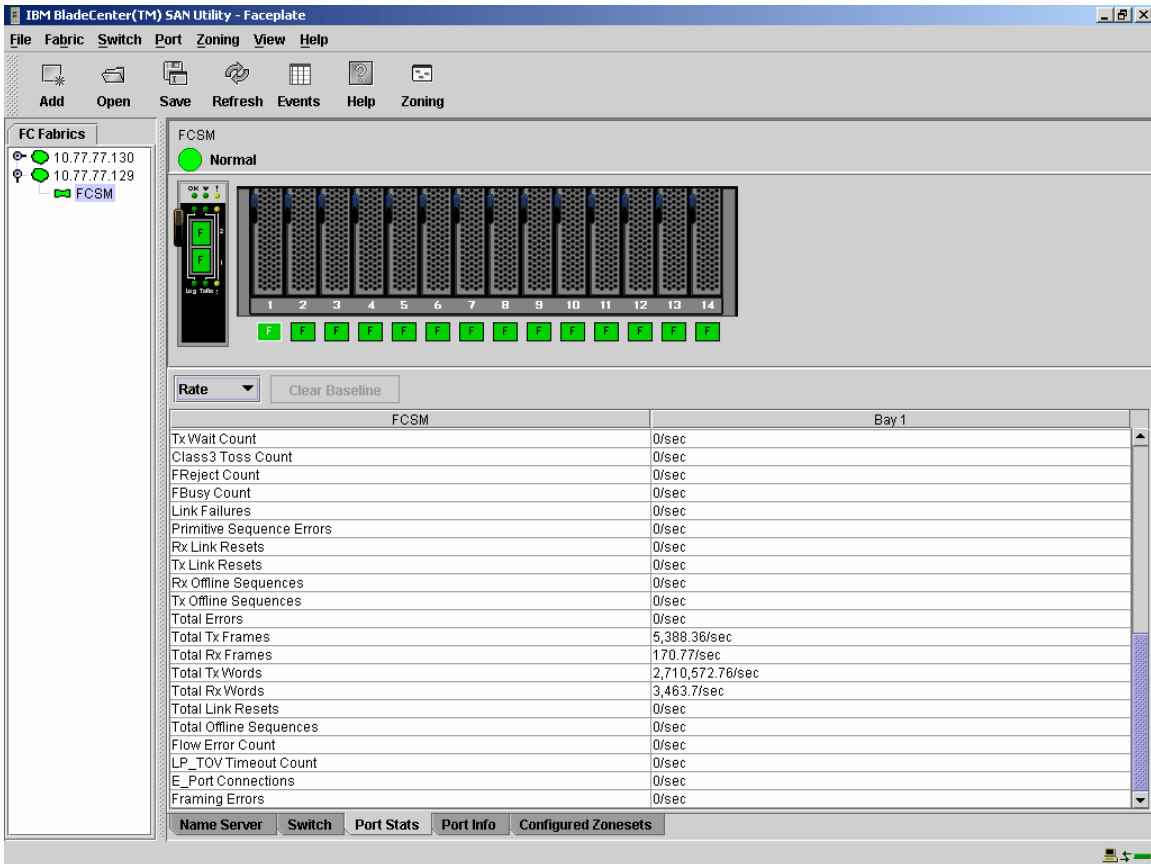


Figure 3.11: Faceplate display from the SAN Utility

The IBM eServer BladeCenter also includes a management tool called Fabric View. This application provides a method to visually monitor realtime traffic performance for each port on a switch as shown below in Figure 3.12. The graphs can be set up to display either Kbytes/sec or number of frames/sec.



Figure 3.12: Fabric View

Storage Subsystem

The IBM TotalStorage DS4000 series is designed to support the large and growing data storage requirements of business-critical applications. The DS4500 Storage Server is a RAID controller device that contains Fibre Channel (FC) interfaces to connect the host systems and the disk drive enclosures. The DS4500 used in this proof point has controllers that use the 2Gbps Fibre Channel standard on both the host side and the drive side. The DS4500 can support up to 224 FC disks and a maximum throughput of 795 MB/s when fully configured with all four host-side connections. To avoid single points of failure, it also supports high-availability features such as hot-swap RAID controllers, two dual-redundant FC disk loops, write-cache mirroring, redundant hot-swap power supplies, fans and dual AC line cords.

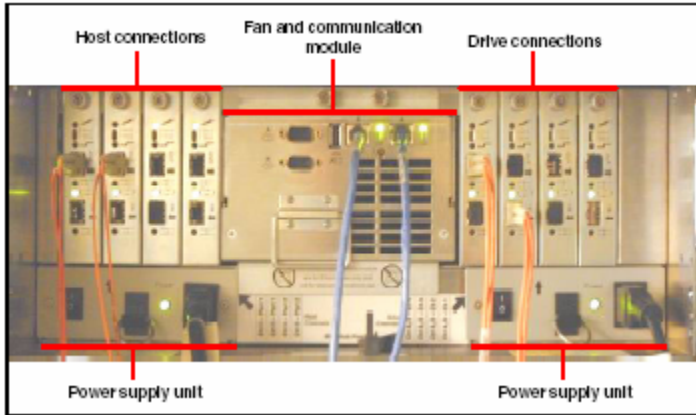
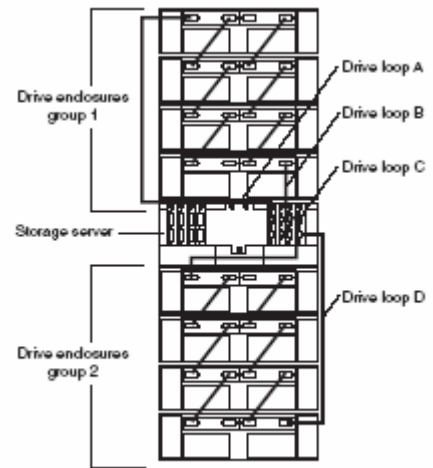


Figure 3.13: Rear view of the DS4500

For this implementation, the DS4500 had six DS4000 EXP700 drive enclosures attached with a total of 84 15K rpm 36.4GB drives. The Storage Manager Software was used to automatically lay out each of the arrays across the multiple controllers and drive loops. The DS4000 allocation heuristic does a reasonable job of distributing I/O traffic across available resources so there was no need to manually define the arrays. Because the DS4500 can support two redundant drive loops, the drive enclosures were set up to take advantage of this redundancy. If one data path fails, the controller uses the other data path to maintain the connection to the drive group.

This back view of the storage server and drive enclosures shows two redundant drive loops. Loop A and Loop B make up one redundant pair of drive loops. Loop C and Loop D make up a second redundant pair. The 84 drives were automatically distributed across four arrays of 20 drives each with a RAID-1 configuration and a 512K stripe size.



Managing the Storage System

The IBM TotalStorage DS4000 Storage Manager software simplifies management of extensive SAN installations. The Storage Manager software allows you to configure arrays and logical drives, to assign logical drives into storage partitions, to replace and rebuild failed disk drives, to expand the size of arrays and logical volumes, and to convert from one RAID level to another. It also gives you the ability to monitor performance. In addition to array/drive level control, the DS4000 Storage Manager can also be used to update the firmware and NVSRAM on the controllers.

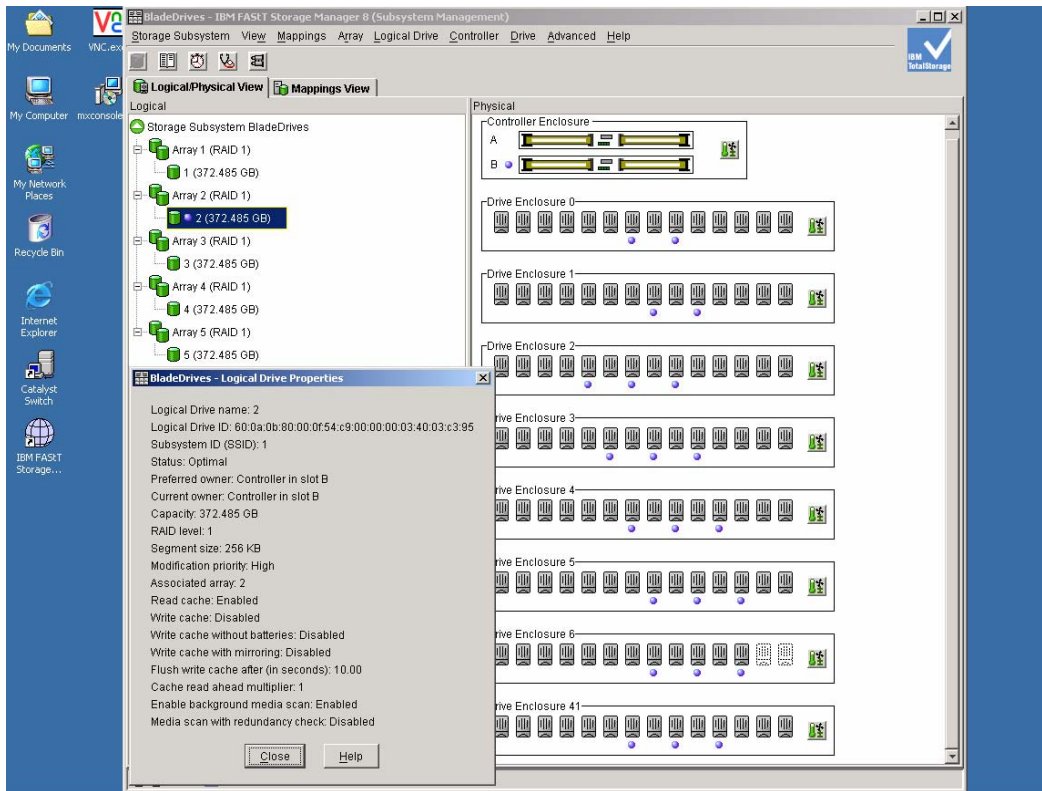


Figure 3.14: IBM DS4000 Storage Manager Software

The DS4000 Storage Manager is supported in both Windows and Linux environments and provides two methods for managing storage subsystems:

- Host-agent (in-band) management through the Fibre Channel I/O path to the host
- Direct (out-of-band) management over the network

The performance monitor data can be used to make storage subsystem tuning decisions. There are many settings in the DS4500 that can have a large impact on performance. These include cache parameters, controller ownership, segment size, RAID levels, logical drive modification priority, and remote volume mirroring. To assist with overall storage tuning, the DS4000 Storage Manager has a built-in performance monitor that displays statistics for Total I/Os, Read Percentage, Cache Hit Percentage, Current KB/s, Maximum KB/s, Current I/Os and Maximum I/Os (Figure 3.15). The successful tuning of the DS4500, like other server components, is dependent on finding the right balance of availability and high performance. The performance monitor in the DS4000 Storage Manager is just one of the tools available to help end users improve SAN performance.

