ORACLE

OPTIMIZED SOLUTIONS

An Oracle White Paper
March 2011

Oracle Optimized Solution for WebLogic Suite: An Optimal In-Memory Data Grid Architecture

ORACLE

Introduction 2
Architecture Design Choices 4
Architecture Overview 6
Compute Nodes 6
Interconnects 8
Client and Initial Switching Tier 10
WebLogic Server Cluster Tier 11
Coherence Switching Layer 11
Coherence Tier 12
Database Tier 13
Oracle's Open Design Principles 14
Coherence Benchmark Testing 15
Workload Overview — On-line Hotel Room Searching and Reservation System 15
Microbenchmark: Coherence Simple Object Benchmark 16
Coherence Hotel Object 17
Benchmark: Hotel Application with Full Configuration 18
Best Practices 23
Conclusion 25
For More Information 26
Related Resources 26
About the Authors 27

Introduction

Oracle WebLogic Suite (WebLogic Suite) provides a way of creating in-memory data grids with features that integrate into existing application and service delivery infrastructures. This paper describes a tested combination of Oracle WebLogic Server (WebLogic Server) as Application Server , Oracle Coherence (Coherence) running as Data Grid with connecting technologies like Oracle TopLink (as the Java® Persistence Architecture, or JPA). This platform provides an in-memory data grid that offers numerous meta-services that help support and enhance the flow of data coming in from various sources. Coherence automatically and dynamically partitions data, optimizes the management of data based on the characteristics of information flow between applications and data sources, moves data closer to applications to reduce latency and improve performance, and offers a simple means to increase the capacity of shared data resources. Through these capabilities, Coherence helps maximize Web-facing application performance and can help foster near-linear scalability for many workloads. The capabilities of Coherence can also serve to off-load back-end resources, such as database servers and storage systems. Additional information regarding the capabilities of Coherence is available on the Oracle Technology Network site located at:
http://www.oracle.com/technology/products/coherence/index.html

Optimizing the benefits of a data grid built with Coherence relies upon creating a modular, balanced architecture. Matching the number of application servers, memory capacity, and quantity of processing nodes to the target workload characteristics is important to achieving the best performance.

This technical white paper discusses a drop-in implementation of Coherence running on Oracle's Sun servers utilizing Oracle Fusion Middleware components, most prominently WebLogic Suite. Benchmark test results are presented within this white paper to substantiate scalability and performance of Web-facing architectures. Best practices derived during implementation and testing of this configuration are also included. Organizations can take advantage of the pre-tested architecture, best practices, and performance results offered in this paper to help speed time to deployment and save on provisioning and implementation costs for similar or adaptable workloads.

Architecture Design Choices

To better understand the difference in the flow of application data with and without the use of in-memory caching technologies, it is important to consider that Coherence essentially adds a new tier to traditional Web-services architectures. In architectures without this added caching Tier, user experience is at times dictated by complex application programming residing on application servers, or in complex stored procedures within the database. Within this new tier, Coherence helps standardize and separate these procedures, and takes care of the "meta" issues around reliably and quickly accessing important data when a requesting client needs it.

Coherence as a technology can be run in many modes, and for many different use cases. The example described in this paper discusses a "Web Apps" architecture where Coherence helps accelerate transactions being received by Application Servers (WebLogic Server) in front of a database, as depicted in Figure 1:

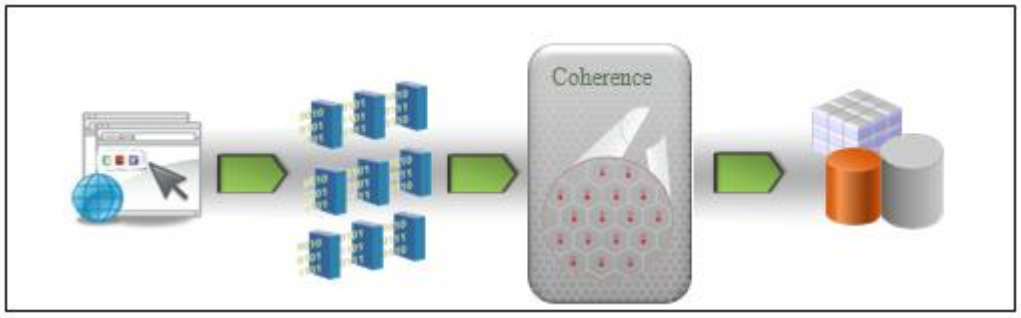

Figure 1 Web Apps

Coherence also runs in other modes that were not tested for the purposes of this paper. One such mode is Coherence's ability to act as a cache in front of mainframes to offload transactions that are billed in individual work units, saving significant costs. Coherence can either act as a front-end cache for an actual mainframe, or can be paired with Oracle Tuxedo to completely rehost a mainframe on open systems without the costs associated with mainframe use, see Figure 2.

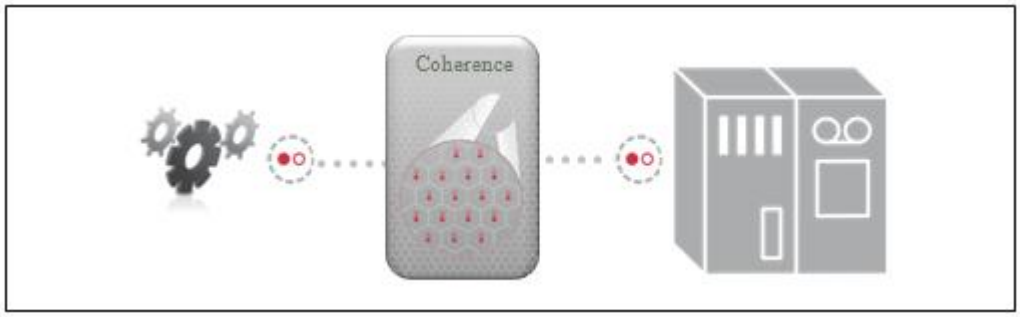

Figure 2 Mainframe Rehosting / Offloading with Coherence

The last use case for typical Coherence deployments is one that involves working on very large datasets both in real time as well as offline. Coherence is used to perform real-time analysis of fast changing datasets in many verticals, such as Financial Services as well as Telco markets, among others. Similarly, that extreme processing capability is also used to work through huge datasets in a non-real-time way, such as overnight processing of huge data, and can cut processing times for such use cases down from hours to minutes. See Figure 3 for a high level diagram describing this functionality.

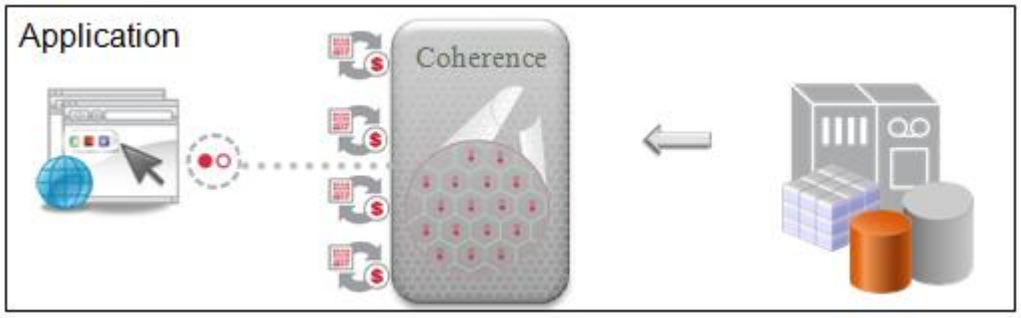

Figure 3 Extreme Transaction Processing / Batch Processing

This paper does not deal with the last two use cases discussed above. For more information on how to use Coherence for these uses, please contact Oracle Technical Sales for more information. The following sections offer a description of a modular, balanced, and pre-tested architecture that is optimized for WebLogic Suite Coherence deployments. An illustration of that architecture can be found in Figure 4.

