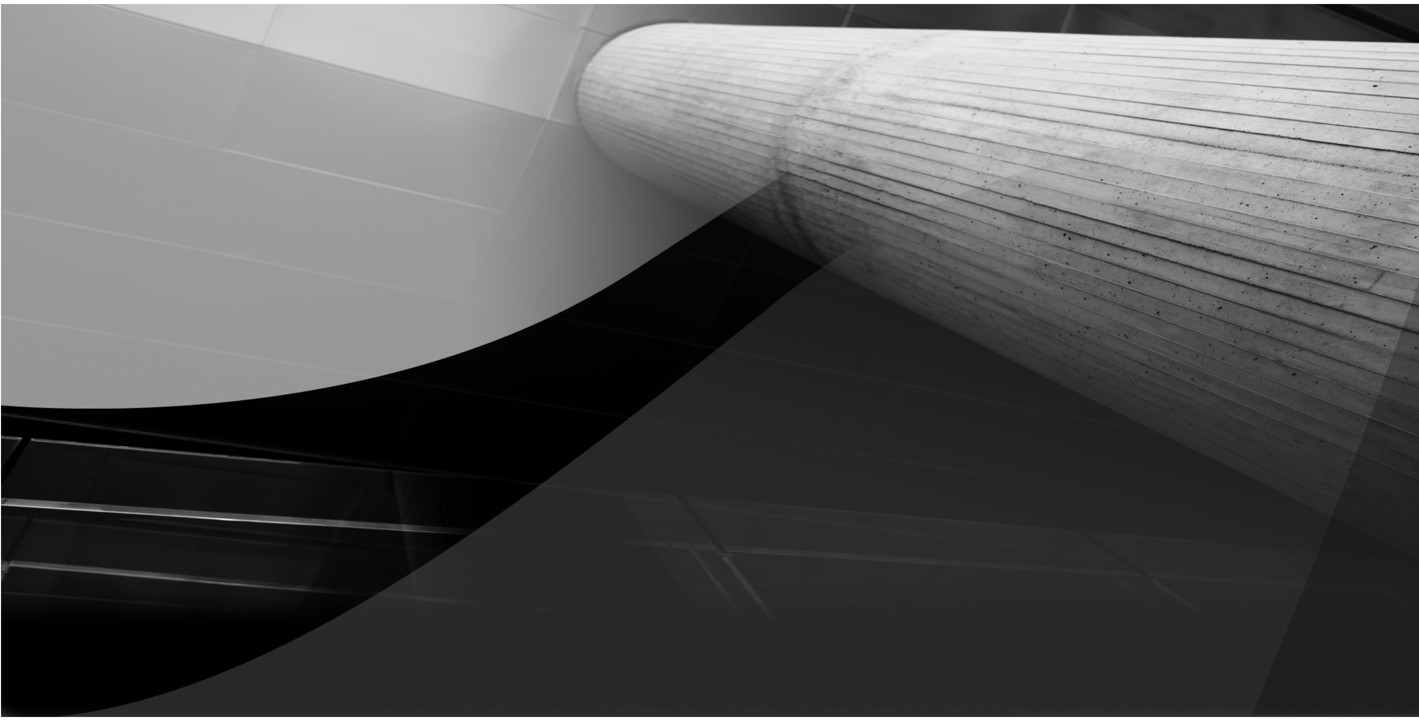

CHAPTER

1

Architecting the Foundation for Cloud Data Centers

Oracle Exalogic Elastic Cloud is Oracle's first engineered system for enterprise Java. Hardware and software are engineered together to optimize extreme Java performance. Exalogic is designed to revolutionize data center consolidation, enabling enterprises to bring together large numbers of performance-sensitive workloads into a single machine. Complex applications can be executed with results not achievable with the typical servers and complex software stacks used in today's data centers.

Cloud computing is an emerging *disruptive technology* (a new technology that unexpectedly displaces an established technology) that will be discussed at length in this chapter, particularly the value of engineered systems to cloud data centers. This chapter is also the first technical chapter in this book and serves as a technical introduction, as many of these topics are described in greater detail throughout the book.

Engineering Hardware and Software to Work Together

Once upon a time, two boys each received a gift.

The first boy opens his present and discovers a box containing a model car. The label on the box indicates that some assembly is required. The box contains plastic and wooden parts, several wheels, and metal axles. Tools, paint, and adhesives are not provided, and have to be obtained separately from other retailers. It will take time and effort for the boy to collect the necessary tools, and he will need to spend additional time and effort assembling the model car. It is possible that the model car will never be fully assembled—or even if it is fully assembled, it may lack functionality due to incorrect tools being used in the assembly process or inadequate instructions provided.