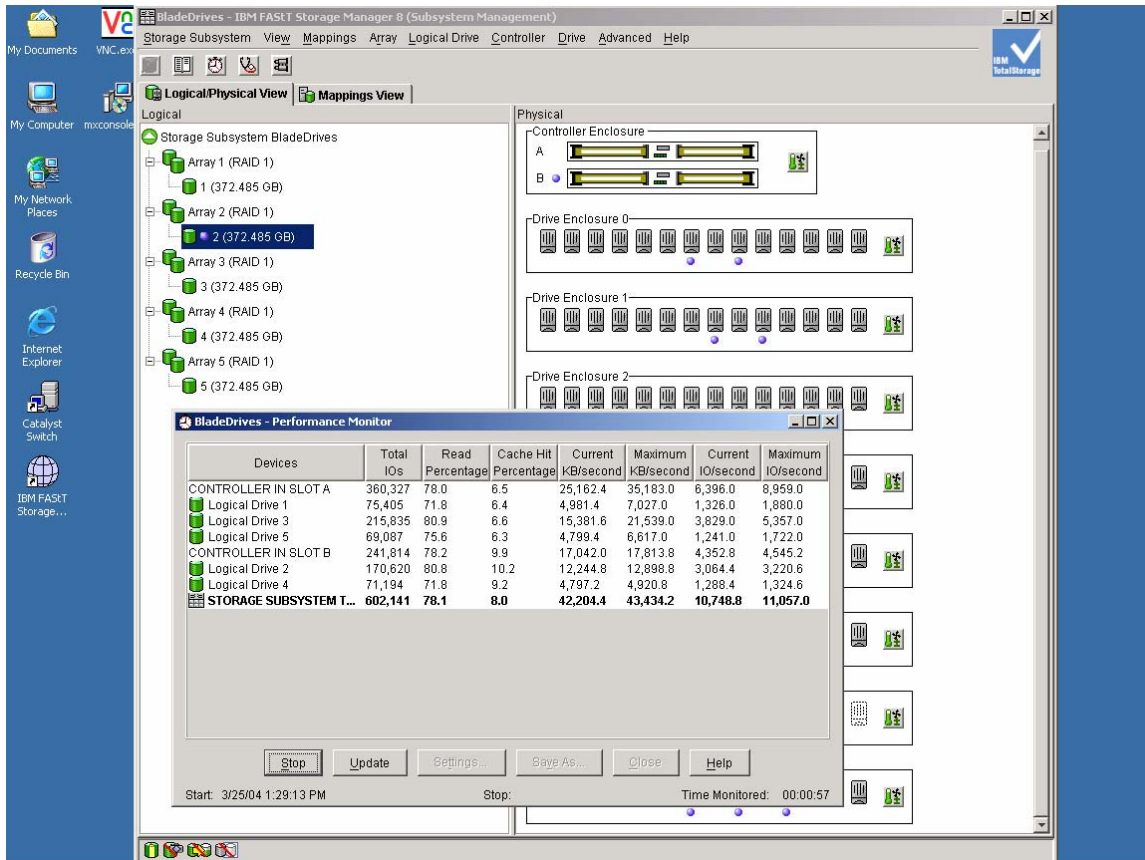


Figure 3.15: Storage Manager Performance Monitor

Test Description

The proof of concept was based on a service provider model wherein 15 fictitious companies are hosted, each with four databases:

- Order Entry (OE). A simple, traditional Order Entry schema consisting of customers, orders, order items, and stock. The Order Entry database for each company had a minimum of 5,000,000 customers⁹.
- Financials (FIN).
- Customer Relationship Management (CRM).
- Manufacturing Resource Planning (MRP).

Each company is assigned a "Company Number." This number combined with the database name is, by convention, the ORACLE_SID value (e.g., the Company 8 ORACLE_SID for CRM is CRM008). However, due to the simplicity of the PolyServe MxDB-Oracle-HiAv Solution Pack, the actual users of the CRM database at Company 8 access the database service simply as CRM.glavol.com (e.g., user/pw@CRM.glavol.com).

⁹ Special thanks to James Morle of Scalabilites, LTD for the use of the Order Entry kit.

The company names are provided in Figure 3.16 to make the proof of concept more understandable.

Company Number	Company Name
001	Proveo, Inc.
002	The Steren Company
003	Glucia
004	Acetal LTD
005	Prunis, Inc.
006	Protol
007	The Sparic Company
008	Glavol
009	Protic, Inc.
010	Tragyl
011	Bromet
012	Ibicid, Inc.
013	Anodix
014	Ozonix
015	Trivium

Figure 3.16: Proof of Concept fictitious company names

Physical Database Layout

Consolidating large numbers of small clusters into a large cluster needs to result in simplified management. Flexibility is essential. The PolyServe cluster volume manager and cluster filesystem both support online growth. Simply allocate another LUN from the storage, import it with a mouse click, add it to the volume and grow the filesystem. So, consolidating a large number of databases into a single large filesystem makes sense. All datafiles were located in the dboptimized, mounted `/u02` filesystem as shown in Figure 2.3. The total number of physical disks supporting the Oracle datafiles, redo logs, control files and archived redo logs was 80.

For the proof of concept, the databases were created using the simplified Oracle Managed Files(OMF) method via DBCA. The proof of concept was meant to prove manageability in a complex environment, so it made little sense to use complex tablespace definitions. In fact, OMF works extremely well. Tablespaces created by OMF are optimized for the norm; however, specialized tuning may still be required in certain cases.

To help illustrate the simplicity of this type of file management, Figures 3.17 and 3.18 show `ls(1)` commands that show how simple the layout can be even with 60 databases in the cluster. The data files are named automatically by OMF, which ensures unique naming. The `mount(1)` command in Figure 3.19 shows that the `/u02` filesystem is mounted with the Matrix Server dboptimized mount option, which provides cache-coherent direct I/O. This means that database

accesses by Oracle are rendered direct, and all other tools (e.g., backup tools) can also get direct I/O without recompiling with code changes to perform the open(2) system call with the O_DIRECT flag. With the dboptimized mount option, if I/O can be rendered direct, it will be. I/O that cannot be rendered direct is serviced through the buffered I/O path, and all the while cache coherency is maintained.

The model followed traditional OFA convention; thus, the filesystem mounted on /u02 was the database volume. Combining OFA, OMF and the PolyServe product makes large numbers of databases quite simple to manage.

```

192.168.3.21 - PuTTY
$ pwd
/u02/oradata
$ ls
CRM001  CRM007  CRM013  FIN004  FIN010  flash_recovery      MRP005  MRP011  OEO01_test_ts  OEO07  OEO13
CRM002  CRM008  CRM014  FIN005  FIN011  flash_recovery_area MRP006  MRP012  OEO02          OEO08  OEO14
CRM003  CRM009  CRM015  FIN006  FIN012  MRP001             MRP007  MRP013  OEO03          OEO09  OEO15
CRM004  CRM010  FIN001  FIN007  FIN013  MRP002             MRP008  MRP014  OEO04          OEO10
CRM005  CRM011  FIN002  FIN008  FIN014  MRP003             MRP009  MRP015  OEO05          OEO11
CRM006  CRM012  FIN003  FIN009  FIN015  MRP004             MRP010  OEO01   OEO06          OEO12
$ ls -l OEO03/datafile | more
total 26888548
-rw-rw---- 1 oracle dba 104865792 2005-08-30 15:49 o1_mf_cstate_i_K6cae1WBx3i81j_.dbf
-rw-rw---- 1 oracle dba 314580992 2005-08-30 17:20 o1_mf_cus_idx_u6cae1DmR3i81b_.dbf
-rw-rw---- 1 oracle dba 1572872192 2005-08-30 17:11 o1_mf_customer_d6cae1Wifii812_.dbf
-rw-r----- 1 oracle dba 629153792 2005-09-01 13:34 o1_mf_data_lkkgv8q3_.dbf
-rw-rw---- 1 oracle dba 104865792 2005-08-30 15:49 o1_mf_date_idx_E6cae18yl3i81g_.dbf
-rw-rw---- 1 oracle dba 104865792 2005-08-30 15:48 o1_mf_delcust_j6cae1VThi1815_.dbf
-rw-r----- 1 oracle dba 943726592 2005-09-01 13:38 o1_mf_idx_lkkgvhtmt_.dbf
-rw-rw---- 1 oracle dba 2516590592 2005-08-30 17:15 o1_mf_item_96cae1gbV1810_.dbf
-rw-rw---- 1 oracle dba 2411732992 2005-08-30 17:13 o1_mf_item_n6cae15aX1817_.dbf
-rw-rw---- 1 oracle dba 2516590592 2005-08-30 17:14 o1_mf_item_p6cae1G2ni818_.dbf
-rw-rw---- 1 oracle dba 524296192 2005-08-30 17:20 o1_mf_itm_idx_P6cae1f2gi81l_.dbf
-rw-rw---- 1 oracle dba 524296192 2005-08-30 17:20 o1_mf_itm_idx_s6cae1Rm3i81a_.dbf

```

Figure 3.17: OFA oradata directory in the cluster filesystem

```

192.168.3.21 - PuTTY
$ pwd
/u02/oradata
$ ls -l */datafile/*system* |more
-rw-rw---- 1 oracle dba 471867392 2005-08-29 01:28 CRM001/datafile/o1_mf_system_a6m7e1on51P050_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 13:54 CRM002/datafile/o1_mf_system_Mfm7e1uCy2Fn60_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 19:19 CRM003/datafile/o1_mf_system_fpm7e1Oqt13W60_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 19:20 CRM004/datafile/o1_mf_system_Fym7e1FqCNU70_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 13:54 CRM005/datafile/o1_mf_system_fIm7e1lvSA380_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 19:18 CRM006/datafile/o1_mf_system_uRm7e1VJfIa0_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 13:58 CRM007/datafile/o1_mf_system_UOn7e1tmO4LJ0_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 19:30 CRM008/datafile/o1_mf_system_F9n7e1FwW1Ji10_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 20:48 CRM009/datafile/o1_mf_system_Ghn7e1aSfZQ10_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 09:54 CRM010/datafile/o1_mf_system_prn7e1ul322q20_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 20:24 CRM011/datafile/o1_mf_system_RAn7e1rXS3GY20_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 20:29 CRM012/datafile/o1_mf_system_2In7e1Ychqx30_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 09:57 CRM013/datafile/o1_mf_system_qOn7e1vaf0540_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 20:30 CRM014/datafile/o1_mf_system_5Vn7e1e1UuE40_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 16:26 CRM015/datafile/o1_mf_system_h2o7e1unR3Vc50_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 13:56 FIN001/datafile/o1_mf_system_K8m7e1cSsX50_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 19:09 FIN002/datafile/o1_mf_system_8im7e1sOE1hw60_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 19:16 FIN003/datafile/o1_mf_system_Arm7e1QcA2U470_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 19:17 FIN004/datafile/o1_mf_system_ZAm7e16Qg3xD70_.dbf
-rw-rw---- 1 oracle dba 461381632 2005-08-27 19:15 FIN005/datafile/o1_mf_system_AKm7e1jcV3bc80_.dbf

```

Figure 3.18: Simple ls(1) command shows system tablespace datafiles for 20 databases

```

192.168.3.21 - PuTTY
$ mount
/dev/hda2 on / type reiserfs (rw,acl,user_xattr)
proc on /proc type proc (rw)
tmpfs on /dev/shm type tmpfs (rw)
devpts on /dev/pts type devpts (rw,mode=0620,gid=5)
/dev/hda1 on /boot type ext2 (rw,acl,user_xattr)
usbfs on /proc/bus/usb type usbfs (rw)
/dev/psd/psd1p2 on /u01 type psfs (rw,shared,data=ordered)
/dev/psv/psv1 on /u02 type psfs (rw,dboptimize,shared,data=ordered)
$ df
Filesystem            1K-blocks      Used Available Use% Mounted on
/dev/hda2              36918868    7044128  29874740  20% /
tmpfs                  2075476         4   2075472    1% /dev/shm
/dev/hda1               99043        10716   83213    12% /boot
/dev/psd/psd1p2       34413436    9866852  24546584  29% /u01
/dev/psv/psv1        689915696  252145192  437770504  37% /u02
$

```

Figure 3.19: Simple df(1) output shows capacity for all databases

Measurement Results

The entire value proposition represented in the FDC architecture would be worthless if not for the absolute performance available in the cluster configuration tested. During the proof of concept, several performance aspects were analyzed and are presented here. The data falls into two categories:

- Microbenchmark Results
- Oracle Server measurements

Microbenchmark Analysis

Collecting low-level performance metrics such as disk I/O bandwidth on large clusters can be difficult, especially when the only I/O stimulus is the collective 60 databases used in the VMDB proof of concept. So, the PolyServe *randio* microbenchmark kit was first used to get low I/O performance measurements of the cluster and SAN hardware configuration.