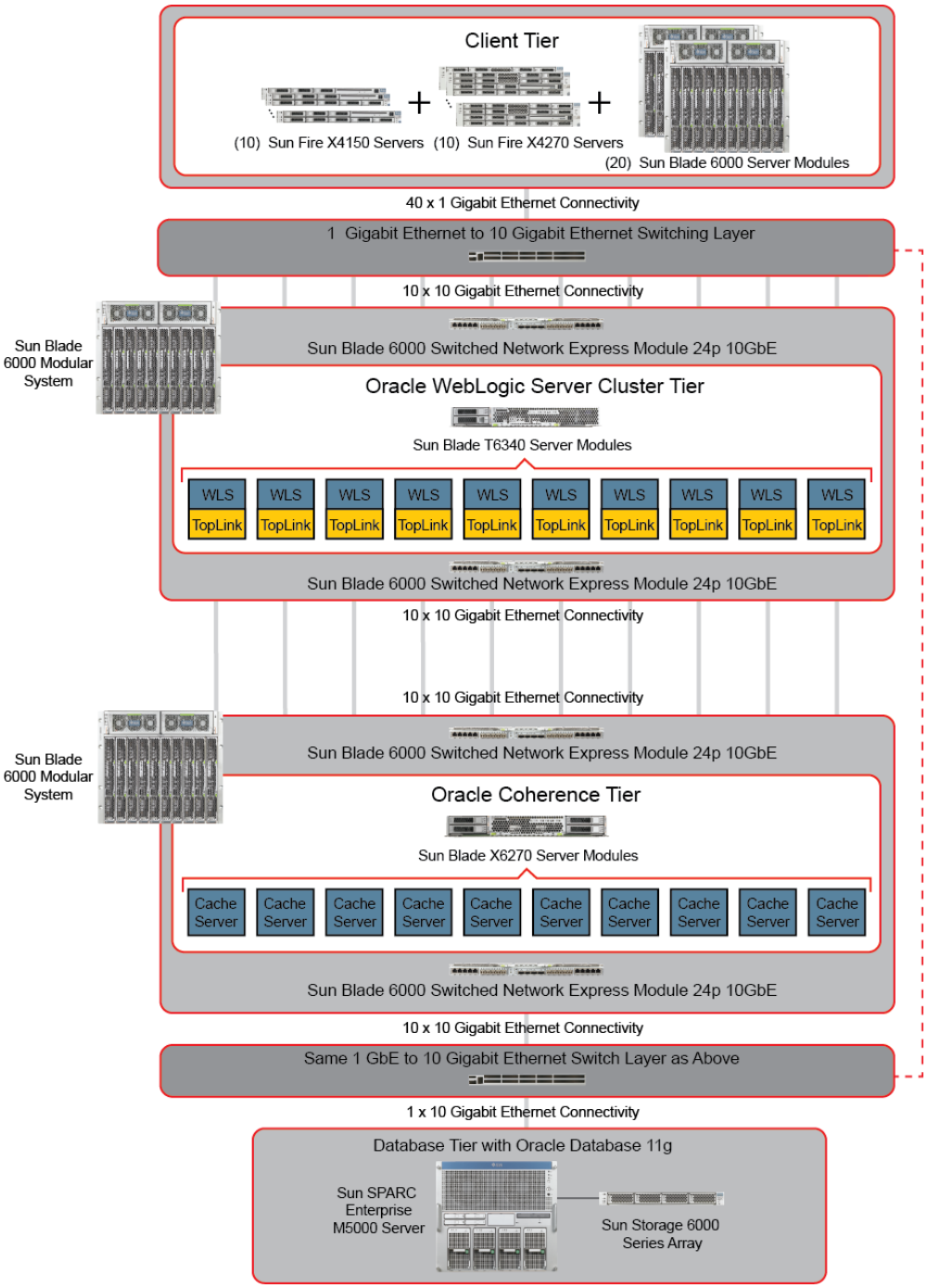

Figure 4 An Optimized Coherence Architecture

Architecture Overview

An architecture optimized for Coherence deployments was constructed and tested by Oracle engineers and consists of the following four basic layers:

Client Tier — A total of 40 systems, consisting of a combination of Oracle's Sun Fire rackmount servers and Oracle's Sun Blade server modules reside in the client tier and utilize the FABAN load generator to create a sizable workload. For more information on the open-source and extensible FABAN load generator, please see: *http://faban.sunsource.net*.

Oracle WebLogic Server Cluster Tier — Oracle's Sun Blade 6000 Modular System with ten Sun Blade T6340 server modules and Sun Blade 6000 Ethernet Switched NEM 24p 10GE supports the WebLogic Server Cluster tier. Each node functions as an application server and includes two eight-core 1.2 GHz UltraSPARC® T2 Plus processors and 64 GB of memory per T6340 blade serer.

Oracle Coherence (Grid Edition) Tier— A Sun Blade 6000 Modular System with ten Sun Blade X6270 server modules and Sun Blade 6000 Ethernet Switched NEM 24p 10GE from Oracle provides compute power for the Coherence tier. Each node contains two quad-core X6270 x86 processors running at 3.2GHz, 48 GB of memory per node, and executes the Java™ Virtual Machines[1] upon which Coherence runs.

Later iterations of testing substituted the Sun Blade X6275 M2 blade server in place of the X6270 server modules. The X6275 M2 server modules loses all internal hard-disks (in favor of 24GB flash-based storage), but in return doubles the CPU and memory density of the blade. The X6275 M2 Blade Module moves to Intel Xeon® 5600-Series processors (up from the Intel Xeon® 5500 Series in the original blades), and consists of two separate computing nodes in the space of one physical blade. As Coherence does not require much disk space at all, the X6275 M2 Blade Server Module is ideal for in-memory caching deployments because the density is doubled in the same footprint, comparing to previous blade server modules such as the X6270.[2]

Database Tier — The Oracle Database 11g is hosted on a Sun SPARC Enterprise M5000 server with 32 GB total memory and four 2.4GHz SPARC64® VII quad-core processors connected via four Fiber Channel (FC) connections to a Sun Storage 6000-class storage array.

Each of the individual tiers and components within those layers are discussed in the sections that follow. Note that all servers as tested ran the Oracle Solaris 10 Operating System. WebLogic Suite runs on either Solaris or Oracle Enterprise Linux.

---

[1] The terms "Java Virtual Machine" and "JVM" mean a Virtual Machine for the Java platform.

[2] In the Sun Blade 600 chassis, it is possible to fill all 10 slots with X6275 M2 blade server modules. Doing so will require both of the power supplies to supply power to the blades in the chassis, therefore degrading power redundancy to the chassis. With the 2.9GHz processor X6275 M2 blades, 8x blades can be in a chassis with fully redundant power. With the 2.4GHz blades, 9x can be in one chassis without losing power redundancy.

Compute Nodes

The descriptions in this technical white paper are intended to provide a sample implementation of WebLogic Suite that can be scaled or modified to suit specific organizational needs. The WebLogic Server and Coherence tiers each use a Sun Blade 6000 Modular System to facilitate tight integration of multiple servers. As the Sun Blade 6000 Modular System supports multiple processing architectures, this approach offers architects the flexibility to adjust the design as necessary to best match the job at hand. Utilizing the Sun Blade 6000 Modular System also simplifies administration of this solution by allowing a single platform technology to administer both tiers.

A Sun Blade 6000 Modular System is a 10 rack unit (10U) compact blade chassis supporting up to ten Sun Blade 6000 server modules, 2.7 Tb/sec maximum I/O throughput, up to 960 cores per rack, and up to 10.24 TB of memory per rack. The rear of the chassis offers two slots for Sun Blade 6000 Network Express Modules (NEMs) and up to 20 slots (two per server module) for PCI Express ExpressModules (EMs). An individual Sun Blade 6000 Modular System can be populated by Sun Blade 6000 server modules with a variety of CPU architectures in any order. These architectures include Intel® Xeon® processors, AMD Opteron™ processors, and Sun UltraSPARC T2 or T2 Plus processors with chip multithreading (CMT) technology. In many deployments, the Sun Blade 6000 Modular System chassis commonly holds a mix of server modules to perform different tasks. For example, Oracle T-Series processors are excellent at handling multi-threaded workloads, while the x64-based server modules are high-performing general purpose computing platforms for a wide variety of workloads.

The intent of using the Sun Blade 6000 Modular System is to show that the blade computing platform is one of the best options for a WebLogic Suite deployment. Using a Sun Blade 6000 Modular System within this architecture also allows the results presented in this paper to easily be inferred both up and down by adding or subtracting server modules to suit target performance and workload needs. However, requirements to utilize existing legacy servers or specific issues around integration with established compute and networking infrastructure may lead to the need to utilize traditional rackmount servers. One of the unique features of the Sun server platforms is "blade to server equivalency". For the most part, each of Oracle's rackmount server models has a direct counterpart in Sun Blade 6000 server module form. Unlike other vendors that have complex and changing rules for how to translate between traditional rackmount and blade modules, there are few functional differences between any given traditional Oracle rackmount server and the corresponding Sun Blade server module. As such, the results that are showcased in this paper can be easily applied to traditional rackmount servers — if those servers are a better fit for a particular datacenter environment.

With that said, there are some notable benefits that stem from utilizing the Sun Blade 6000 Modular System chassis for a Coherence deployment. The main benefits over rackmount servers are:

- Sun Blade Modular Systems offer up to a 10% savings in energy costs versus rackmount servers.

- By offering single chassis management for all ten server modules, Sun Blade Modular Systems help organization avoid the need to manage ten separate servers.
- The availability of a variety of Sun Blade 6000 NEMs and industry-standard PCI-e Express Modules (EM's) provide significant benefits related to interconnect technology cost and connectivity options.
- Support for mixing and matching multiple types of processor technologies within the Sun Blade 6000 Modular System chassis at any time and in any order, greatly enhances implementation flexibility.
- The I/O functionality enabled by the NEM form factor is an extension of the on-board capabilities of the server modules. Low-cost Fabric Expansion Modules (FEMs) attach to the server modules and extend the native networking capabilities of the server module motherboard to the NEMs.
- Management of the system, from bare metal hardware provisioning to patch management, automated software loads, and integration with on-the-fly virtualization technology is performed through Oracle Enterprise Manager Ops Center. The same software that is used to manage and administer Oracle Databases and Oracle Fusion Middleware is also used for management of the physical disks themselves. For more information on how to unify system to data management, please see the Enterprise Manager Ops Center webpage at: *http://www.*Oracle*.com/us/products/enterprise-manager/044497.html*

Interconnects

Client Tier: This sample architecture utilizes traditional 1 Gigabit Ethernet (1GbE) with individual Ethernet connections between the clients and application servers.

Server Tier: Interconnect technology plays an important role between each server tier. In particular, utilizing a high-bandwidth, low-latency interconnect between the WebLogic Server tier and Coherence tier is critical to reaching top performance. This sample application introduces a full featured integrated switching module Sun Blade 6000 Ethernet 24p 10GbE switched NEM which provides cut-thru switching capability for the low latency (as low as 300ns). Each Sun Blade 6000 Ethernet Switched NEM 24p 10GE delivers a 10Gb Ethernet connection to each server module installed in the chassis. Redundant 10 GbE connections to each server module can be provided by installing two NEMS into the Sun Blade 6000 Chassis. In order to offer non-blocking throughput, each NEM exposes a total of 14 external 10 GbE connections through the back panel: two SFP+ connectors and three 4x QSFP (quad SFP) connectors (Figure 5).

