An Oracle White Paper
July 2010

Oracle’s Sun Systems for Oracle Coherence: An Optimal In-Memory Data Grid Architecture
Introduction

Organizations across nearly every type of industry are experiencing a steep increase in the rate and volume of data creation and consumption. At the heart of this trend is the conversion from traditional client-server application models to external Web-based services. Applications that previously served hundreds of internal users for everything from shipment tracking to financial transactions are being transformed to support millions of consumers. As Web-based applications emerged, processing architectures met new scalability targets by embracing grid computing strategies that can distribute workload across a large number of systems. Now, data processing strategies must also evolve to support escalating data scalability and availability requirements.

Constructing a data grid can help organizations simplify data management strategies by partitioning data across multiple servers. Data grids provide structure for more efficient scalability and greater reliability while also speeding performance by caching data closer to the application. In addition, data grids offer massively parallel processing capabilities that applications can exploit to perform data searching, sorting, and aggregating functions.

The group of software components called Oracle WebLogic Suite (WebLogic Suite) provides a way of creating such an in-memory data grid with features that integrate into existing application and service delivery infrastructures. The 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) is known collectively as Oracle ActiveCache. 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. ActiveCache 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, ActiveCache helps maximize Web-facing application performance and can help foster near-linear scalability for many workloads. The capabilities of ActiveCache can also serve to off-load back-end resources, such as database servers and storage systems. Additional information regarding the capabilities of ActiveCache/Coherence is available on the Oracle Technology Network site located at:

Optimizing the benefits of a data grid built with ActiveCache 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 the impressive scalability and performance of this Web-facing architecture. 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

A sudden rise in popularity of an on-line application can leave organizations scrambling to rapidly expand processing capacity. In response to escalating throughput demands, infrastructure purchases are often made to rapidly scale the environment without thinking about the ability of the software to handle a ten, twenty, or hundred fold increase in the number of active users. Many times, a series of point fixes are applied to the application to address specific problem areas. Over time the code gets more and more complex and that complexity leads to scaling and reliability issues. ActiveCache is specifically designed to provide a more effective approach to application scalability.

To better understand the difference in the flow of application data with and without the use of ActiveCache, it is important to consider that ActiveCache essentially adds a new tier to traditional Web-services architectures. In architectures without ActiveCache, 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, ActiveCache 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. The following sections offer a description of a modular, balanced, and pre-tested architecture that is optimized for ActiveCache deployments. An illustration of that architecture can be found in Figure 1.
Figure 1. An optimized architecture for Coherence utilizing Sun systems
Architecture Overview

An architecture optimized for ActiveCache 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](http://faban.sunsource.net).

- **Oracle WebLogic Server Cluster Tier** — Oracle's Sun Blade 6000 Modular System with ten Sun Blade T6340 server modules 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 blade server.

- **Oracle Coherence (Grid Edition) Tier** — A Sun Blade 6000 Modular System with ten Sun Blade X6270 server modules from Oracle provides compute power for the Coherence tier. Each node contains two quad-core x86 processors running at 3.2GHz, 48 GB of memory per node, and executes the Java™ Virtual Machines¹ upon which Coherence runs.

- **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 SAS connections to a Sun Storage 5100 Flash Array from with 960 GB of flash storage.

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.

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 ActiveCache 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

---

¹ The terms “Java Virtual Machine” and “JVM” mean a Virtual Machine for the Java platform.
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 have 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 Express EMs 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:


Interconnects

Interconnect technology plays an important role between each 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 architecture utilizes a traditional approach, with individual Ethernet connections between the components and a mix of 1 Gigabit Ethernet (1 GbE) and 10 Gigabit Ethernet (10 GbE) connections.

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 2) working as load generators by running a Coherence benchmark on top of the FABAN load generator.

Figure 2. Client and initial switching tier for the Oracle Coherence test setup.

Though the specific configurations of the servers in the client tier is not important, each server 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. Switching layers are installed between each tier in the architecture.
The switching layers that reside between the client tier and the application server tier and between the Coherence tier and the Database 11g tier are similar. These switch layers can aggregate multiple 1 GbE connections to the 10 GbE connections utilized by the Sun Blade 6000 server modules. This layer typically already exists within a datacenter, and generally may just require the addition of 10 GbE technology in order to take advantage of the 10 GbE switching inside of the Coherence solution.