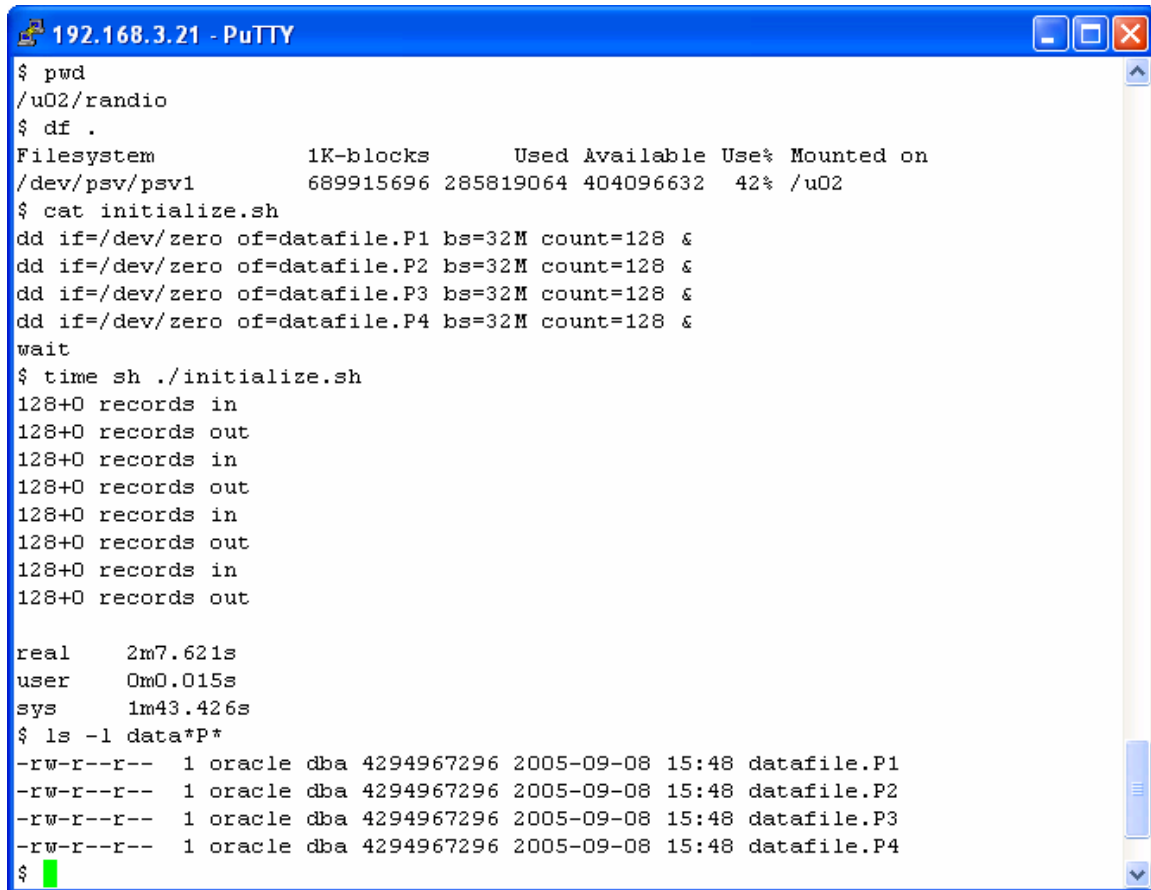
The *randio* kit has the following characteristics:

- **Multiple Execution Streams.** Based on a Linux process model, this closely mimics the Oracle dedicated server model.
- **Synchronous I/O.** Since the majority of I/O with Oracle OLTP is synchronous single blocks reads, this characteristic helps mimic the Oracle server.
- **Tunable Read:Write Ratio.** Unless otherwise noted, test was set at 75:25.
- **Tunable I/O Size.** This testing was conducted at 4K, which is a common Oracle block size for OLTP.
- **Random I/O.** I/O is randomized throughout the entire target file. There are no hotspots coded into the benchmark.
- **Random Memory Stress.** The read buffer is a random location in an 8MB memory segment in the address space of each process. This exercises memory management.

Sequential Write Performance

To prepare for the randio test, target datafiles needed to be created. This was seen as an opportunity to measure the sequential write performance available through the PolyServe cluster filesystem, cluster volume manager and IBM TotalStorage SAN. Since the storage is laid out with RAID 1+0, there is a gross overhead of 100% on writes at the storage array level.

Figure 5.1 shows a shell session using a simple script that executes four `dd(1)` processes in parallel, each initializing 4GB files in the `/u02/randio` directory. The filesystem on `/u02` is mounted with the PolyServe `dboptimized` parameter so all I/Os were direct I/O. The screen capture shows a complete time of 127.6 seconds or 128MB/s (16GB/127.6). Again, considering the array-level cost of the mirror writes, this is considerable sequential write throughput for this storage configuration.



```
192.168.3.21 - PuTTY
$ pwd
/u02/randio
$ df .
Filesystem            1K-blocks      Used Available Use% Mounted on
/dev/psv/psv1         689915696 285819064 404096632  42% /u02
$ cat initialize.sh
dd if=/dev/zero of=datafile.P1 bs=32M count=128 &
dd if=/dev/zero of=datafile.P2 bs=32M count=128 &
dd if=/dev/zero of=datafile.P3 bs=32M count=128 &
dd if=/dev/zero of=datafile.P4 bs=32M count=128 &
wait
$ time sh ./initialize.sh
128+0 records in
128+0 records out
128+0 records in
128+0 records out
128+0 records in
128+0 records out
128+0 records in
128+0 records out
128+0 records in
128+0 records out

real    2m7.621s
user    0m0.015s
sys     1m43.426s
$ ls -l data*P*
-rw-r--r-- 1 oracle dba 4294967296 2005-09-08 15:48 datafile.P1
-rw-r--r-- 1 oracle dba 4294967296 2005-09-08 15:48 datafile.P2
-rw-r--r-- 1 oracle dba 4294967296 2005-09-08 15:48 datafile.P3
-rw-r--r-- 1 oracle dba 4294967296 2005-09-08 15:48 datafile.P4
$
```

Figure 5.1: One node, four processes achieve aggregate 128MB/s sequential write throughput

Random I/O Performance

The random I/O test suite consisted of three main tests. Unless otherwise noted, the multiple stream test executes 80 streams. All tests perform as close to 10,000,000 random 4KB I/O operations as possible. For example, at the full end of the scalability spectrum (14 nodes), there were 1,120 streams each performing 8,928 random I/O operations ($80 * 14 * 8,928 == 9,999,360$).

The test suites executed were:

Randio Test 1: Single Stream, Multiple Node Scale-up, Read Only, Small File. This test executes a single stream of the benchmark accessing a single file small enough to fit in the array cache (2GB) using synchronous I/O. The test is executed on varying node counts with the intent of measuring “SAN Fairness,” a measurement of the I/O symmetry of the cluster. Near-linear scalability on this test is fully expected up to the point where the array controller approaches maximum theoretical throughput. Anything significantly less likely points to nodes that may not be configured correctly for SAN access. Also, since all I/O occurs to a single CFS file, any software locking or serialization issues would be completely transparent.

At first glance, this test may seem of no relevance in an Oracle context. That couldn't be further from the truth. There have been a number of array controllers brought to market with bottlenecks in the sheer number of I/O operations they can service per second (I/Ops). That is, they bottleneck on I/Ops before hitting the limit on bandwidth. It is very helpful, in high-rate Oracle OTLP performance troubleshooting, to know that there are no arbitrary limitations on operations per second.

Randio Test 2: Single Stream, Multiple Node Scale-up, 25% Write, Small File. This test is the same in all respects to Randio Test 1 except 25% of the I/O requests are writes. The test measures the point at which the array controller starts to bottleneck on cache write-backs.

Randio Test 3: Multiple Streams, Multiple Node Scale-up, 25% Write, Large File. This test measures the relationship between the overall SAN bandwidth and the size of the cluster. A poor scale curve in this test merely indicates that the servers are capable of more I/O than the current SAN configuration can satisfy. This situation is easily remedied by adding disks to the array and then dynamically adding those disks to the PolyServe cluster volume manager and cluster filesystem. Most often, a scalability problem with this test indicates there are not enough physical disks to handle the I/O load.

The file being accessed in this test case was 128GB and resided in the /u02 cluster filesystem for direct I/O.

Single Stream, Multiple Node Scale-up, Read Only, Small File.

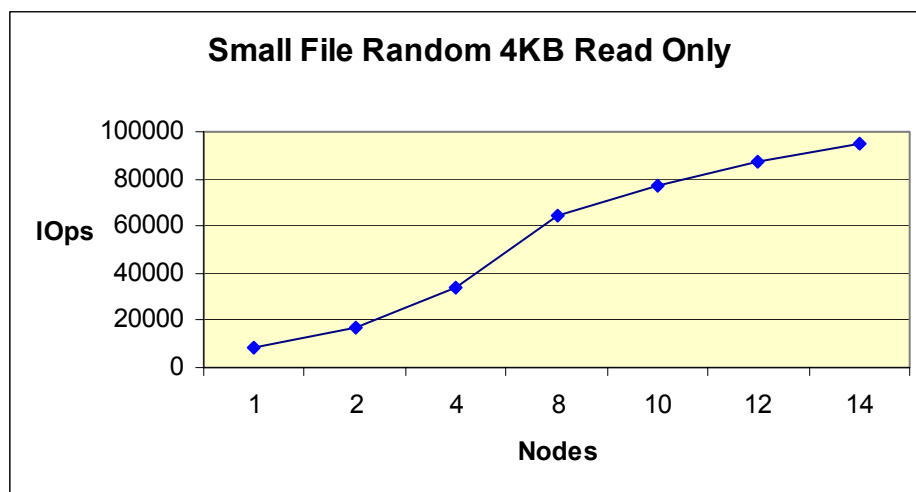


Figure 5.2: Rand IO Test 1. Small file random 4K physical transfers, read only

The single stream, multiple-node, scale-up tests (Randio Test 1) showed that the SAN was able to deliver 90% scalability through 10 nodes as shown in Figure 5.2. At 10 nodes, the I/O rate was 86,956 4KB reads per second for 339 MB/s, which was getting close to the maximum theoretical

limit of the DS4500 array as configured for this proof of concept with only two host-side mini hubs. (The DS4500 is capable of supporting two more high-density mini hubs, essentially doubling the data throughput.) Scalability from that point dropped to 85% at 12 nodes and finally, 83% at 14 nodes as should be expected. This test proved that there were no scalability limits imposed by the cluster configuration as it pertained to the SAN up to the point at which the array controller is saturated. The demonstrated scale curve indicates that given sufficient array controller bandwidth, any number of nodes would be serviced satisfactorily by this architecture.

Single Stream, Multiple Node Scale-up, 25% Write, Small File.

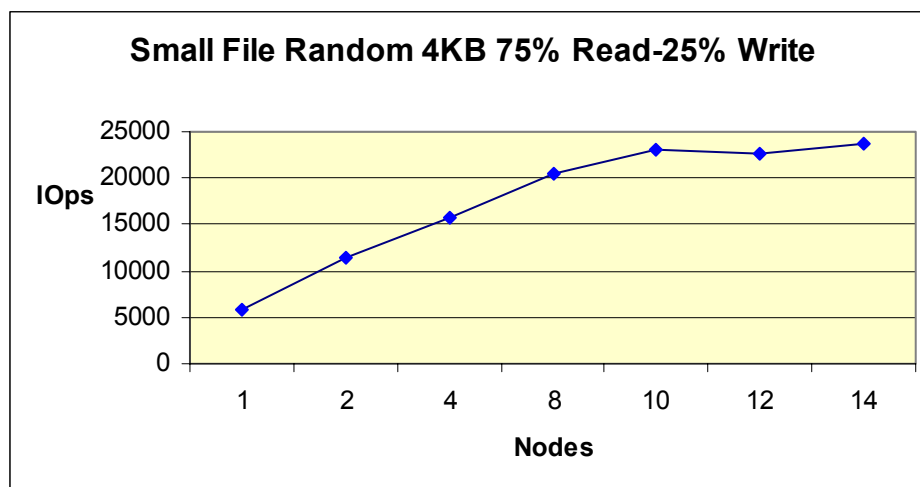


Figure 5.3: Rand IO Test 2. Small file random 4K physical transfers, 25% write

As shown in Figure 5.3, the single stream 25% write test (Randio Test 2) showed that the SAN (as configured) was able to deliver 98% scalability to two nodes with an I/O rate of 11,325 transfers per second. Given the built-in 25% write ratio, 2,831 of the 11,325 were write operations. Since the storage was a RAID 1+0 configuration, this equates to 70 writes and 106 reads per second, per spindle (80 spindles total). While the reads were satisfied in the array cache, the writes had to be flushed to disk. Given the write cost, the scalability was reduced to 68% at four nodes, where the I/O rate was 15,797 transfers per second of which 3,949 were writes (49 per second). The added mirror write overhead at four nodes would therefore be 98 writes per second per physical drive.