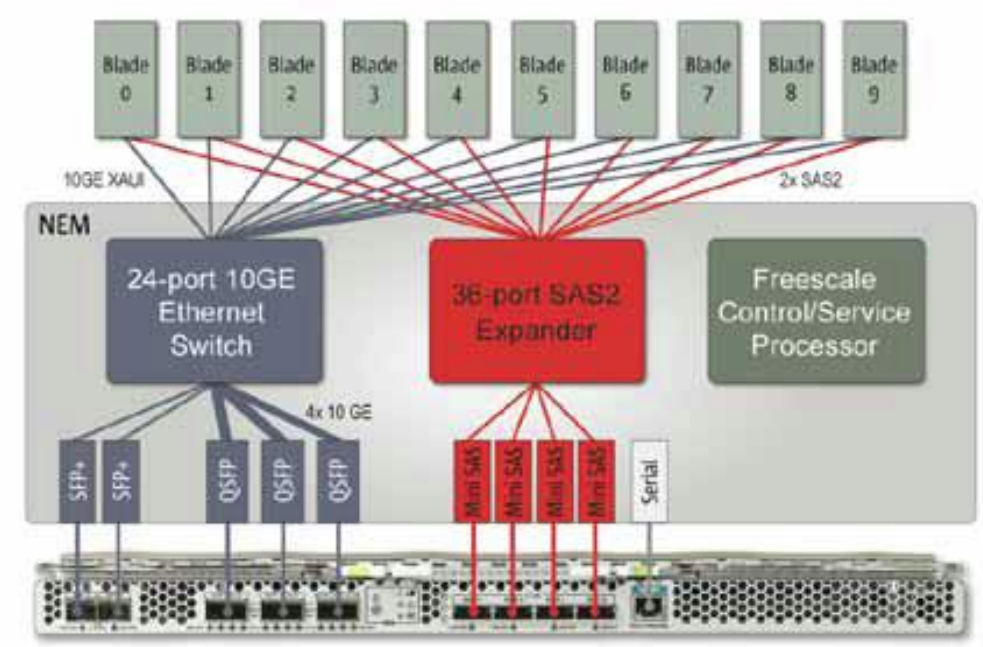

Figure 5 Switched NEM Detail

In addition to switch consolidation offered by the NEM, the 4x QSFP connectors also provide for considerable cable consolidation when connecting to a compatible rack or enterprise switch.

Consistent with Oracle's modular design principals, the NEM is easy to manage, and offers standard interfaces and network protocols, including:

- Unified chassis management
- A Web browser interface and a standard command-line interface (ILOM shell)
- Multiple user privileges
- Single sign on support for enterprise naming services
- ILOM support via the Chassis Monitoring Module
- Environmental monitoring
- An industry standard L2 / L3 network stack CLI and command set.

The server modules utilize the Sun Dual 10 GbE PCI-e 2.0 Fabric Expansion Module to connect to the NEM. The FEM (Figure 3) provide dual 10 GbE interfaces to the server module, and is pictured in Figure 6 below.

Figure 6 Fabric Expansion Module

NOTE: The use of InfiniBand® interconnects is also supported, but that interconnect technology was not tested in this configuration. As with any interconnect technology, the use of faster-speed networking between the application server tier and the datagrid tier is supported, without exposing a heterogeneous networking technology to the rest of a customer's datacenter.

Client and Initial Switching Tier

Coherence deployments serve intense workloads that include a high volume of simultaneous data requests to an application server cluster. In order to simulate this type of workload — with as much accuracy as possible — the test configuration required the ability to drive a significant load. To meet this need, the client tier consisted of a mix of 40 rackmount and blade server modules (Figure 7) working as load generators by running a Coherence benchmark on top of the FABAN load generator.

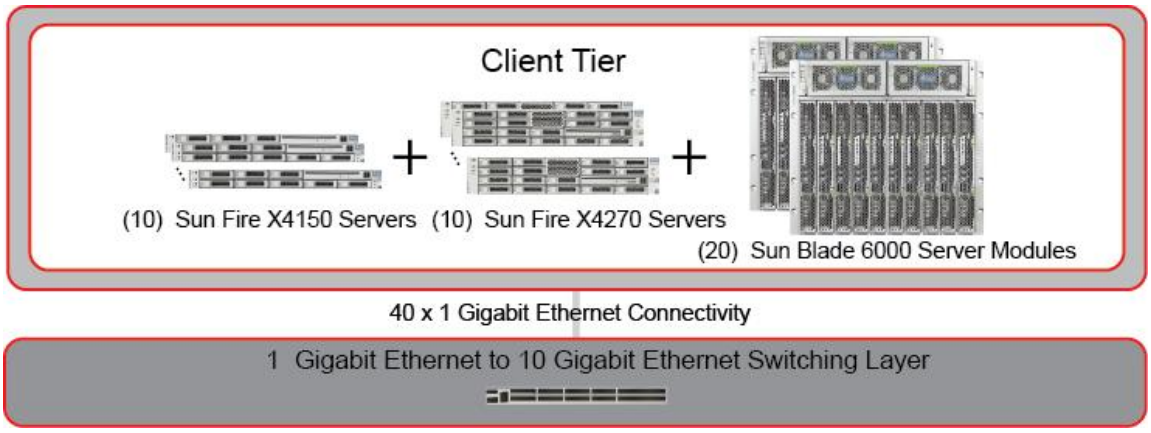

Figure 7 Client Tier

Though the specific configuration of the servers in the client tier is not important, each server is connected to the application server tier and by extension to the Coherence framework through 1 GbE connections. From there data flowed to an Ethernet switching layer that can bridge between 1 GbE networks and 10 GbE networks.

WebLogic Server Cluster Tier

Within the WebLogic Server tier, two Sun Blade 6000 Ethernet Switched NEM 24p 10GE handle the network connectivity. The NEM form factor is unique to the Sun Blade 6000 Modular System, and is designed to improve I/O connectivity to all the server modules in the chassis. The Sun Blade 6000 Modular System chassis provides a space that can house of a variety of different NEMs. The chassis supports either two single-height NEMs, or one dual-height NEM. Since the 10 GbE Ethernet switch NEM is single-height, this configuration includes two of these NEMs each providing 24 ports of 10Gbe switching capacity and a total of 280 Gb/sec bi-directional uplink bandwidth.

In this configuration, the two NEMs provides 10 GbE connectivity for each server module to the switch that connects the load-generating client systems in a redundant fashion by spreading the connections to each Tier among the two available NEMs (Figure 8). A second NEM shares a connection for each server module to the 10 GbE switch attached to the Coherence cluster. On each server module, the WebLogic Server instances communicate with the TopLink (Eclipse Link) layer, a Java Persistence Architecture (JPA) implementation for storing and accessing objects between (in this case) application server layers. Oracle TopLink allows for abstraction of the resources provided by the application servers to external customers in the case of application server failure.

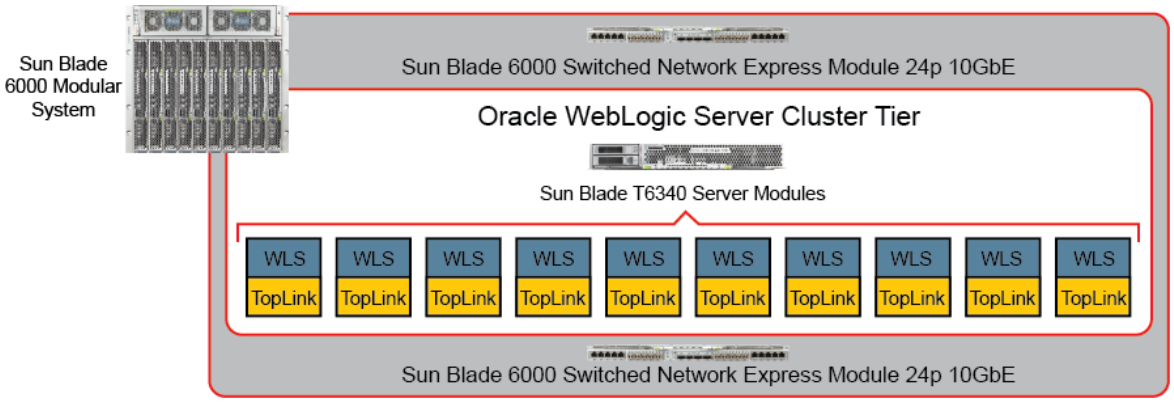

Figure 8 WebLogic Cluster Tier

Coherence Switching Layer

The client and application server tiers just described are generally familiar to those who work with application servers, or a similar persistent data layer that services large numbers of users. In such architectures, the application servers most often communicate directly to the database tier. Coherence adds another tier in the flow of data. In a Coherence architecture, this new layer essentially informs the application that the data it is requesting is local to the application server and not on a remote database. This is done through a small piece of connector software that resides in the JVM. This piece of code handles the translation of requests from language that the Application Servers understand to the serialized format that Coherence uses. In reality, the data each application server needs resides in main memory on a set of other servers that are connected to each other inside of a Coherence clustered grid. In order for the application servers to realize the fastest access to the data that they need (and that they believe is locally stored), it is important to connect all of the application servers to the Coherence cluster in the fastest manner possible. For that reason, all Coherence nodes are connected to each other through 10 GbE networking in this architecture.

In the overall architecture shown in Figure 1, there are two separate Sun Blade 6000 Modular Systems. One chassis houses the WebLogic Server instances, and the other chassis houses all of the Coherence servers. The two Sun Blade 6000 Ethernet Switched NEMs connects servers on both Sun Blade 6000 chassis using 3 QSFT cables (Figure 9).