WebLogic Server Cluster Tier

Within the WebLogic Server tier, two Sun Blade 6000 10 GbE Multi-Fabric NEMs handle 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 a variety of different NEMs. The chassis supports either two single-height NEMs, or one dual-height NEM. Since the 10 GbE passthrough NEM is single-height, this configuration includes two of these NEMs, each providing ten 10 GbE ports, for a total of twenty 10 GbE ports.

In this configuration, one NEM provides 10 GbE connectivity for each server module to the switch that connects the load-generating client systems (Figure 3). A second NEM provides 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.

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 an architecture with Coherence, 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. Servers in both Sun Blade 6000 chassis are connected to an external switch via the pass-through NEMS installed in both chassis (Figure 4).

Figure 4. 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 5). 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.

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 6, the Database 11g instance running on the Sun SPARC Enterprise M5000 server is connected to the Coherence layer 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).

Oracle’s Open Design Principles

Oracle’s approach to system design results in the creation of interchangeable parts and technologies, allowing for hardware and software modification to suit specific needs. This Coherence deployment exemplifies the value of this open approach. Any technology in this setup can be interchanged easily. Although these tests use Solaris 10 Update 8 operating system on Sun Blade 6000 server modules with x86 and UltraSPARC T2 Plus processors, a choice of operating systems, server module architectures, memory densities, and NEMs are readily available.

Architectures that utilize Sun systems are not governed by strict adherence to a set of guidelines. Instead, Oracle acts as an integration partner and openly encourages the interchange of parts to help derive a solution that best fits project requirements. In addition, Oracle is unique among top-tier vendors in that all pieces of the architecture described here are provided by and supported by Oracle.
By adopting Sun systems, organizations gain the benefits of open system design and architecture flexibility, while maintaining a single point of contact for sales, service, and support for the established configuration.

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

Oracle engineers created a workload that has been open-sourced and placed on the Project Kenai site for download at: http://kenai.com/projects/coherencebench. This benchmark 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.

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. There are two tests that the Oracle engineers 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

Engineers wanted to test the scalability of Coherence running a microbenchmark that simply applies GETs and PUTs of a small 1 KB object into and out of the Coherence tier. As shown in Figure 7, 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 7 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 8 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 7. 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.
The microbenchmark test runs compared the performance of generic 10 Gigabit Ethernet cards and the 10 Gigabit Ethernet interfaces provided by the Blade 6000 Network Express Modules (NEMs). 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, Oracle engineers conclude 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” the Oracle engineers 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 9 shows how the Hotel Object is generated and flows through the Coherence architecture:

![Figure 9. A flow chart demonstrating the steps that generate a Hotel Object](image)

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 10). 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.

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 11). Notably, the test configuration scaled performance up to 1 million operations per second.

![Figure 11. Performance and response time metrics for the Hotel Object benchmark](image)

Additional metrics gathered during the benchmark runs show that utilizing Coherence increases the number of users supported by the test configuration (Figure 12) and dramatically reduces CPU utilization on the database server (Figure 13).
Figure 12. 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.
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 14). 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 13. 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.

Figure 14. 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.

Best Practices
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 due to InfiniBand® having less stringent CPU requirements compared to traditional TCP/IP processing overhead.

Conclusion

Sun servers paired with the scalability and resiliency of the Oracle Fusion Middleware stack — consisting of the WebLogic Server and Coherence software — are ideal matches for online enterprise and Web workloads. The superior scalability and modularity offered by Sun hardware platforms, coupled with the excellent software design of Coherence provides high levels of performance and flexibility to address demanding application workloads.

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 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 per second by utilizing Coherence on Sun systems
  • An Coherence deployment on Sun systems can help maximize ROI by delivering near-linear scalability with superior Quality of Service
  • 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

2 Note: An e-commerce workload is considered to be 80% read & 20% write of Coherence operations for a typical 10 K object size
• 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 many advantages including a 10% power savings over rackmount servers, server module to rackmount server equivalency, unmatched networking flexibility through the use of industry-standard PCI Express Express Modules, and innovative Network Express Modules offering a choice of networking options.

• Oracle offers the flexibility to deploy 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 through provisioning and operating system management of both Linux and Solaris environments all the way up to patch management and alert monitoring for the hardware.

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

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.

Related Resources

3SPEC and SPECjAppServer are registered trademarks of the Standard Performance Evaluation Corporation (SPEC). Please see www.spec.com for the latest results.
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.