All disk drives have a fixed threshold for random writes. Even at 15,000 rpm, the response time will increase as the number of IOs exceed 98 random writes per second. The affect of the physical disk configuration is seen in the scale curve in Figure 5.3. All is not lost, however. The random read-only test established that the array controller has significantly more bandwidth. The solution to this scalability issue would be as simple as adding disks to the array and subsequently adding them to the PolyServe dynamic volume and cluster file system.

Simply configuring more spindles under the data rectifies this sort of performance issue. Simple performance issues such as this are easily understood and rectified. No need to change to radical new storage approaches, just simple optimized filesystem storage with the flexibility to add capacity on the fly. The small file tests exercised the array controller and the ability to perform cache write-backs.

Multiple Streams, Multiple Node Scale-up, 25% Write, Large File.

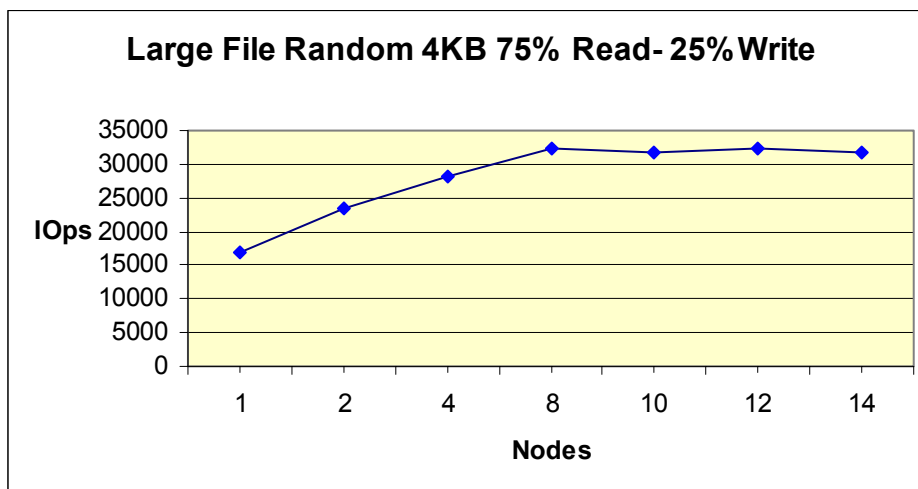


Figure 5.4: Large file random read/write test

The next test was to access a file that was significantly larger than the array cache. In preparing for Randio Test 3, a target datafile of 128GB was chosen. With an array cache capacity of 2GB, a 128GB file will render the cache footprint less than 2% with this disk access pattern.

Randio Test 3 uses multiple execution streams per node, 80 in fact. Figure 5.4 shows that the baseline at one node was 16,806 operations per second, which is 290% more I/O than was possible with a single stream even in the small file test where all reads are satisfied in array cache. However, entering the scale curve at 210 physical transfers per second per spindle makes scalability very difficult. Scalability was 70% to two nodes where the I/O rate was 23,529 I/Os per second. The results go flat at eight nodes, where the I/O rate was 32,258 I/O per second. This point was the demonstrated full bandwidth level for the physical disk layout. At that scale, 403 operations per second per spindle were being issued, of which 100 were writes. Again, given the RAID overhead, the total disk-level I/O per second would therefore be 603 transfers per second.

Given these test results on this SAN configuration, the apparent optimum 4KB random I/O rate for real-life Oracle applications would fall in the 16,000-20,000 I/O per second range. Being ever mindful of the RAID cost at the array level, the high end of this estimate would equate to 30,000 array-level I/Os per second or an average of 375 transfers per second per spindle. Results would vary based on significant differences in read:write ratio as would be expected.

Oracle Parallel Query Tests

Each CRM database had a Customer Credit analysis component with varying record counts. The queries conducted against this set of tables simulates customer spending trend analysis. In the case of the Steren Company (Company 2), there were 200,000,000 rows in the main table called *card*. The card table was located in a tablespace with a 16KB blocksize with datafiles stored in the PolyServe cluster filesystem. The card table consumes 7.2GB of the card tablespace.

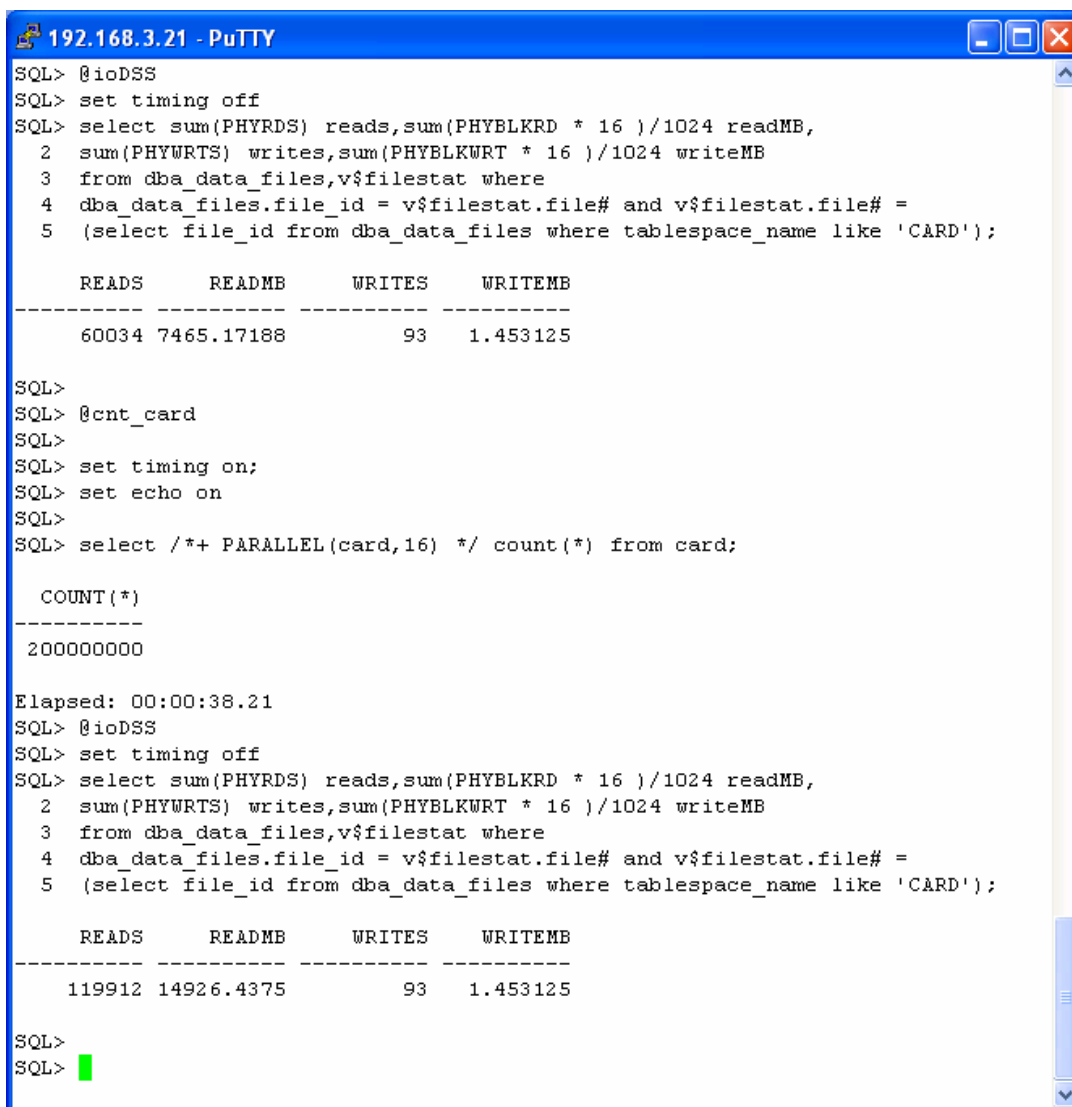
Two tests were run using the card table:

Light Weight Scan. A simple select count(*) was executed to measure single node Oracle Parallel Query throughput.

Index Create Time. An index was created on the vendor_id column of the card table, which was of type number(7).

Full Table Scan Results

The CRM database for Company 2 (CRM002) was started on blade2 of the cluster. The degree of parallelism for the table was set to 16. Figure 5.5 shows that the full table scan returned a count of 200,000,000 rows in 38.2 seconds. Using the **ioDSS.sql** script shows the scan throughput was 195MB/s with an average of 1,567 transfers of 128KB per second. This is very close to the maximum theoretical throughput of the single active 2Gb FC host bus adaptor.



```
SQL> @ioDSS
SQL> set timing off
SQL> select sum(PHYRDS) reads,sum(PHYBLKRD * 16 )/1024 readMB,
 2 sum(PHYWRTS) writes,sum(PHYBLKWRT * 16 )/1024 writeMB
 3 from dba_data_files,v$filestat where
 4 dba_data_files.file_id = v$filestat.file# and v$filestat.file# =
 5 (select file_id from dba_data_files where tablespace_name like 'CARD');

      READS      READMB      WRITES      WRITEMB
-----
      60034 7465.17188          93  1.453125

SQL>
SQL> @cnt_card
SQL>
SQL> set timing on;
SQL> set echo on
SQL>
SQL> select /*+ PARALLEL(card,16) */ count(*) from card;

      COUNT(*)
-----
200000000

Elapsed: 00:00:38.21
SQL> @ioDSS
SQL> set timing off
SQL> select sum(PHYRDS) reads,sum(PHYBLKRD * 16 )/1024 readMB,
 2 sum(PHYWRTS) writes,sum(PHYBLKWRT * 16 )/1024 writeMB
 3 from dba_data_files,v$filestat where
 4 dba_data_files.file_id = v$filestat.file# and v$filestat.file# =
 5 (select file_id from dba_data_files where tablespace_name like 'CARD');

      READS      READMB      WRITES      WRITEMB
-----
      119912 14926.4375          93  1.453125

SQL>
SQL> █
```

Figure 5.5: Single Node Parallel Query Scan Throughput

The DS4500 Performance Monitor aids in assessing how balanced the I/O is at the LUN level. Figure 5.6 shows that during the full table scan test, logical drives 1, 2, 3 and 4 were very evenly accessed due to the striping characteristics of the PolyServe cluster volume manager. Logical drive 5 held the /u01 filesystem and no database accesses were performed there.