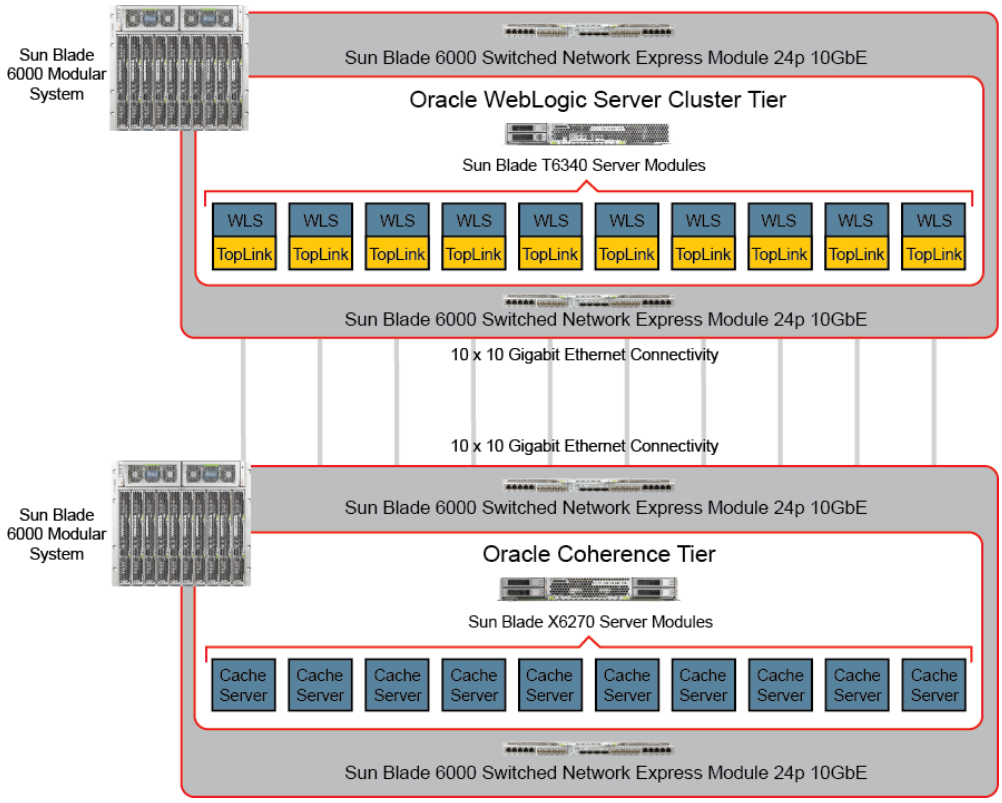

Figure 9 The Coherence switching layer utilizes a 10 Gigabit Ethernet switch to connect the application server tier to the Coherence tier. NOTE: It is entirely possible (and likely) that a separate Sun Blade 6000 chassis is not needed. The WebLogic application servers can co-reside with the Coherence nodes as long as there are N+1 Coherence nodes (where N = 2 or greater).

Coherence Tier

Memory speed, density, and CPU processing speed are the most important factors for Coherence performance and scalability. In this tier, the sample architecture utilizes ten Sun Blade X6270 server modules with Intel Xeon Processor 5500 Series CPUs (Figure 10). Based on a number of internal tests that compared various processing architectures, these systems offer the best balance between memory density, and CPU processor speed, and high performance for Coherence workloads running Enterprise 2.0 and Web 2.0 applications as the memory controller on these processors talks directly to its associated bank of memory first, before looking to a more distant (and therefore slower) resource. This automatic memory placement tuning is part of how Coherence handles memory to provide consistent response times.

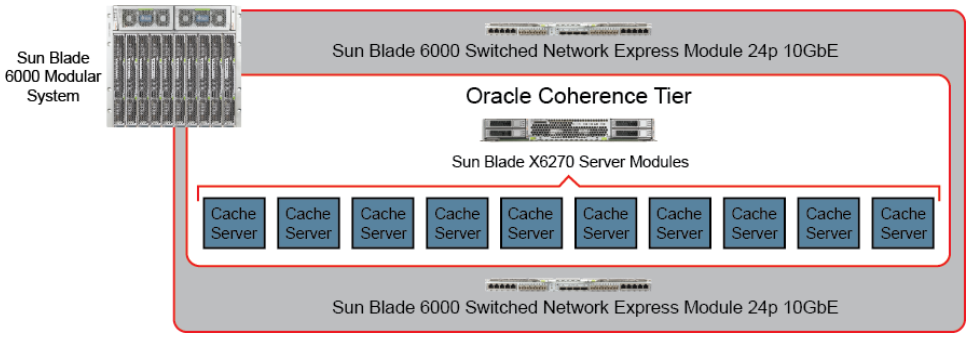

Figure 10 The Coherence tier benefits from the balanced architecture of the Sun Blade X6270 server modules.

Alternate Coherence Tier Hardware

As technology progresses, newer and more updated hardware is released. In December 2010 the Sun Blade X6275 M2 server (Figure 11 X6275 M2 Front View) (Figure 12) module was announced. This blade server model differs from the previously-used X6270 server module in a few important ways, relevant to Coherence deployments:

- Each physical X6275 M2 blade server module holds two compute "nodes". Each of these nodes is a fully independent server with its own operating system, CPU, disk and storage.
- The X6275 M2 blade server model moves to a faster and more densely architected x86 CPU (4 or 6 core Intel Xeon® 5600 series) but keeps the ability to have 2x CPU's per compute node. Thus, in one X6275 M2 Blade Module there are two separate compute systems, with a total of 2x the number of processors and 2x the amount of memory (96GB per node DDR3 1333 MHz) compared to previous blade modules.
- Each node has one flash-based boot disk, in a Flash Module (FMOD) form factor. The FMOD for each node has 24GB of ultra-high speed flash-based storage on which to load the Solaris Operating System. There is no redundant/mirrored storage capability in the X6275 M2 Blade server module. For Coherence deployments, though, that is acceptable as all data is backed up at least once on other nodes, providing for failure of any single node.[3]

[3] Note that even though the two compute nodes are independent, in the unlikely event that something happens to the physical blade (powering off, or some other blade-based service interruption), planning needs to be taken into account for the Coherence data to not be backed up to the other node sharing the same physical slot as any other node. This is part of typical Coherence redundancy planning to not have collocated backups on any servers that share a single point of failure. Placing the data on another X6275 M2 in the same or different chassis is completely acceptable.

- The X6275 M2 Blade Server Module has **only** 10GbE or 1GbE connectivity (based on which version is purchased) and that capability is part of the blade itself, not part of a separate Fabric Expansion Module.[4]

Internal testing of the X6275 M2 Blade Server Module showed an approximate 20% increase in single-node Coherence performance with a 25% increase in networking throughput due to the faster processor and the low latency FMOD disk storage native to the X6275 blade. The X6275 M2 Blade Server Module is **ideal** for deployments of Coherence due to its CPU and memory density, higher performance processors, and high performance flash-based boot disks. It is recommended that this product be used when Coherence is the only thing running on the blade module, as small local storage (24GB) is not sufficient for normal application server uses, but is just fine to hold the operating system and Java Virtual Machine binaries.

Figure 11 X6275 M2 Front View

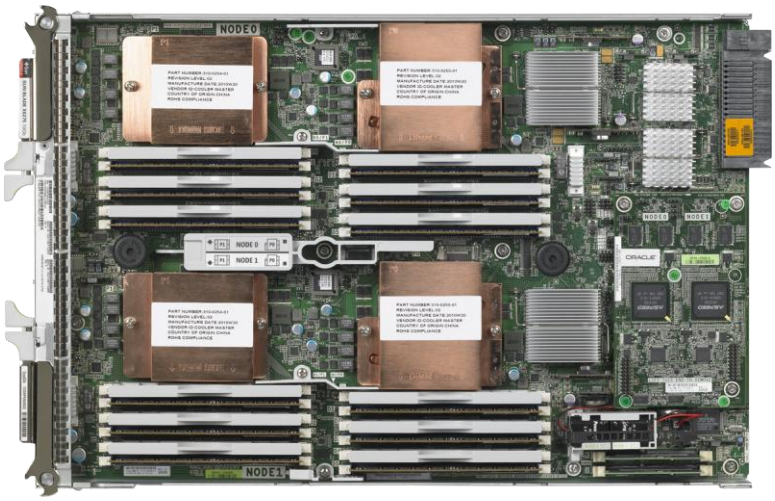

Figure 12 X6275 M2 Interior View

When switching to the X6275 M2 Blade Server Module the configuration does not change significantly, the only change being a high density of blade server modules in the Coherence caching Tier, and potential decrease of blade server modules, depending on processor power levels **(see footnote 2).** As shown in (Figure 13) the only change is in the Coherence Tier.

