Very Many Database (VMDB) Clusters with IBM eServer BladeCenter and Oracle10*g*

A Summary 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 either 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.

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. The Flexible Database Cluster concept lowers administrative overhead and offers higher availability and on-demand scalability beyond that of several small clusters.

The prime ingredients of the Flexible Database Cluster 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. The extensive testing described in this white paper has validated the architecture and technology of the Flexible Database Cluster 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 reprovisioned 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.

Infrastructure for a Flexible Database Cluster

The PolyServe Matrix Server product and the MxDB-Oracle-HiAv Solution Pack were the core platform software technology used for testing the Flexible Database Cluster. 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.

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.

- A single Shared Oracle Home was configured for all 14 nodes.
- 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.
- 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 Manager[1] (ODM) was used for asynchronous I/O and improved clusterwide I/O monitoring.

Matrix Server Cluster Volume Manager

The filesystem that held 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).

Configuring cluster volumes is very simple using the PolyServe Management Console. The administrator simply chooses which disks the volume will be comprised of and specifies the internal label and stripe width. Once the volume is created, the Management Console is used to create and mount a filesystem.

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.

[1] 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

The MxODM I/O statistics package provides I/O performance information at a cluster-wide level (all databases in aggregate), database global level, instance, or node level. Because MxODM understands Oracle file, process, and I/O types, it offers specialized reporting that focuses on key Oracle ”subsystems” such as the Parallel Query Option (PQO), Log Writer, and Database Writer.

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 general health of the cluster is monitored by the core High Availability engine embedded in the Matrix Server product.

Proof of Concept

State-of-the-art and robust technologies were key to creating a suitable test system to prove the VMDB architecture. Figure 1 shows the cluster system components used for this proof of concept.

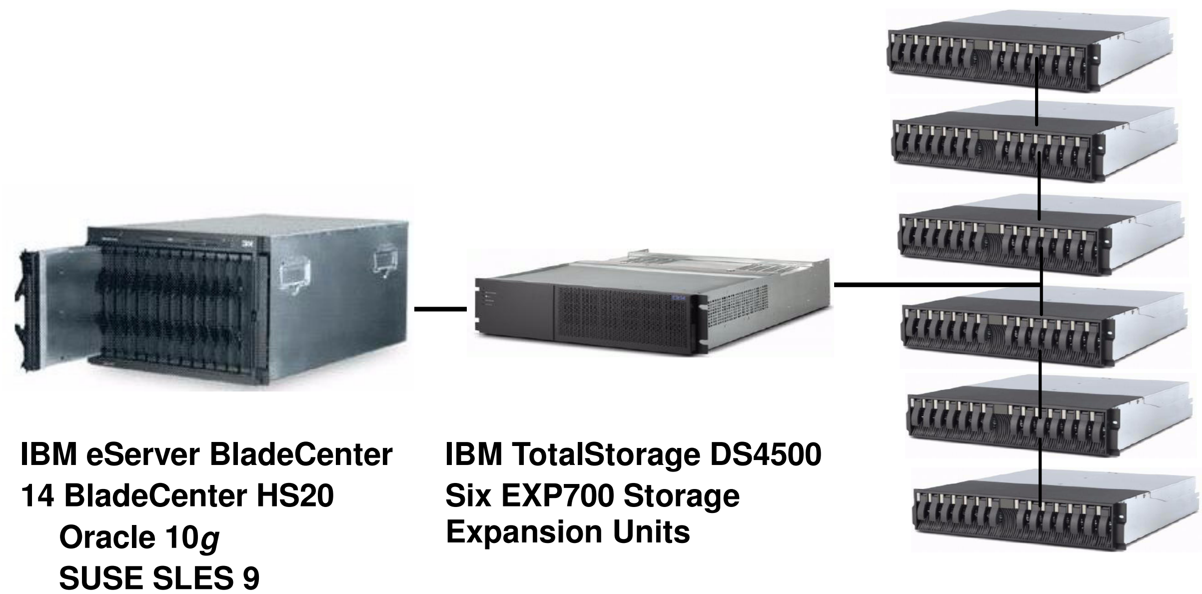

Figure 1: The proof-of-concept VMDB Database Cluster

Overview of the IBM eServer BladeCenter

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.
- Fourteen IBM eServer BladeCenter HS20 blades. These servers are high-throughput, two-way SMP-capable Xeon processor-based and 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.

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 customers[2].
- Financials (FIN).
- Customer Relationship Management (CRM).
- Manufacturing Resource Planning (MRP).

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 cluster filesystem. 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.

The model also followed traditional OFA convention. Combining OFA, OMF and the PolyServe product makes large numbers of databases quite simple to manage.

[2] Special thanks to James Morle of Scale Abilites, LTD for the use of the Order Entry kit (http://www.scaleabilities.com/).

Measurement Results

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, the 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.