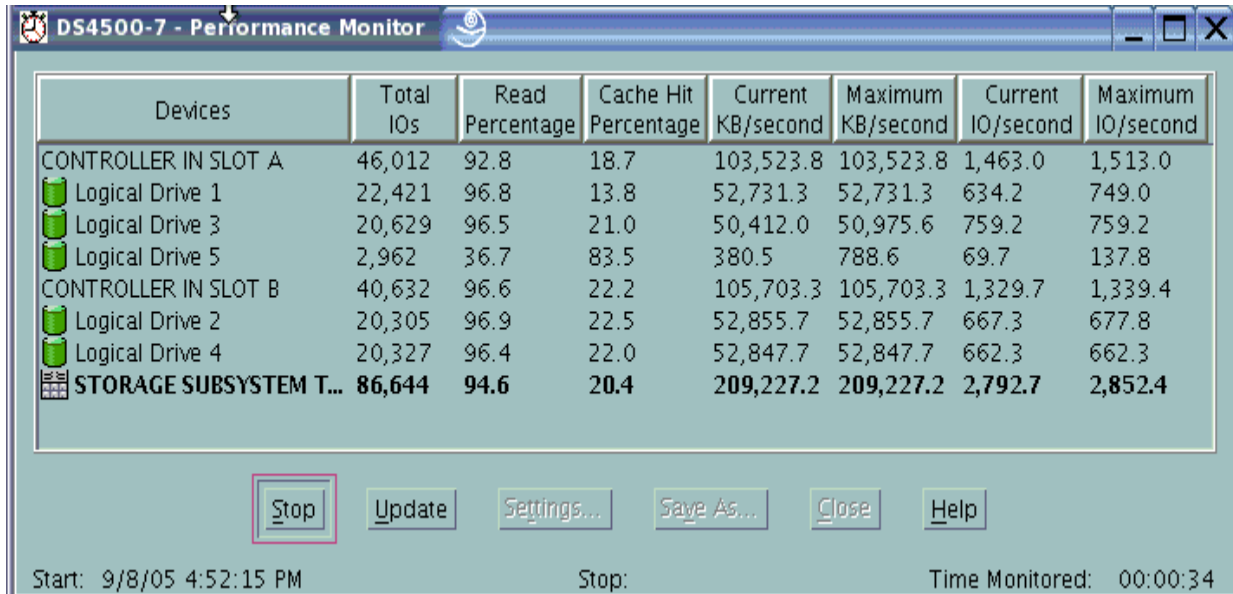


Figure 5.6: DS4500 Performance Monitor during Parallel Query Full Scan Operation

Index Creation Results

In this test, both the table and the index reside in the card tablespace, as there was sufficient space and a high performance physical disk layout via the PolyServe cluster volume manager to accommodate both objects. Figure 5.7 shows that the index create time was 703 seconds. This equates to the very processor-intensive activity involved in scanning and sorting 284,495 index keys per second on a dual-processor system. The I/O profile consisted of 186 physical disk transfers per second, of which 53% were writes. The I/O throughput was 15.3 MB/s.

```

192.168.3.21 - PuTTY
SQL> @../iodSS

```

READS	READMB	WRITES	WRITEMB
10	.15625	3	.046875

```

SQL> @index1
SQL> set timing on
SQL> create /*+ PARALLEL(DEGREE 16) */ index c_vid_idx on card(vendor_id)
 2 tablespace card
 3 PCTFREE 0
 4 NOLOGGING
 5 PARALLEL (DEGREE 14);

Index created.

Elapsed: 00:11:42.87
SQL> @../iodSS
SQL> set timing off
SQL> select sum(PHYRDS) reads,sum(PHYBLKRD * 16 )/1024 readMB,
 2 sum(PHYWRTS) writes,sum(PHYBLKWRT * 16 )/1024 writeMB
 3 from dba_data_files,v$filestat where
 4 dba_data_files.file_id = v$filestat.file# and v$filestat.file# =
 5 (select file_id from dba_data_files where tablespace_name like 'CARD');

```

READS	READMB	WRITES	WRITEMB
59983	7465.5625	71201	3284.5

```

SQL>

```

Figure 5.7: Timed index creation on 200,000,000 row table

High Availability and Manageability

Manageability

When application performance starts to degrade, DBAs invariably look into the database server with such tools as statspack, dbconsole or Grid Control. How many statspacks and Grid Control Web pages need to be perused when the problem is one application nestled amongst dozens of others? Whether in a sea of little pair-wise clusters or in a consolidated Flexible Database Cluster, it is very difficult to dial in on just one database.

Even more difficult is determining the disruption one application is causing to others due to shared resources such as processors, memory, HBA and SAN infrastructure. Does this mean SANs are bad? No. It is cumbersome enough managing, say, 20 failover clusters, how much more so if they each have dedicated “silos” of DAS storage. For successful consolidation, the platform needs to offer cluster-aware performance monitoring to augment the database-level monitoring available in Grid Control or dbconsole.

The MxODM library used in the Flexible Database Cluster is a critical tool used in analyzing performance of certain databases relative to all the others in the cluster. Remember, the cluster shares the SAN, too. To illustrate this point, a test was set up where two companies and their associated databases were hosted on a single node – blade1.

The Order Entry database for Proveo, Inc (Company 1) was stressed with 48 zero-think-time users. The Order Entry workload for Proveo is fairly consistent. The test harness produces statspack reports roughly every two minutes. A simple look at the number of physical reads in each report provides a reasonable representation of the application throughput. That is, the workload cannot take new orders if it isn't reading blocks of data from the tablespaces. A quick glance at the MxDB-Oracle-HiAv GUI or CLI or at the PolyServe Management Console quickly reveals that the other company hosted on blade1 was The Steren Company (Company 2).

The scenario was such that Proveo, Inc. is not getting serviced at the established service level. Where to begin?

In Figure 5.8, **mxodmstat** was used to report I/O only for the OE001 and OE002 instances broken out into read and write, synchronous and asynchronous. The output is quite revealing. The OE001 instance is performing a steady stream of synchronous single-block reads along with a fairly consistent stream of asynchronous writes. This is the typical OLTP I/O profile. On the other hand, OE002 is performing large amounts of asynchronous large reads, as well as significant bursts of large asynchronous writes.

These two databases are being hosted on the same server, but they are in no way alike in terms of I/O profile. Further analysis was required.

```

$ mxodmstat -i2 -a op -I OE001 OE002
      OE001                                OE002
      Read                                Write                                Read                                Write
Sync Async KB/s Ave ms Sync Async KB/s Ave ms Sync Async KB/s Ave ms Sync Async KB/s Ave ms
13914 577 117839 2 56 43398 322768 3 10881 53299 5694170 3 2540 94954 8178096 7
564 0 4512 3 0 164 1203 3 0 82 10016 9 0 48 9600 8
660 0 5280 2 0 118 838 2 0 72 8736 9 0 40 7944 19
669 0 5352 2 0 138 954 2 0 61 7688 12 0 16 3312 20
776 0 6204 2 0 180 1343 7 0 101 12392 6 0 28 5612 21
772 0 6180 2 0 162 1195 2 0 108 13284 6 0 27 5276 15
838 6 6776 2 1 194 1369 2 0 104 12832 6 0 22 4376 17
852 0 6820 1 0 142 1038 2 0 88 10844 9 0 27 5168 15
454 0 3629 2 0 101 718 3 22 108 12746 8 9 357 28255 10
625 0 5000 2 0 192 1362 10 36 190 17187 4 15 560 44510 8
610 0 4921 2 4 169 1516 7 27 184 17442 4 11 502 41464 10
631 0 5048 2 0 164 1159 3 40 172 15319 6 15 498 40020 11
860 0 6876 2 0 169 1195 3 48 124 12773 7 5 209 16046 7
719 0 5752 2 0 163 1128 2 0 90 10980 8 0 1 16 1
960 0 7684 1 0 190 1384 2 0 86 10464 8 0 1 16 2
1102 0 8812 1 0 178 1266 2 0 94 11552 7 0 2 4 1
1200 0 9600 1 0 225 1736 3 0 83 10204 8 0 1 16 4
1122 0 8972 1 0 192 1254 2 0 76 9548 14 0 1 16 3
1020 0 8160 1 0 190 1478 2 0 72 8920 8 0 28 5756 8
1106 0 8844 1 0 188 1305 2 0 74 8992 6 0 55 11140 5

```

Figure 5.8: Using **mxodmstat(8)** to monitor two databases simultaneously

Continuing with **mxodmstat**, Figure 5.9 shows how informative it is to drill down to the process types that are doing the I/O. With two executions of the **mxodmstat** command, we were able to determine that OE001 and OE002 do not have a similar I/O profile and that more importantly, OE002 is involved in some Parallel Query activity such as index creation.

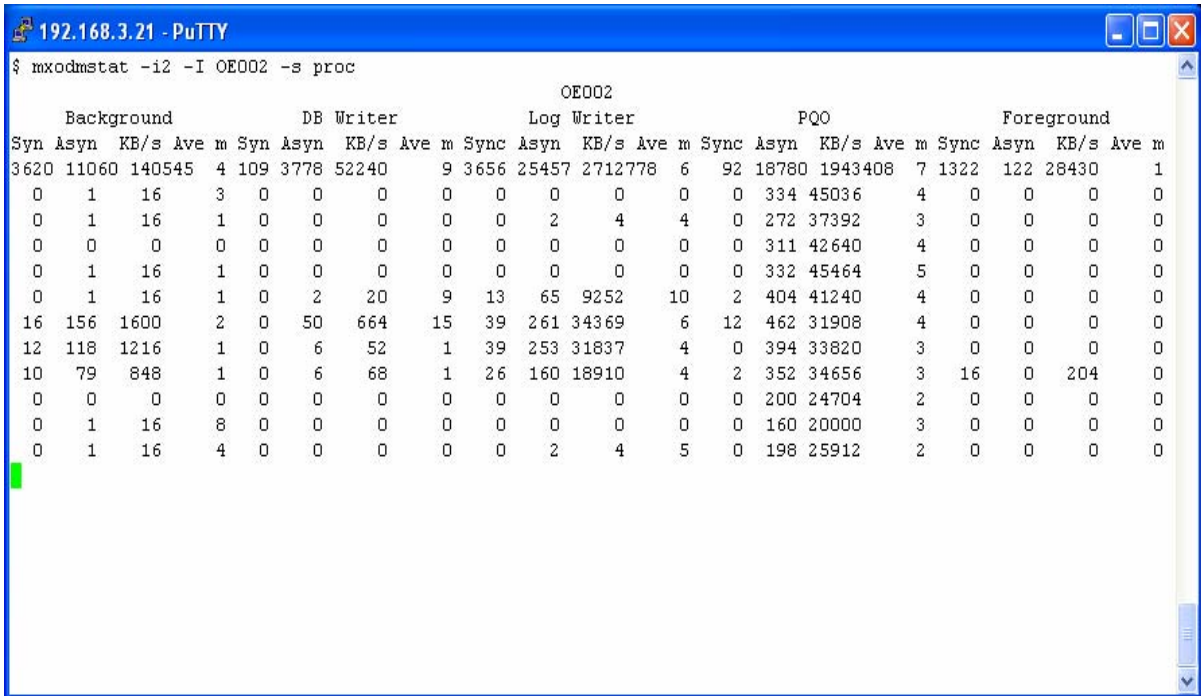
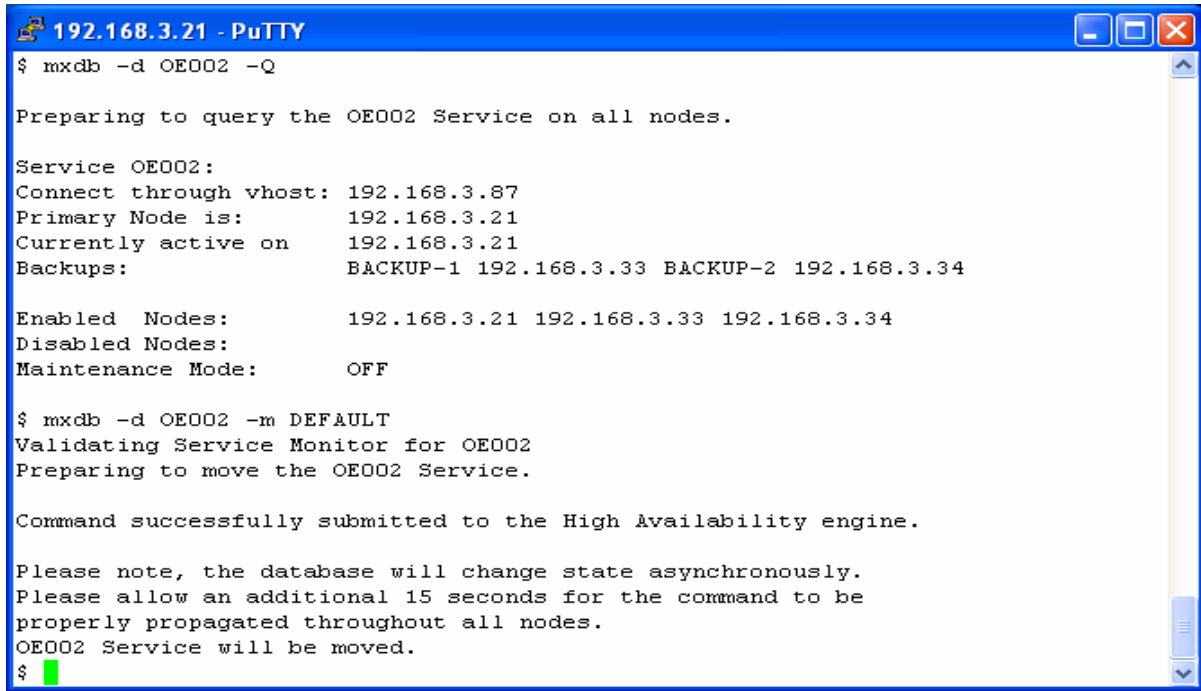