---

4 InfiniBand QDR is also supported through the addition of industry-standard ExpressModules.

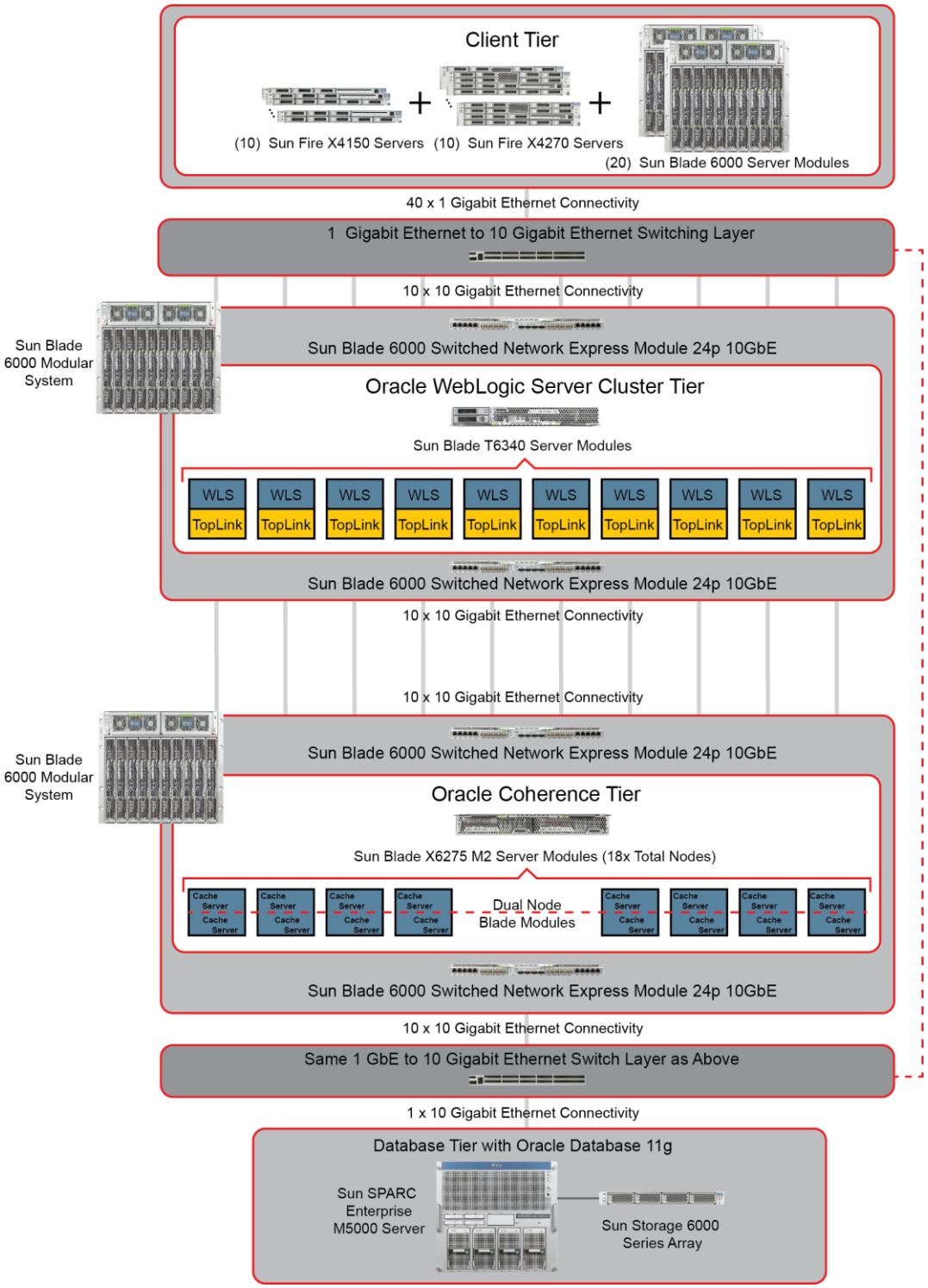

Figure 13 Architecture including X6275 M2 Blade Server Modules

Database Tier

For Enterprise 2.0 and Web 2.0 workloads, much of the data handled by Coherence resides primarily in the Coherence memory layer. Data that is not frequently accessed settles to the back-end database at a later time when fast memory-based access is not needed, decided through policy-based data backup and retention schedules. In these cases, memory density and networking speed between all nodes represent the key performance indicators. In addition, a standard Coherence deployment allows for redundancy and resiliency of all in-memory objects against single-instance failure through replication of the objects in other nodes. As a result, data housed in memory is safe to stay in memory and does not require immediate storage into a database.

For some data objects, however, Coherence requires access to external resources such as a database server or Network Attached Storage (NAS) device to get or store data. Coherence offers back-end architectural flexibility to support workloads that might require incoming data to be accepted and stored into a database before data storage is confirmed to the end user — such as financial transactions or HPC workloads. For these use cases, the back-end architecture can be easily modified to provide faster access to the database, depending on need. In fact, Coherence can be run in many different modes, including the following:

- Coherence running as Level-2 (L2) cache for the WebLogic Server instances.
- Coherence running as a read-only cache for data requested by the application servers. Writes go directly to the backend Database while that data is then pulled into the datagrid by the Coherence software for ultra-fast read-only access.
- Coherence running as a read/write cache for objects required by the application servers.
- Coherence hosting the entire business application in main memory.

The benchmark testing scenarios reported in this article ran Coherence as a read/write cache for objects required by the application servers.

For the benchmark tests described in this white paper, the transactions within the Coherence layer did not require immediate commitment and notification of committed write in the back-end database. Therefore, the interconnect speed to the database was not critical to performance in this configuration. As shown in Figure 14, the Database 11g instance running on the Sun SPARC Enterprise M5000 server is connected to the Coherence tier's Sun Blade 6000 Ethernet Switch NEM's SFP+ port via one 10 GbE add-in PCI Express card. The load on the database is not great, and since Coherence handles redundancy, the transactions that need to be stored in the database do not need to be immediately committed (For any given piece of data in the test case, one master copy and one backup copy existed on another physically separate server module running Coherence).

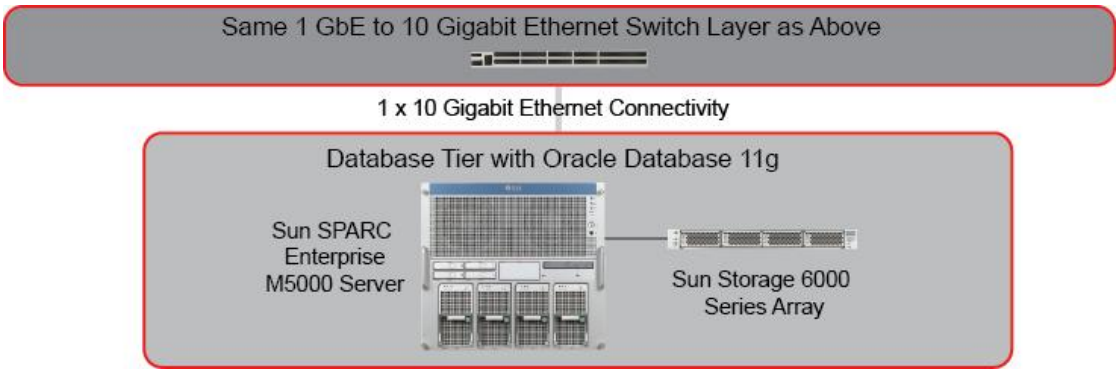

Figure 14 The minimal network bandwidth required at the database tier is handled by a single 10 GbE connection.

Coherence Benchmark Testing

In order to best illustrate the impact of hardware choices on Coherence deployments, Oracle engineers chose a workload that was easy to understand and document. As such, these benchmark tests simulate an online transaction and online search use case (Online Transaction Processing, OLTP). Test engineers for this project strived to create a workload that mimics an online, customer-facing hotel room search and reservation system. When considering the results from the testing, please bear in mind that this architecture may not be an exact match to every target deployment. However, many of the components can be exchanged to better fit alternative workloads.

Workload Overview — On-line Hotel Room Searching and Reservation System

The On-line Hotel Room Searching and Reservation System defines an online hotel room search and reservation system. Customers login to the fictional Web site through a Web browser interface, then perform a search, select a room to reserve, and then logout. This workload has been opensourced and placed on the Project Kenai site for download at: http://kenai.com/projects/coherencebench

User requests flow to a particular application server hosted on a Sun Blade 6000 server module, that in turn talks to the Coherence tier and if needed, the back-end database. Results are then passed back to the application server, and finally to the client system to produce results for the user to act upon. This workload represents a "typical" target implementation of Coherence for online Web and enterprise applications. As a side benefit, the workload provides an understandable and flexible deployment that demonstrates how optimal hardware choice affects scalability. Two workloads were created to test the scalability and performance of this reference architecture:

- Coherence Simple Object Microbenchmark — utilizes a 1 KB object size with GET and PUT operations designed to show raw Coherence performance with the smallest possible object and is designed to test raw platform and software performance without a typical user-workload being run on the hardware
- Online Hotel Reservation System Workload — the typical 10 KB packet size simulates the full hotel room search and reservation system utilizing the client systems, WebLogic Server tier, Coherence tier, and the back-end database.

Microbenchmark: Coherence Simple Object Benchmark

Coherence microbenchmark simply tests the scalability of Coherence by executing Cache GETs and PUTs of a small 1 KB object into and out of the Coherence tier. As shown in Figure 8, the scalability of the Sun servers and of Coherence is near-linear for the 1 KB object without using the application server tier. The graph of results in Figure 15 also reveals that the systems offered a steady rate of responsiveness throughout the test. At peak load during the microbenchmark test the CPU reached 95% utilization, demonstrating that equal scalability was achieved and there were no bottlenecks. The graph in Figure 16 shows that the network also provided scalable throughput and delivered consistent response times throughout the test. These microbenchmark results illustrate that the Coherence framework does not introduce measurable latency onto the raw hardware. These tests were a microbenchmark — initiated from the Coherence servers themselves without using the application server tier at all. The load for these tests was approximately 80% reads and 20% writes of the 1 KB object.