The datafiles were created via a simple script that executed four dd(1) processes in parallel, each initializing 4GB. Because the target cluster filesystem was mounted with the PolyServe dboptimized parameter, all I/Os were direct I/O. The four processes achieved aggregate 128MB/s sequential write throughput. Considering the array-level cost of the mirror writes, this is considerable sequential write throughput for this storage configuration.

Random I/O Performance

The random I/O test suite consisted of three main tests. All tests performed 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.

The test showed that the SAN was able to deliver 90% scalability through 10 nodes as shown in Figure 2. At 10 nodes, the I/O rate was 86,956 4KB reads per second for 339 MB/s, which was 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.

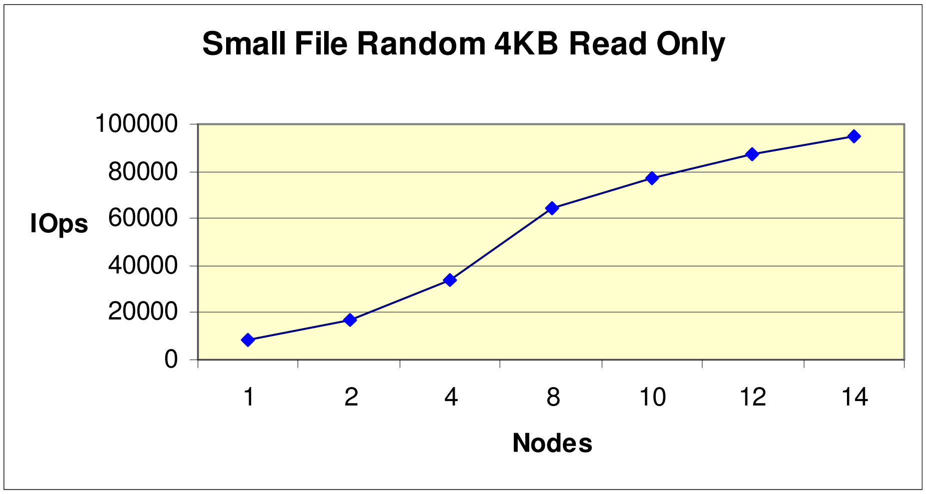

Figure 2: Rand IO Test 1. Small file random 4K physical transfers, read only

Randio Test 2: Single Stream, Multiple Node Scale-up, 25% Write, Small File. This test is the same as Randio Test 1 except that 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.

The test showed that the SAN 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 I/Os exceed 98 random writes per second. The effect of the physical disk configuration is seen in the scale curve in Figure 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 is to add disks to the array and then add them to the PolyServe dynamic volume and cluster file system.

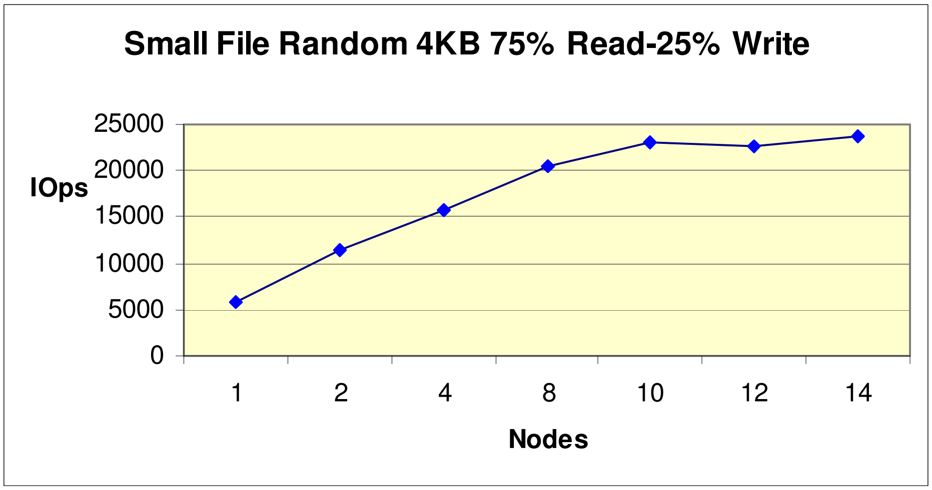

Figure 3: Rand IO Test 2. Small file random 4K physical transfers, 25% write

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 the test case was 128GB and resided in the cluster filesystem for direct I/O. The file was significantly larger than the array cache. With an array cache capacity of 2GB, a 128GB file will render the cache footprint less than 2% with this disk access pattern.

The test uses multiple execution streams per node—80, in fact. Figure 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 is 603 transfers per second.