Figure 5.9: Using **mxodmstat(8)** to drill down to the process-level I/O for a given instance

In the model, contact was made with the DBA of the Steren Company (Company 2). A short planned outage was agreed upon to rehost the Steren Company databases to another, less utilized server. This would be a win-win situation for both companies since both would then have 100% dedicated server resources.

Figure 5.10 shows the simplicity of the MxDB-Oracle-HiAv CLI command to query current status and then to rehost the service to the first backup node (blade13 in this case). The same functionality is also available through the MxDB-Oracle-HiAv GUI and the PolyServe Management Console. Contrast the simplicity of this powerful architecture with the complete lack of functionality offered by alternative approaches.



```
192.168.3.21 - PuTTY
$ mxdb -d OE002 -Q

Preparing to query the OE002 Service on all nodes.

Service OE002:
Connect through vhost: 192.168.3.87
Primary Node is:      192.168.3.21
Currently active on  192.168.3.21
Backups:              BACKUP-1 192.168.3.33 BACKUP-2 192.168.3.34

Enabled Nodes:       192.168.3.21 192.168.3.33 192.168.3.34
Disabled Nodes:
Maintenance Mode:   OFF

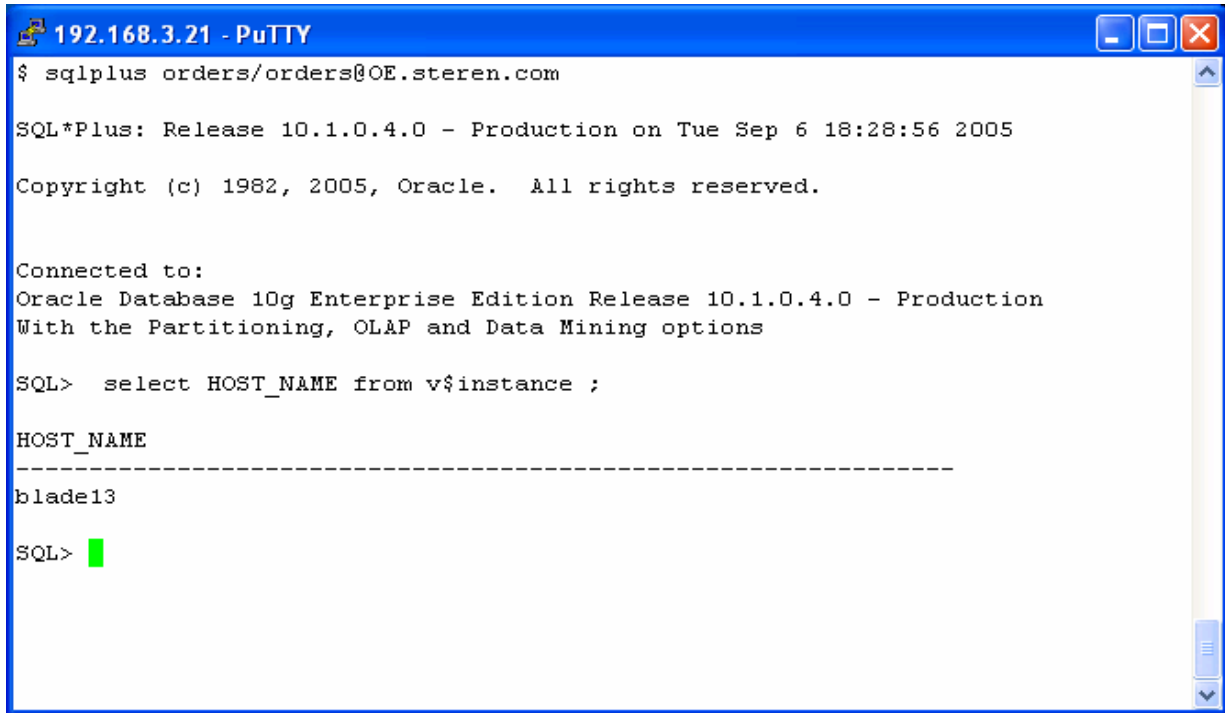
$ mxdb -d OE002 -m DEFAULT
Validating Service Monitor for OE002
Preparing to move the OE002 Service.

Command successfully submitted to the High Availability engine.

Please note, the database will change state asynchronously.
Please allow an additional 15 seconds for the command to be
properly propagated throughout all nodes.
OE002 Service will be moved.
$
```

Figure 5.10: Rehosting a database with one simple CLI command

In Figure 5.11, a sqlplus session connected to the OE.steren.com database service and executed a query from v\$instance showing that the service is now hosted on blade13. However, the users of Order Entry at Steren wouldn't really need to know that detail since they connect through the PolyServe Virtual Oracle Service. This was just a way to determine the host on which the Oracle server is executing.



```
192.168.3.21 - PuTTY
$ sqlplus orders/orders@OE.steren.com

SQL*Plus: Release 10.1.0.4.0 - Production on Tue Sep 6 18:28:56 2005

Copyright (c) 1982, 2005, Oracle. All rights reserved.

Connected to:
Oracle Database 10g Enterprise Edition Release 10.1.0.4.0 - Production
With the Partitioning, OLAP and Data Mining options

SQL> select HOST_NAME from v$instance ;

HOST_NAME
-----
blade13

SQL>
```

Figure 5.11: After the database is rehosted, a query from v\$instance reports the current host

Continuing with the exercise, **mxodmstat** was used again to monitor the activity generated by both companies after the rehosting. This time, however, the reporting level is *Node* using the **-N** option (**-I** could also have been used). Whereas the **mxodmstat** output in Figure 5.12 showed the Order Entry instance for Proveo peaked at a little over 1,000 physical reads per sample, once OE002 was rehosted, the same workload was able to generate peaks of nearly 2,300 physical reads per second. Likewise, peak writes jumped from roughly 1,700 to roughly 2,500 per sample period. Steren's PQO workload, on the other hand, was reaching peaks of roughly 75MB's after being rehosted, whereas before it topped out at roughly 60MB/s.

While this was not a traditional benchmark-style performance measurement, the proof of concept was not a benchmark. The intent was to prove improved manageability through advanced monitoring and simplified application rehosting. The architecture proved to offer the necessary tools to perform such dynamic systems management.

```

192.168.3.21 - PuTTY
$ mxodmstat -i3 -a op -N blade1 blade13
          blade1                               blade13
          Read                                 Read
          Write                                Write
Syn Asy  KB/s Ave m Syn Asy  KB/s Ave m Syn Asy  KB/s Ave m Syn Asy  KB/s Ave m
39403 26308 578818 1 431 152993 1203176 2 2668 4156 495735 4 98 5765 392832 4
2259 0 18069 0 0 211 1645 12 0 137 16813 8 0 111 24233 7
2299 0 18395 0 0 246 1877 4 8 213 26681 7 3 188 18830 7
1853 0 14821 1 0 237 1849 4 40 214 19278 2 17 853 51841 4
1754 0 14032 1 0 316 2364 3 85 209 19813 2 14 698 41913 4
2090 0 16717 0 0 321 2361 2 0 138 16707 7 0 0.67 11 1
1057 0 8453 0 0 272 1951 3 0 105 12843 7 0 100 20459 6
1565 4 12571 1 0.67 336 2522 4 0 152 18539 5 0 98 19965 7
1792 0 14361 1 3 281 2098 4 0 142 17216 6 0 77 14188 7
606 0 4845 1 0 224 1695 4 14 157 22065 6 6 332 28016 7
467 0 3739 2 0 27 182 2 49 211 17825 2 20 925 57142 5

```

Figure 5.12: Substantial performance gains after dynamic database rehosting

After rehosting OE.steren.com, statspack reports were checked for the expected physical I/O increase by the OE.proveo.com database service. In Figure 5.13, a simple **grep(1)** command showed that physical reads increased as much as 333% from the 700-775 range prior to rehosting OE.steren.com.

```

192.168.3.21 - PuTTY
$ ls -lt OE.proveo.com_statspack*
-rw-r--r-- 1 oracle dba 80270 2005-09-06 18:35 OE.proveo.com_statspack.out.09.06_18.35
-rw-r--r-- 1 oracle dba 80619 2005-09-06 18:32 OE.proveo.com_statspack.out.09.06_18.32
-rw-r--r-- 1 oracle dba 81344 2005-09-06 18:30 OE.proveo.com_statspack.out.09.06_18.30
-rw-r--r-- 1 oracle dba 80613 2005-09-06 18:28 OE.proveo.com_statspack.out.09.06_18.28
-rw-r--r-- 1 oracle dba 80326 2005-09-06 18:26 OE.proveo.com_statspack.out.09.06_18.26
-rw-r--r-- 1 oracle dba 83163 2005-09-06 18:25 OE.proveo.com_statspack.out.09.06_18.25
-rw-r--r-- 1 oracle dba 80791 2005-09-06 18:23 OE.proveo.com_statspack.out.09.06_18.23
$ grep 'Physical reads:' *statspack.out.*
OE.proveo.com_statspack.out.09.06_18.23:          Physical reads:          772.20          21.47
OE.proveo.com_statspack.out.09.06_18.25:          Physical reads:          721.42          29.12
OE.proveo.com_statspack.out.09.06_18.26:          Physical reads:        1,374.64          32.65
OE.proveo.com_statspack.out.09.06_18.28:          Physical reads:        2,130.90          42.05
OE.proveo.com_statspack.out.09.06_18.30:          Physical reads:        2,571.12          51.46
OE.proveo.com_statspack.out.09.06_18.32:          Physical reads:        1,520.77          40.47
OE.proveo.com_statspack.out.09.06_18.35:          Physical reads:        1,651.48          32.26
$

```

Figure 5.13: Statspack reports increased throughput after dynamic rehosting

High Availability

PolyServe Matrix Server with the MxDB-Oracle-HiAv Solution Pack offers a vastly improved availability model over traditional “pair-wise” clusters. Gone are the days of configuring a new two-node cluster every time the need to deploy a new application arises. With the M: N model supported by PolyServe high availability, adding an application costs at most, the addition of one server to the existing cluster. Depending on utilization levels, a new application could just as easily be hosted by an existing lightly-utilized server. There is no need to dedicate a server for failover. One server can be configured as the first backup server for any number of Virtual Oracle Services. In the proof of concept, nodes 13 and 14 were established as the failover nodes for all 60 databases. But how well does this model work?