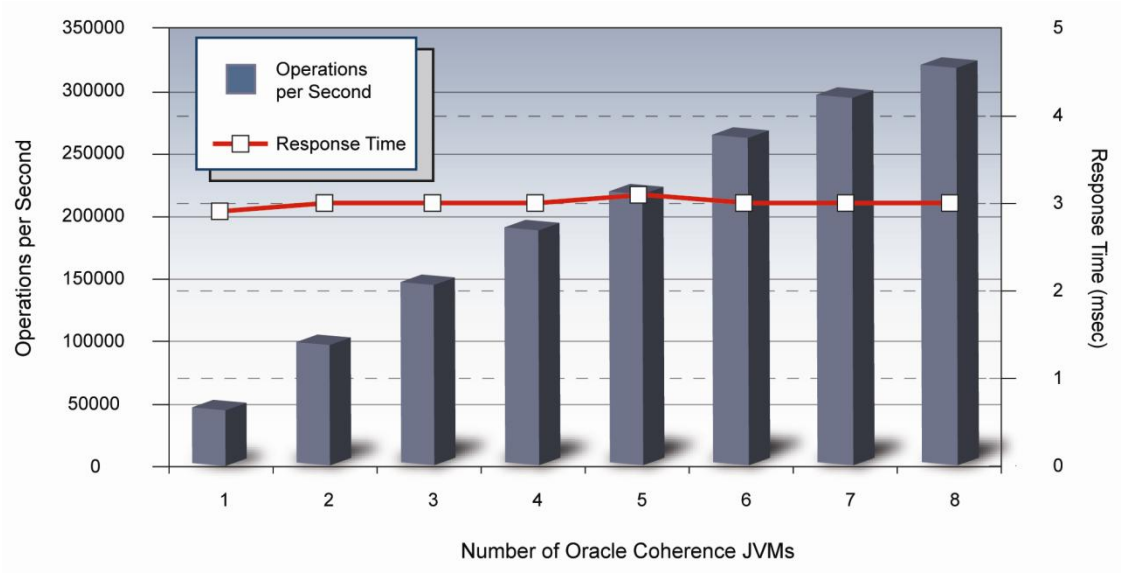

Figure 15 Microbenchmark test results reveal consistent response times and near-linear scalability of the test configuration in terms of operations per second, shown on 1 Coherence node.

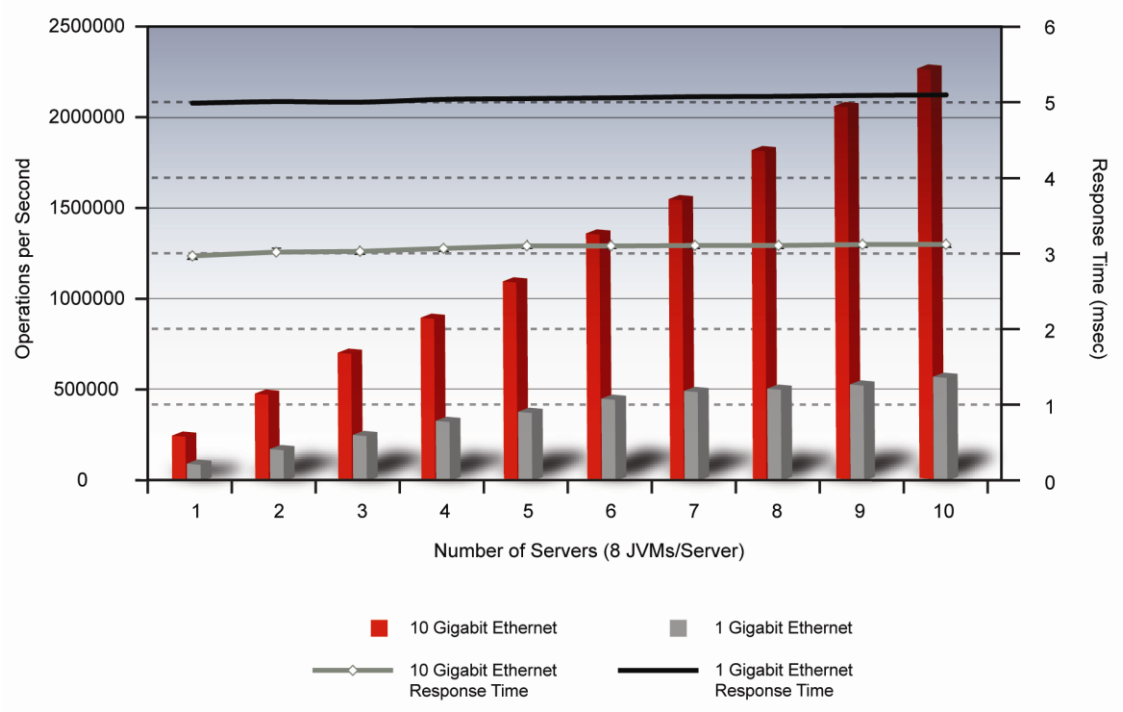

Figure 16 Network throughput scaled in a near-linear fashion and network response times remained consistent during microbenchmark test runs.

The microbenchmark test runs compared the performance of generic 10 Gigabit Ethernet cards and the Sun Blade 6000 Ethernet Switched 24p 10GbE NEM . Test results showed that the NEM network interfaces offered the same response times and levels of throughput as the generic 10 Gigabit Ethernet cards. This test was meant not only to gain a baseline for successive rounds of testing, but also to show that Sun NEM product performs on par with other 3rd party networking solutions, while offering manageability and power benefits over industry-standard add-in options.

As visible in the graphs, there is a significant performance increase moving from a 1 GbE interconnect to a 10 GbE interconnect and the Sun Blade 6000 Modular System does not introduce significant latency with the addition of server modules. From extensive testing it was also concluded that the optimal number of JVMs per Sun Blade X6270 server module reached maximum throughput at eight JVMs — or about one JVM per Intel Xeon processor core. More information on this finding is documented in the "Best Practices" section of this white paper.

Coherence Hotel Object

For the majority of Coherence deployments, the average object size is somewhere between 1 KB and 10 KB, with the majority being about 8 KB. The "hotel object" defined was 10 KB — a bit larger than a normal Coherence object. The Hotel Object contained text and numbers corresponding to what a typical hotel would want to broadcast about itself when a user searches for that hotel. The contents of that 10 KB object include:

- Hotel name
- Hotel address
- Hotel phone number and fax number
- Sample review of hotel

These fields add up to about 10 KB worth of data, and are roughly representative of the typical size of an online-search based application. Figure 17 shows how the Hotel Object is generated and flows through the Coherence architecture:

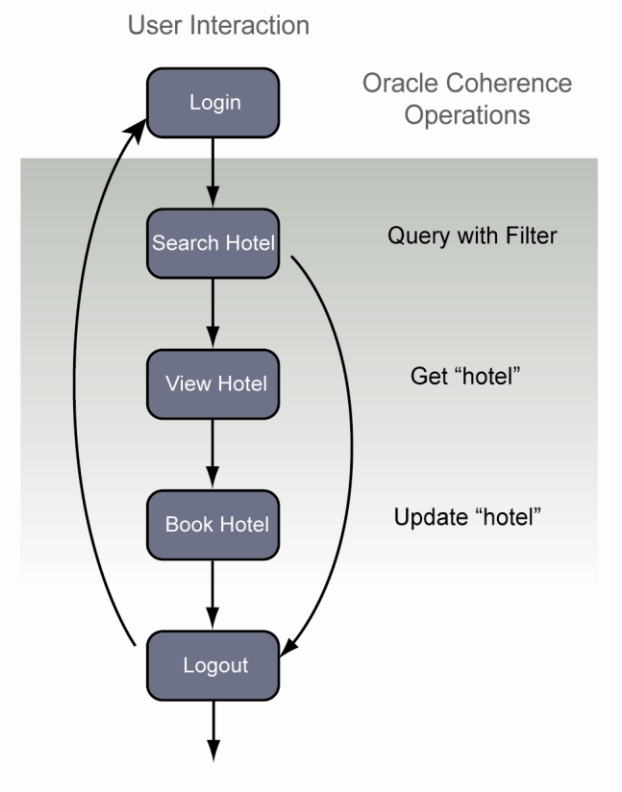

Figure 17 A flow chart demonstrating the steps that generate a Hotel Object

Benchmark: Hotel Application with Full Configuration

The hotel application benchmark tests the scalability of the entire configuration, from the client machines communicating with the application servers, through to the Coherence tier, on to the database tier asynchronously, and back. In order to understand how the test applies to the Coherence tier, it is important to understand how Coherence works. The Hotel Application uses the SpringSource framework and the Fusion Middleware stack to emulate an online hotel search and booking system (Figure 18). The simulated users communicate with the WebLogic Server instances which are running the TopLink JPA, which in turn transparently integrates with the Coherence framework. This use case makes the most sense for an online OLTP workload running Coherence as a read/write cache to achieve the best scaling and performance for online/enterprise applications such as the one modeled.

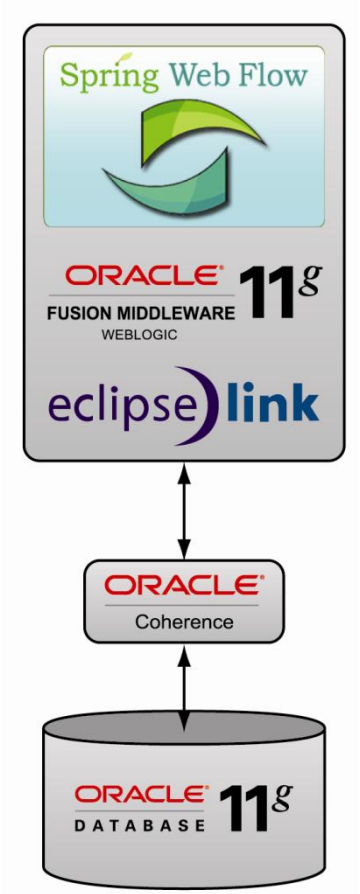

Figure 18 The Coherence scalable application architecture

As described, the Coherence testing involved a client layer provided by a mix of machines connected by a 1 GbE network communicating with the WebLogic Server. The WebLogic tier utilizes processors with Chip Multithreading (CMT) technology as engineers found the scalability of the UltraSPARC T2 Plus processors provided the best results for WebLogic Server out of any architectures tested. With up to 8x UltraSPARC cores, and eight threads per core (giving up to 64 hardware addressable threads per processor), multi-threaded Java applications like WebLogic Server perform exceptionally well on these processors. The WebLogic Server tier utilizes TopLink as the JPA provider as well as for communicating with Coherence on the back-end as the data repository for all objects. If any specific piece of data is not found in Coherence then the back-end Sun SPARC Enterprise M5000 server with the Database 11g is contacted for data retrieval.