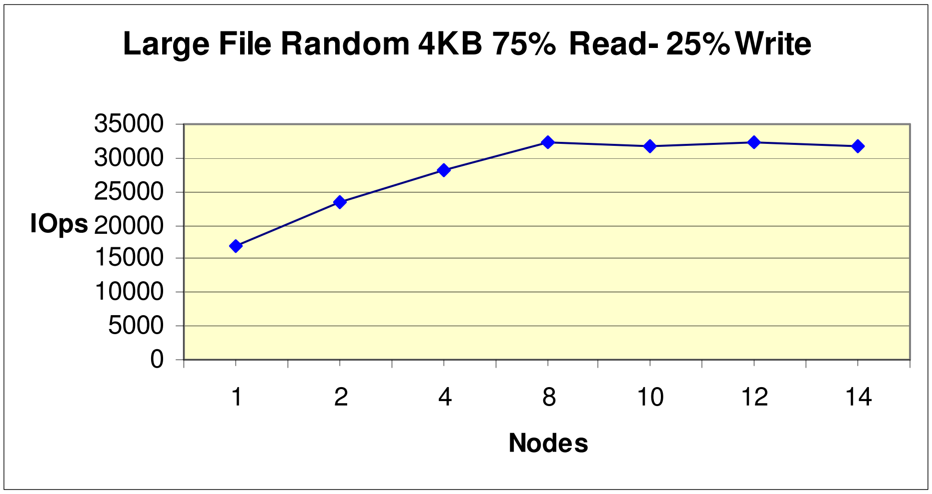

Figure 4: Large file random read/write test

Given these test results on this SAN configuration, the apparent optimum 4KB random I/O rate for real-life Oracle applications is 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 equates to 30,000 array-level I/Os per second or an average of 375 transfers per second per spindle. Results 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 simulate customer spending trend analysis. For the test, 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 degree of parallelism for the table was set to 16. The full table scan returned a count of 200,000,000 rows in 38.2 seconds. The scan throughput was 195MB/s with an average of 1,567 transfers of 128KB per second. This is close to the maximum theoretical throughput of the single active 2Gb FC host bus adaptor.

The DS4500 Performance Monitor aided in assessing how balanced the I/O was at the LUN level. During the full table scan test, the logical drives holding the databases were very evenly accessed due to the striping characteristics of the PolyServe cluster volume manager.

Index-Creation Results

In this test, both the table and the index reside in the card tablespace because there was sufficient space and a high-performance physical disk layout via the PolyServe cluster volume manager to accommodate both objects. In the test, the index-creation 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.

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? 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.

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

The Order Entry database for the Company A was stressed with 48 zero-think-time users. The Order Entry workload 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. The scenario was structured so that Company A was not getting serviced at the established service level. Where to begin?

The **mxodmstat** command provided with MxODM was used to report I/O only for the Company A and Company B instances broken out into read and write, synchronous and asynchronous. The instance for Company A was performing a steady stream of synchronous single-block reads along with a fairly consistent stream of asynchronous writes, a typical OLTP I/O profile. The instance for Company B was performing large amounts of asynchronous large reads, as well as significant bursts of large asynchronous writes. The two databases were hosted on the same server, but were in no way alike in terms of I/O profile. Further analysis was required.

The **mxodmstat** command was then used to drill down to the process types doing the I/O. We were able to determine that the Company A and Company B instances do not have a similar I/O profile and that, more importantly, the instance for Company B is involved in some Parallel Query activity such as index creation.

In the model, a short planned outage was then agreed upon to rehost the Company B databases to another less-utilized server. This is a win-win situation for both companies since both would have 100% dedicated server resources.

A simple MxDB-Oracle-HiAv CLI command was used to rehost the service to the first backup node. (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.

Although the service for Company B was hosted on a different blade, the users of Order Entry at Company B did not need to know that detail since they connect through the PolyServe Virtual Oracle Service.

Continuing with the exercise, **mxodmstat** was used again to monitor the activity generated by both companies after the rehosting. Originally the Order Entry instance for Company A peaked at a little over 1,000 physical reads per sample; however, after the instance 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. The PQO workload for Company B was reaching peaks of roughly 75MB's after being rehosted, whereas before it had 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 that advanced monitoring and simplified application rehosting improved manageability. The architecture proved to offer the necessary tools to perform such dynamic systems management.

After Company B was rehosted, statspack reports were checked for the expected physical I/O increase by the Company A database service. A simple **grep**(1) command showed that physical reads increased as much as 333% from the 700-775 range seen before the 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 whenever a new application is deployed. 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. During the proof of concept, a server failure was simulated by powering off a blade. A single company running four database instances was hosted on the blade. Although all four databases failed over, the purpose of the test was to analyze the outage effects on Order Entry during the blade failure.

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.

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 middle tier software. In the event of a failure such as this, the middle tier would simply reconnect and reissue the transaction. The user would have experienced a delay, but no transaction loss.

Summary

The Database Cluster architecture used for this proof of concept—the IBM eServer BladeCenter and PolyServe Matrix Server—clearly enhances the value of Oracle10*g* in consolidation scenarios and provides:

- 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 levels 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.

IBM Systems and Technology Group

Department 23U

Research Triangle Park, NC 27709

Produced in the USA.

10-05

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 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