During the proof of concept, a server failure was simulated by powering off blade2. The company hosted on blade2 was The Steren Company; the instances running on that node were OE002,MRP002,CRM002 and FIN002. Although all four databases failed over, the purpose of the test was to analyze the outage to Steren Order Entry in the event of blade2 failure. Figure 5.14 shows the visual feedback on the Management Console at the moment Matrix Server determined blade2 was no longer functioning.

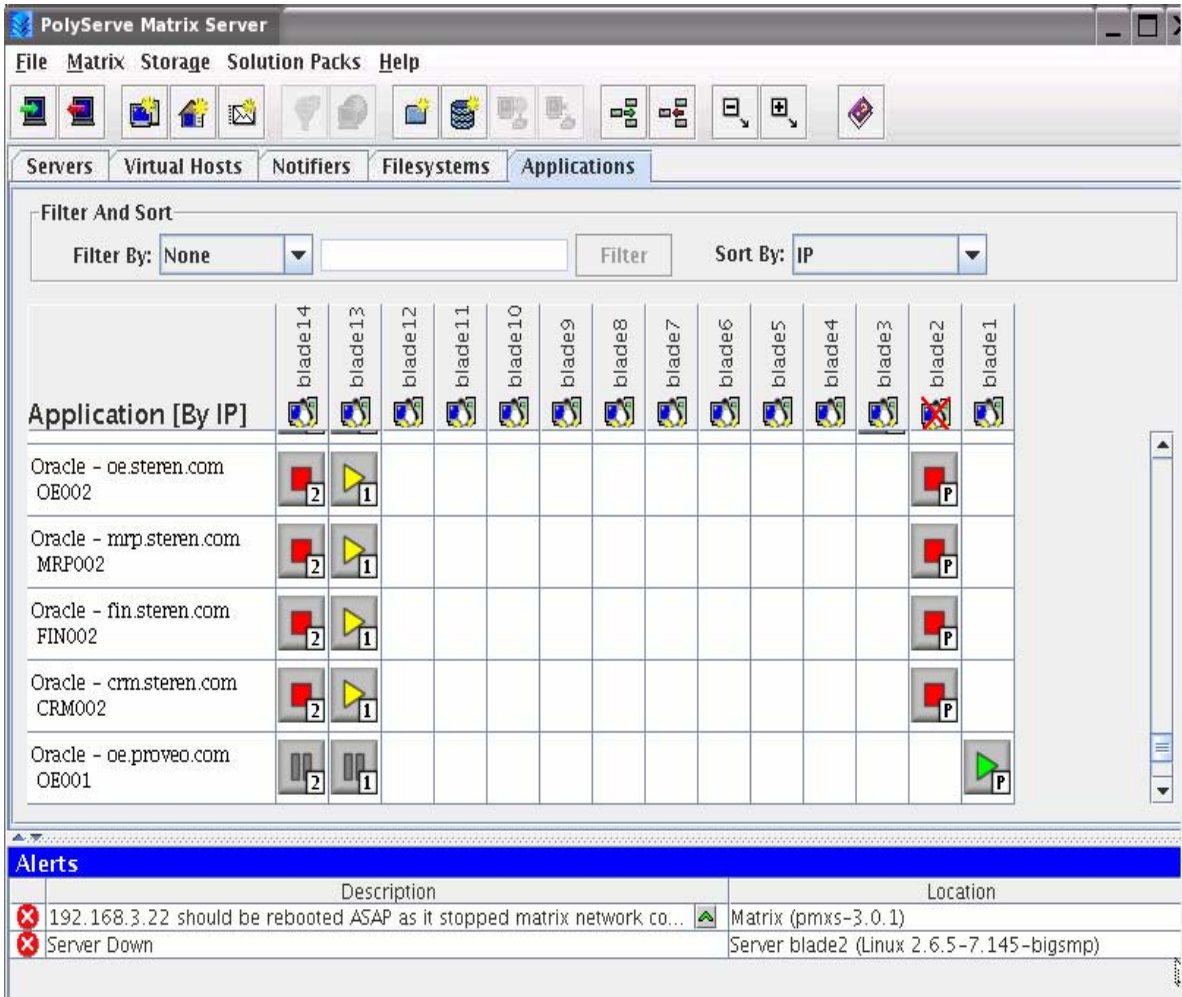
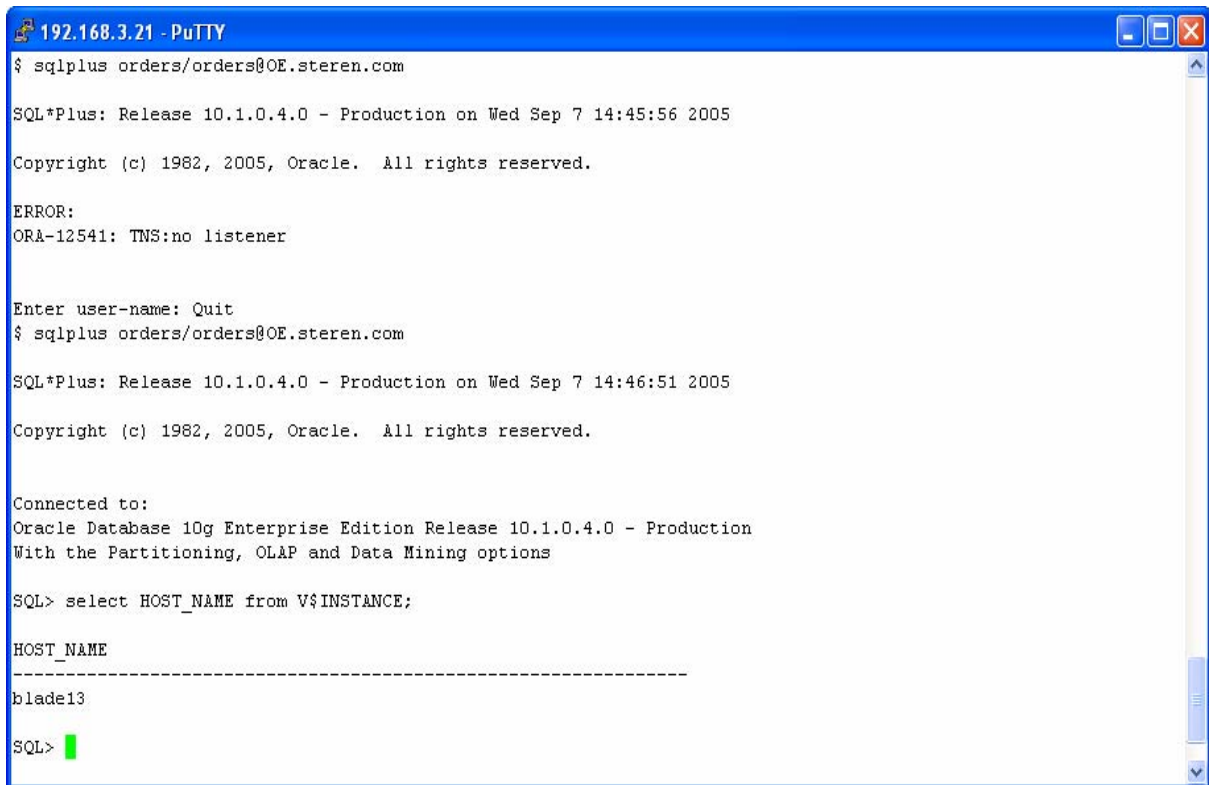


Figure 5.14: Management Console showing blade2 is no longer in the matrix and Company 2 databases are failing over

Figure 5.15 shows a connection attempt failing since the listener was not available; however, a subsequent connection attempt was satisfied. The session discovered that the instance was no longer running on blade2, but had failed over to blade13. The second connection attempt was not made until nearly a minute had elapsed.



```
192.168.3.21 - PuTTY
$ sqlplus orders/orders@OE.steren.com

SQL*Plus: Release 10.1.0.4.0 - Production on Wed Sep 7 14:45:56 2005

Copyright (c) 1982, 2005, Oracle. All rights reserved.

ERROR:
ORA-12541: TNS:no listener

Enter user-name: Quit
$ sqlplus orders/orders@OE.steren.com

SQL*Plus: Release 10.1.0.4.0 - Production on Wed Sep 7 14:46:51 2005

Copyright (c) 1982, 2005, Oracle. All rights reserved.

Connected to:
Oracle Database 10g Enterprise Edition Release 10.1.0.4.0 - Production
With the Partitioning, OLAP and Data Mining options

SQL> select HOST_NAME from V$INSTANCE;

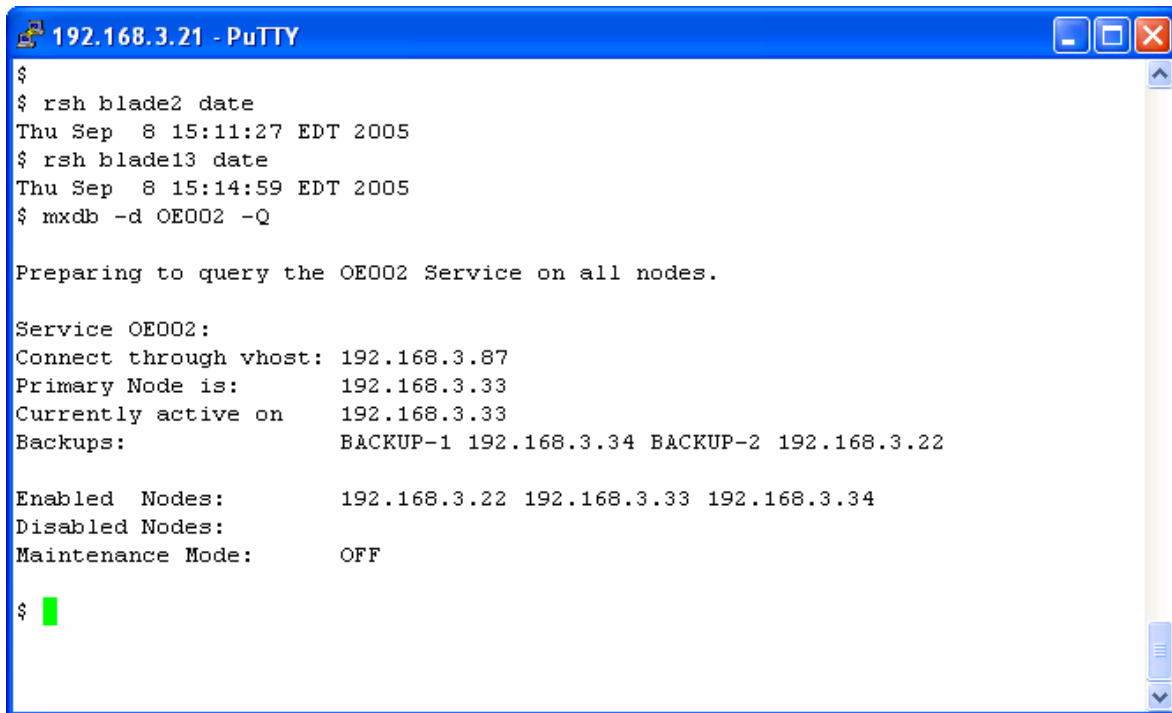
HOST_NAME
-----
blade13

SQL>
```

Figure 5.15: Successful connection to the database after the failover

A deeper look into the failover timeline is necessary. The primary resource for determining the events that occurred during a failover is the central MxDB-Oracle-HiAv log. No matter where a Virtual Oracle Service is hosted, all logging is central.