The Coherence cache cluster warms the cache by batch loading data into the Coherence framework from the Database 11g running on the Sun SPARC Enterprise M5000 server — avoiding the need for each initial access from the clients to contact the database directly. Only when there is a cache miss will data be pulled from the database server and sent back out to the requesting client, otherwise batch

reads from the database are performed for commonly accessed data. In testing, cache misses did not happen frequently as the **Coherence tier cached most all data inside of the Coherence grid**, and only rarely did the database have to act as the originating server for any specific piece of data. In all instances, data in Coherence was replicated in one other place on another Coherence node for redundancy purposes**. This fact alone shows that database scalability can be greatly improved by adding the Coherence Cache Cluster to remove load from the database.**

Hotel Application Test Results

The test configuration provided near-linear scalability and consistent responsiveness during the Hotel Object benchmark testing (Figure 19). Notably, the test configuration scaled performance up to 1 million operations per second.

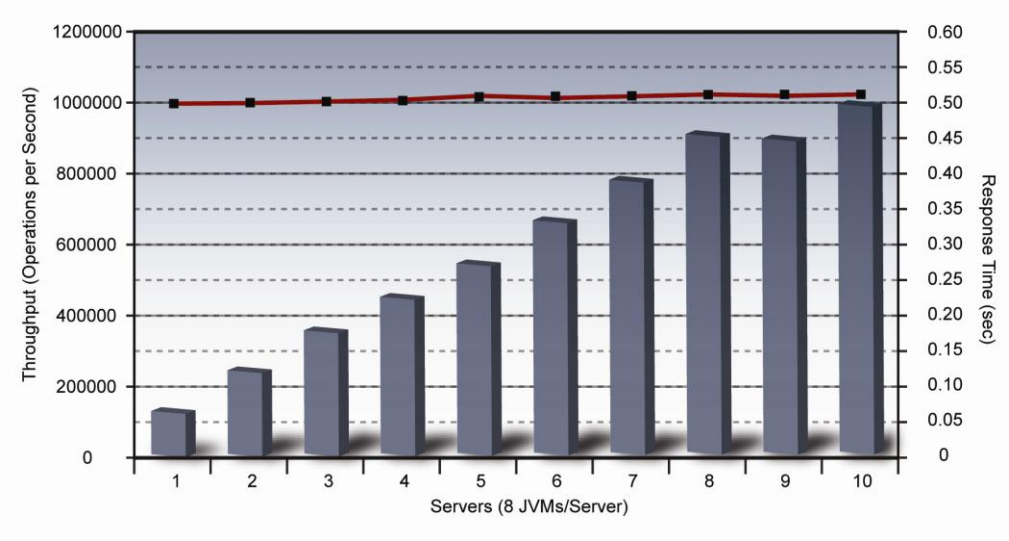

Figure 19 Performance and response time metrics for the Hotel Object Benchmark

Additional metrics gathered during the benchmark runs show that utilizing Coherence increases the number of users supported by the test configuration (Figure 20) and dramatically reduces CPU utilization on the database server (Figure 21).

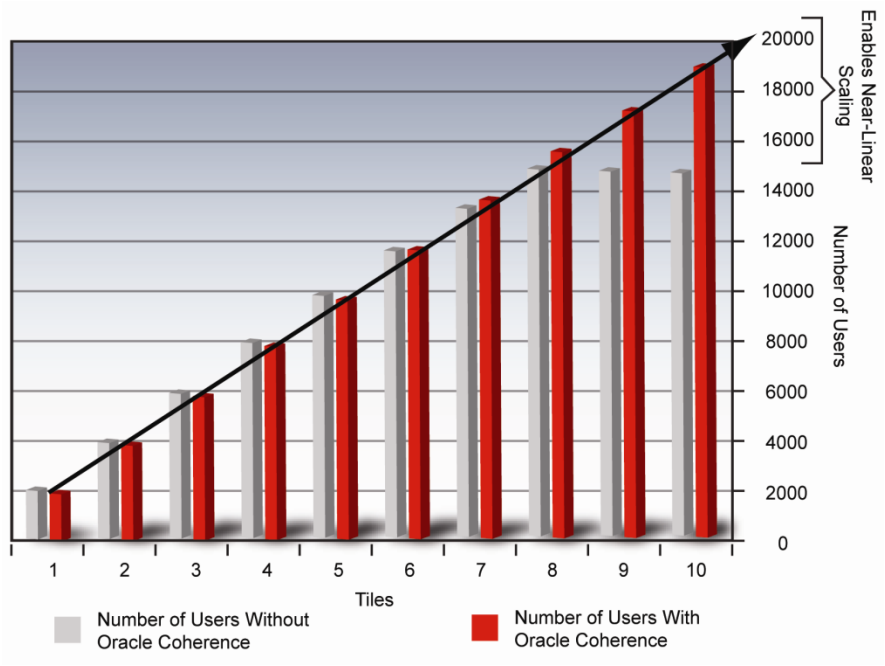

Figure 20 Utilizing Coherence for the Hotel Object benchmark increased the number of users supported by the test configuration. The term "tiles" refers to one WebLogic server node paired with one Oracle Coherence node, and the associated networking to connect them together.

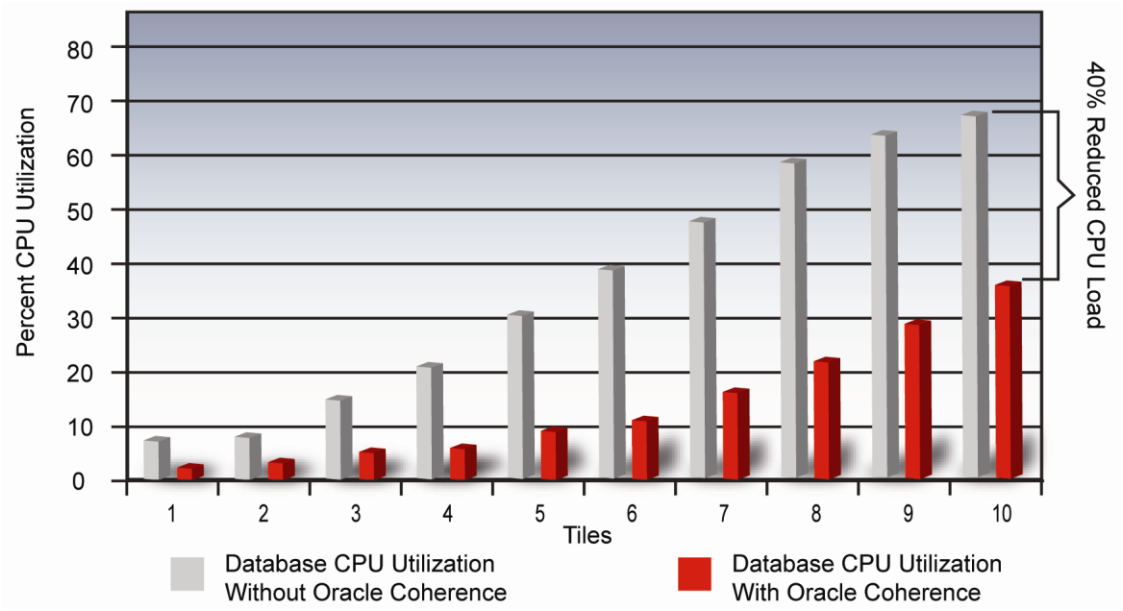

Figure 21 Utilizing Coherence for the Hotel Object benchmark reduced the workload on the database server. The term "tiles" refers to one WebLogic server node paired with one Coherence node, and the associated networking to connect them together.

Benchmark test runs also revealed that utilizing Coherence can lead to more consistent response times that can help organizations more easily meet service level agreements (Figure 22). Note that Coherence performance actually improves as more nodes come online, and that response time in these tests stayed the same or decreased as more nodes were brought online. Bringing more resources online will obviously lead to better performance but being able to judge specifically how the addition of hardware resources affects growth and performance curves is a critical piece to understand for datacenter planning as well as application provisioning.