In the event of a failover, it is important to determine whether the wall clock times on the servers involved are synchronized. Figure 5.16 establishes that the service is up on blade13 and that there is a 3-minute 32-second wall clock time disparity between the test systems. This is an important fact when reconstructing such a timeline¹⁰.



```
192.168.3.21 - PuTTY
$
$ rsh blade2 date
Thu Sep  8 15:11:27 EDT 2005
$ rsh blade13 date
Thu Sep  8 15:14:59 EDT 2005
$ mxdb -d OE002 -Q

Preparing to query the OE002 Service on all nodes.

Service OE002:
Connect through vhost: 192.168.3.87
Primary Node is:      192.168.3.33
Currently active on   192.168.3.33
Backups:              BACKUP-1 192.168.3.34 BACKUP-2 192.168.3.22

Enabled Nodes:        192.168.3.22 192.168.3.33 192.168.3.34
Disabled Nodes:
Maintenance Mode:    OFF

$ █
```

Figure 5.16: Verifying the system times on the failover node and verifying that the service is back up

¹⁰ Production clusters are set up with network-based time synchronization, but it is always smart to check.

Moving to the pertinent section of the MxDB-Oracle-HiAv central log, it can be seen that blade2 was in the middle of a probe at 14:46:23 on blade2 (14:49:55 on blade13 time).

At 14:50:21, 26 seconds later, the Virtual Oracle Service was started on blade13. A cluster with 14 nodes and 60 databases was being monitored, and yet the filesystem and cluster state were recovered from the loss of a node sufficiently to fail over a Virtual Oracle Service in 26 seconds.

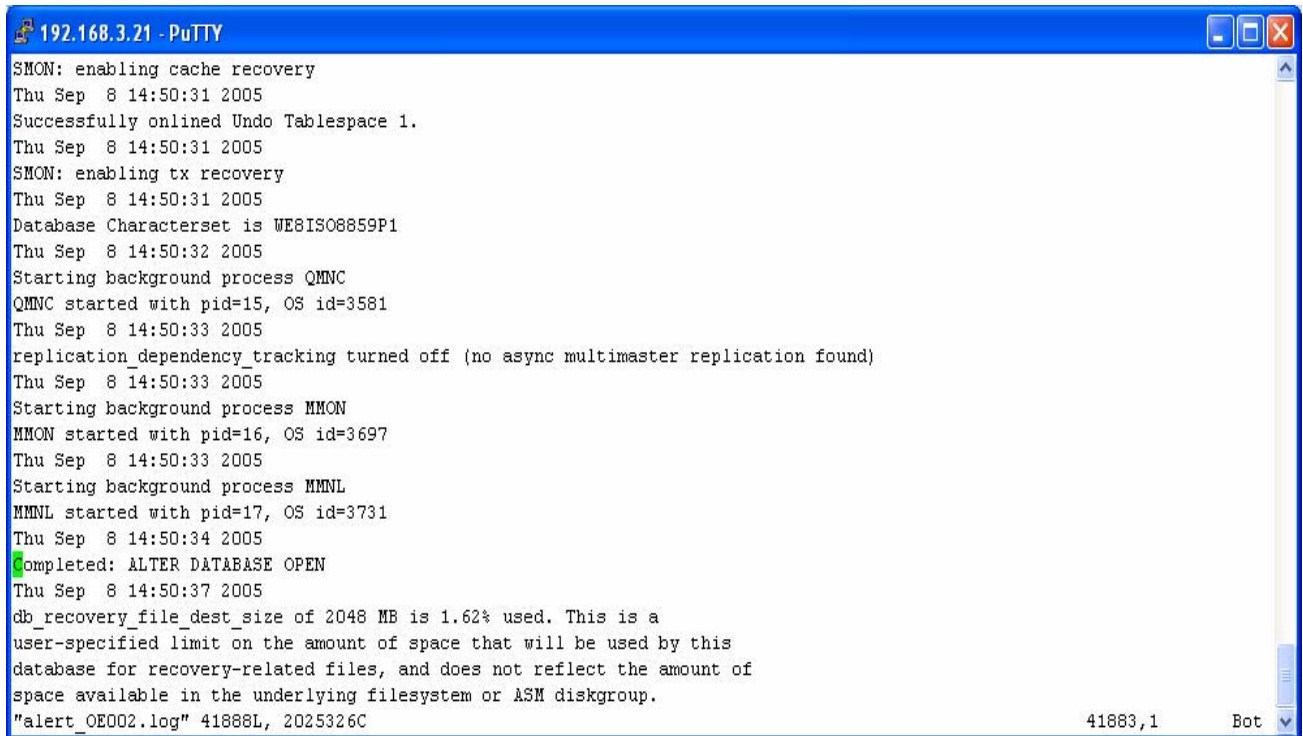
```

050908_144600 : blade2 : NOTIFY : 4967 : 6051 : lsnr.sh : verify_method: Start
050908_144601 : blade2 : NOTIFY : 4967 : 6051 : lsnr.sh : verify_listener: Start
050908_144604 : blade2 : NOTIFY : 4967 : 6051 : get_listener_password: Start
050908_144606 : blade2 : NOTIFY : 4967 : 6051 : verify_method: Listener LISTENER_OE002 is up
050908_144608 : blade2 : NOTIFY : 4965 : 4967 : verify_method: Database OE002: Probe Success. Probe duration 4 seconds
050908_144609 : blade2 : NOTIFY : 4965 : 4967 : verify_method: NORMAL END
=====
050908_144620 : blade2 : NOTIFY : 6907 : 6928 : verify_method: Probe of OE002 Instance started
050908_144623 : blade2 : NOTIFY : 6907 : 6928 : verify_method: calling oracle.sh verify
050908_145021 : blade13 : NOTIFY : 756 : 757 : startup_method: Service Monitor Startup for OE002
050908_145021 : blade13 : NOTIFY : 1483 : 1673 : lsnr.sh : start_method: Start
050908_145021 : blade13 : NOTIFY : 1483 : 1673 : lsnr.sh :start_listener: Start
050908_145021 : blade13 : NOTIFY : 1483 : 1673 : get_listener_password: Start
050908_145021 : blade13 : NOTIFY : 1483 : 1673 : start_method: LISTENER_OE002 started
050908_145022 : blade13 : NOTIFY : 1483 : 2086 : chk_for_critical_db_files: Start
050908_145022 : blade13 : NOTIFY : 1483 : 2086 : Calling verify routine to make sure the database isn't already running
050908_145023 : blade13 : NOTIFY : 1483 : 2086 : startup_database: Start
050908_145034 : blade13 : NOTIFY : 1483 : 2086 : startup_database: Database Instance OE002 is started
050908_145035 : blade13 : NOTIFY : 1483 : 2086 : do_startup: Startup of OE002 successful.
050908_145035 : blade13 : NOTIFY : 1483 : 2086 : do_startup: PMON data ( 2739 /u01/app/oracle/product/10.1.0 ) registered in /opt/polyserve/mxdb_oracle_ha/etc/.OE002
050908_145035 : blade13 : NOTIFY : 756 : 757 : startup_method: Changing the current host to PRIMARY if necessary
954,1 58%

```

Figure 5.17: MxDB-Oracle-HiAv central logging

The next place to look is the Oracle alert log. Although the database was formerly executing on blade2, the entries are available because this is a shared Oracle Home and OFA directory configuration. Unlike other High Availability solutions where the alert log from the failed system is unavailable on an internal drive in the dead server, all information is available in the same cluster filesystem as the Oracle Alert log. Figure 5.18 shows that the database achieved the fully open state by 14:50:34.



```
192.168.3.21 - PuTTY
SMON: enabling cache recovery
Thu Sep  8 14:50:31 2005
Successfully onlined Undo Tablespace 1.
Thu Sep  8 14:50:31 2005
SMON: enabling tx recovery
Thu Sep  8 14:50:31 2005
Database Characterset is WE8ISO8859P1
Thu Sep  8 14:50:32 2005
Starting background process QMNC
QMNC started with pid=15, OS id=3581
Thu Sep  8 14:50:33 2005
replication_dependency_tracking turned off (no async multimaster replication found)
Thu Sep  8 14:50:33 2005
Starting background process MMON
MMON started with pid=16, OS id=3697
Thu Sep  8 14:50:33 2005
Starting background process MMNL
MMNL started with pid=17, OS id=3731
Thu Sep  8 14:50:34 2005
Completed: ALTER DATABASE OPEN
Thu Sep  8 14:50:37 2005
db_recovery_file_dest_size of 2048 MB is 1.62% used. This is a
user-specified limit on the amount of space that will be used by this
database for recovery-related files, and does not reflect the amount of
space available in the underlying filesystem or ASM diskgroup.
>alert_OE002.log" 41888L, 2025326C
41883,1 Bot
```

Figure 5.18: Oracle Alert log in a CFS. Simple event reconstruction due to central logging

Total service outage for the Order Entry database, without any special tuning, was only 39 seconds. During these 39 seconds, modern applications would not have failed transactions. Most applications are browser-based and served by a middle tier. In the event of a failure such as this, the middle tier would simply reconnect and reissue the transaction. The user experience would have been a delay, but no transaction loss.

Summary

The Database Cluster architecture, with the IBM eServer BladeCenter and PolyServe Matrix Server, clearly enhances the value of Oracle10g in consolidation scenarios, providing:

- A means to consolidate and deploy multiple databases and associated applications in a single, easily managed cluster environment.
- Simplified management of large database clusters, made possible by the PolyServe Matrix Server product suite.
- Dynamic repurposing of server resources on demand to quickly and easily move processing capacity to where it is most needed.
- The ability to adopt improved hardware at the server and SAN level, made simple by the modular architecture.
- An autonomic, always-on operating environment with fast or even immediate self-healing and little or no performance degradation (and therefore increased utilization rates).
- Dramatic incremental TCO benefits from improved manageability, scalability, expandability, availability and asset utilization.

Appendix

ioDSS.sql

```
select sum(PHYRDS) reads,sum(PHYBLKRD * 16 )/1024 readMB,  
       sum(PHYWRTS) writes,sum(PHYBLKWRT * 16 )/1024 writeMB  
from dba_data_files,gv$filestat  
where dba_data_files.file_id = gv$filestat.file#;  
REM exit;
```





© IBM Corporation 2005

IBM Systems and Technology Group

Department 23U

Research Triangle Park, NC 27709

Produced in the USA.

10-05

All rights reserved.

Visit www.ibm.com/pc/safecomputing periodically for the latest information on safe and effective computing. Warranty Information: For a copy of applicable product warranties, write to: Warranty Information, P.O. Box 12195, RTP, NC 27709, Attn: Dept. JDJA/B203. IBM makes no representation or warranty regarding third-party products or services including those designated as ServerProven or ClusterProven.

IBM, the eight bar logo, the eServer logo, eServer, xSeries, BladeCenter, ServerProven, and TotalStorage are trademarks or registered trademarks of International Business Machines Corporation in the U.S. and other countries. For a list of additional IBM trademarks, please see <http://www.ibm.com/legal/copytrade.shtml>

Intel and Xeon are trademarks or registered trademarks of Intel Corporation.

Oracle, Oracle9i and Oracle10g are trademarks or registered trademarks of Oracle Corporation.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

PolyServe and the PolyServe logo are trademarks of PolyServe, Inc.

Other company, product, and service names may be trademarks or service marks of others.

IBM reserves the right to change specifications or other product information without notice. References in this publication to IBM products or services do not imply that IBM intends to make them available in all countries in which IBM operates. IBM PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING THE IMPLIED WARRANTIES OF

MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

This publication may contain links to third party sites that are not under the control of or maintained by IBM. Access to any such third party site is at the user's own risk and IBM is not responsible for the accuracy or reliability of any information, data, opinions, advice or statements made on these sites. IBM provides these links merely as a convenience and the inclusion of such links does not imply an endorsement.

This document is for informational purposes only and does not set forth any warranty, expressed or implied, concerning any software, software feature, or service offered or to be offered by PolyServe, Inc. PolyServe, Inc., reserves the right to make changes to this document at any time, without notice, and assumes no responsibility for its use. This informational document describes features that may not be currently available. Contact PolyServe corporate headquarters for information on feature and product availability. The PolyServe Matrix Server product uses software developed by Spread Concepts LLC for use in the Spread toolkit. For more information about Spread, see <http://www.spread.org>.