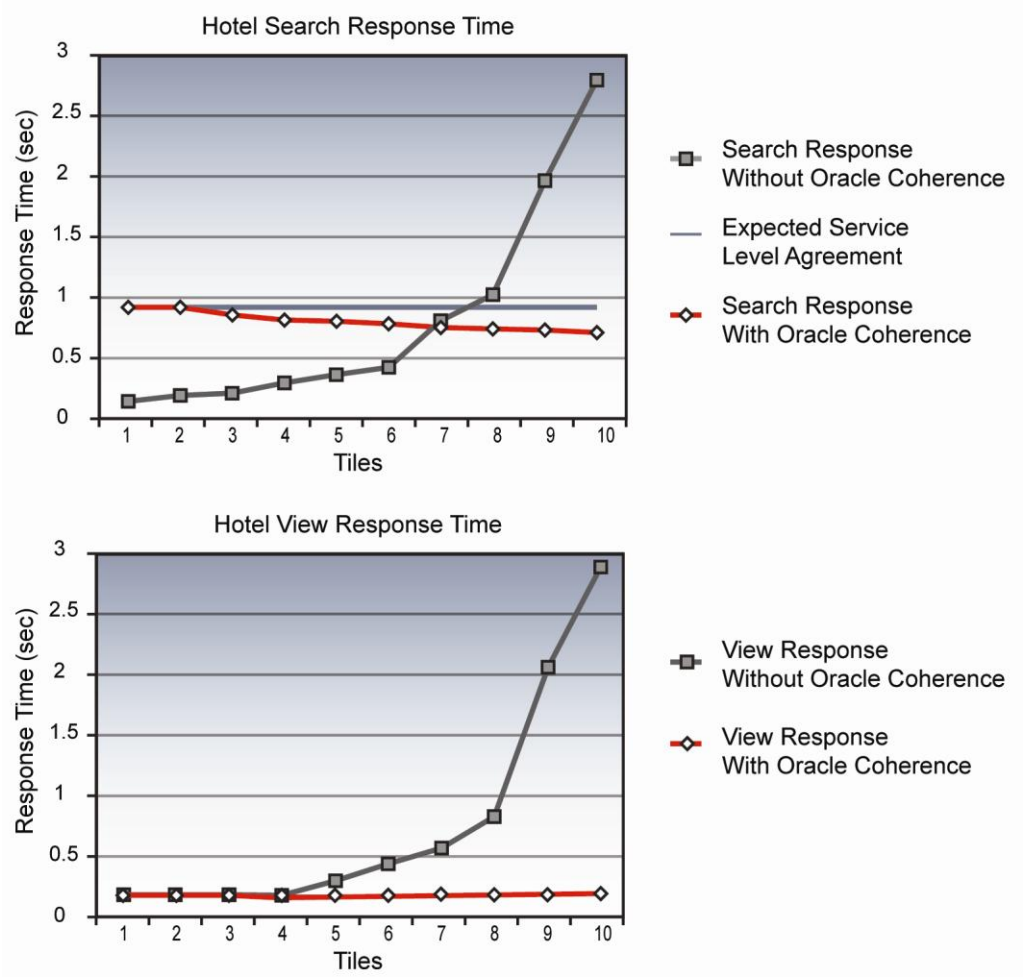

Figure 22 Utilizing Coherence for the Hotel Object benchmark supported better consistency in regards to response times. The term "tiles" refers to one WebLogic server node paired with one Coherence node, and the associated networking to connect them together.

As part of testing this configuration, Oracle engineers tuned various JVM parameters, and worked through performance and scalability related issues. By utilizing Oracle Solaris, engineers had access to a

powerful observability and tuning tool called Solaris Dynamic Tracing (DTrace). For more information on DTrace, please see: http://www.sun.com/bigadmin/content/dtrace/index.jsp.

Because DTrace provides hooks into the kernel of the operating system, test engineers were able to quickly identify performance areas that needed to be addressed. When using DTrace, Oracle engineers used these best practices to assist them in finding areas to improve performance, including:

- Starting the JVM with: -XExtendedProbes
- Looking at the Garbage Collection Probe for GC Intervals and Pauses
- Monitoring the Lock Activity
- Examining the Method entry/exit probe for frequent and expensive calls

By using these DTrace probes, test engineers were able to tune the JVM for optimal performance for this workload. DTrace also helped the test engineers focus tuning efforts by offering valuable insight into the amount of processing time spent on various aspects of application execution. The final breakdown of performance was as follows:

- Serialization: 33%
- Network Read/Write: 31%
- Garbage Collection: 26%
- LockWait for Put: variable from 5% to 20%
- InvocableMap Lookup: 5%

From the details given by DTrace, Oracle engineers were able to follow through and address any performance issues. Some additional tuning areas that helped improve results for similar workloads include:

Serialization — Consider using the Portable Object Format (POF) for serialization of the object data. Serialization and deserialization is the process by which intact objects are essentially "flattened" and "unflattened" for storing in the Coherence framework. Objects as a whole unit cannot be stored in the Coherence framework because storing complete objects is too complex and would very significantly affect performance. Therefore, the objects themselves need to be serialized for storing, then reassembled once the data is requested by a client and application server. The two main benefits of using POF include:

- About 20% CPU utilization reduction as compared to Java serialization
- Cache entry size reduced as compared to Java serialization

Network Performance — The online Coherence documents offer assistance in tuning a network to fit the size of the incoming requests by the application server tier. It is very important to **tune the UDP buffer size to the average object payload** in order to best match networking performance and expectations on both sides of the application server and Coherence tiers. Also, Oracle engineers

achieved approximately a 22% CPU reduction when using 10 GbE by using the Jumbo Frames features of the networking stack.

JVM Tuning — Some tuning of the JVM also yielded higher performance. Using the 1.6 version of JVM with large page support offered significant gains over earlier versions of the JVM, as well as using parallel or aggressive Garbage Collection techniques.

Network Interconnect Choice — During testing, engineers were able to drive the network and processing subsystems at nearly full capacity (around 95%). Since InfiniBand® can help reduce the CPU workload, replacing the 10 GbE networking with **Quad Data Rate (QDR)** InfiniBand® **offers the potential to reach even greater throughput levels**.

Conclusion

Creating an architecture that transfers the burden of scalability from the vertically-scaled database tier to the horizontally-scaled data grid can allow applications to reach new levels of scalability. Based on these benchmark results, organizations can gain the following benefits by utilizing a Coherence and Sun server solution:

Higher Performance

- The tested solution offers **up to 2.5 times faster eCommerce queries**[5][6][7] and transaction per second by adding Coherence versus running the same test without Coherence.
- Based on a workload similar to the benchmark test, applications can **scale to a million operations or more** per second by utilizing Coherence on Sun systems
- A Coherence deployment on Sun systems can help maximize ROI by delivering **near-linear scalability** while delivering to solid Quality of Service guarantees that can be passed on to customers
- By adding a Coherence tier, organizations can reduce CPU utilization in the database tier by up to **40% and support up to 25% more users**.

More advanced features

---

[5] Note: An e-commerce workload is considered to be 80% read & 20% write of Coherence operations for a typical 10 K object size

[6] Testing performed in this paper was run on X6270 blade server modules. Since publication, X6270 M2 versions of the blade servers were announced, with **general performance increases yielding approximately 2x that of the systems described in this paper**. Workload numbers may vary from application to application.

[7] Adding on to the increases for the X6270 M2 server modules, the X6275 M2 Blade Server Modules showed an increase in performance of another 20% in Coherence single node processing and up to 25% better networking throughput above the gains of the X6270 M2 while doubling the server density in the same throughput.

- DTrace offers **more advanced observability than any other tool**, and is included free in Solaris

Open Network Systems

- Sun Blade 6000 server modules from Oracle offer a **10% power savings over rackmount servers**, server module to rackmount server equivalency, networking flexibility through the use of industry-standard PCI Express ExpressModules, and Network Express Modules offering a choice of networking options.
- Oracle offers standalone Coherence and WebLogic Server installations with all tiers represented, or just a Coherence tier to an existing environment. In addition, Oracle has performed the work to help define small, medium, and large configurations for online, Web-based Coherence workloads.
- Through integration with Enterprise Manager, Ops Center provides an integrated platform for application-to-disk management. Ops Center works from the bare metal to Operating System provisioning and patching of both Linux and Solaris environments.

Sun T-series systems hold strong performance numbers running WebLogic Server, as reported to the Standard Performance Evaluation Corporation (SPEC®) organization at the following Web sites[8]:

- http://www.spec.org/osg/jAppServer2004/results/res2009q3/jAppServer2004-20090701-00135.html
- http://www.spec.org/osg/jAppServer2004/results/res2009q1/jAppServer2004-20090113-00127.html

The engineers at Oracle have architected this system to run Coherence at high levels of performance on Sun hardware. Details are available by visiting the main Web site for this architecture at: http://www.Oracle.com/us/products/middleware/coherence/index.html. The work that Oracle has performed is designed to save customers time in architecting a similar configuration. These results can be used to create an Coherence deployment that balances the Coherence software stack with appropriate Sun servers from Oracle.

For More Information

For more information on the benefit of Coherence and the benefits of Sun servers, please see the related resources below.

---

[8] SPEC and SPECjAppServer are registered trademarks of the Standard Performance Evaluation Corporation (SPEC). Please see www.spec.com for the latest results.

Sun Blade Systems
*http://www.*Oracle.*com/us/products/servers-storage/servers/blades*

Sun Networking Technologies
*http://www.*Oracle.*com/us/products/servers-storage/networking*

Sun JVM Tuning Guide
http://java.sun.com/performance/reference/whitepapers/tuning.html

Oracle Coherence
http://www.Oracle.com/goto/coherence

Oracle Coherence Technical Information on Oracle Technology Network
http://www.Oracle.com/technology/products/coherence

Oracle Optimized Solutions Homepage

http://my.oracle.com/goto/optimizedsolutions

About the Authors

Nick Kloski is a Principal Infrastructure Solutions Manager at Oracle. As a Sun employee for 13 years, Nick gained a wide exposure to both Sun and competitive systems, including systems administration work, over six years of technical Field Service, internal QA testing, and as a member of the Technical Marketing Department. In the role of Senior Infrastructure Solution Manager, Nick is responsible for educating customers on The Oracle Fusion Middleware software family, and how those products fit with Oracle's world-class hardware products.

Nitin Ramannavar is a part of the core Performance Engineering team at Oracle where his responsibilities include improving end-to-end system and application performance. Nitin specializes in Web technologies with emphasis on distributed computing, scale-out application architectures using virtualization, Web and application caching, and providing cloud services with guaranteed quality of service.

Satish Vanga serves as the technical lead for WebLogic Server, Coherence, and Oracle TimesTen In-Memory Database within the ISV Engineering group at Oracle. Satish has published several SPECjAppServer benchmarks and developed benchmarks for SugarCRM and Coherence. His expertise is in massively scalable systems designed with distributed caching engines such as Coherence, Terracotta, and TimesTen In-Memory Database.

Oracle is committed to developing practices and products that help protect the environment

Oracle Optimized Solution for
WebLogic Suite:
An Optimal In-Memory Data Grid
Architecture
March 2011, Version 1.1
Authors:
Nick Kloski
Nitin Ramannavar
Satish Vanga

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com

Copyright © 2011, Oracle and/or its affiliates. All rights reserved.
This document is provided for information purposes only and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle and Java® are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. UNIX is a registered trademark licensed through

Hardware and Software, Engineered to Work Together