The second boy opens his present and discovers a box containing a preassembled model car. The second boy can immediately use the fully functional model car.

For many years, traditional information technology (IT) systems have been assembled using a process similar to the first model car example. Various layers of hardware and software components were obtained separately. In some cases, the complexity of integrating the acquired components was such that the purchased components were never used (giving rise to the term "shelfware"). Even when the components were assembled together to function correctly, the assembled solution often exhibited poor performance, chronic reliability problems, and major difficulties to manage and maintain. There are many reasons why a do-it-yourself platform may suffer from problems, including reasons due to project management, application development, unmanaged specification changes, and complexity.

In an effort to overcome these problems, the IT industry is moving toward appliances and engineered systems, which are preassembled systems that combine software and hardware that is optimized, packaged, tested, and deployed together. These systems arrive preconfigured and fully assembled, and they can be placed into operation in a single day. Engineered systems are necessary to overcome the problems of the past. These systems are also necessary to overcome the problems of the future, and the ever increasing amount of information that must be managed.

Furthermore, a common centralized platform for hosting software services provides businesses with benefits such as a reduced total cost of ownership and a faster time to market.

Figure 1-1 illustrates the example steps that a traditional IT customer might follow when developing a platform to host its software solutions.[1]

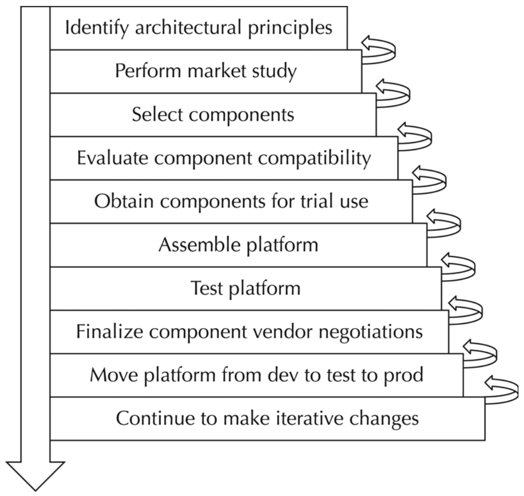
Identify architectural principles
Perform market study
Select components
Evaluate component compatibility
Obtain components for trial use
Assemble platform
Test platform
Finalize component vendor negotiations
Move platform from dev to test to prod
Continue to make iterative changes

FIGURE 1-1. *Iterative platform development process from a Fortune 500 IT organization (simplified to fit on one page)*

An *iterative platform development process* looks like this:

1. Identify architectural principles and/or platform requirements. In some cases, only architectural principles are known at the start of this process. In other situations, actual requirements have already been determined. Example architectural principles include support for open standards, a preference for integrated product suites, a preference for single-vendor–integrated stack solutions, service orientation, and so on. Platform requirements, on the other hand, will be application focused in terms of functional and nonfunctional requirements. Example requirements include specific capabilities, response times, quality of service, service-level agreements, and so on.

2. Perform market study and rate different technologies and products. During this step, different products are evaluated by the architecture team. Evaluation procedures vary but may include recommendations from analysts such as Forester and Gartner, previous experience with the products by team members, and testing the products in a lab environment.

3. Select software and hardware components. Note that this may be an iterative process, as selection of some components may compel selection of supporting components. Some selections may already have been made prior to this step, as key project stakeholders may have a strong preference for one or more components.

4. Evaluate component compatibility. After evaluating compatibility, you may need to go back to step 3 and iteratively make additional selections to increase compatibility. The first set of selections (and possibly subsequent selections) may not be sufficiently compatible, requiring further rounds of selections. For example, version 1 of product A may be compatible with version 2 of product B, but version 2 of product C requires version 2 of product A. This will force the platform selection team to determine which version of product A to standardize on, requiring the selection of a substitute component for product B or product C. It can even become possible to find a catch-22 situation where there is no certified line through a matrix of products and versions.

5. Obtain components from vendor(s) for trial use. Some vendors make software components available for free download for evaluation. (Oracle makes virtually all of its software components freely downloadable for trial use.) However, other vendors may require a nondisclosure agreement (NDA) or provide only a time limited or trial license. Furthermore, hardware components are often further restricted.
6. Assemble hardware and software components. In the case of limited access to hardware components, it may be acceptable to utilize a virtual machine environment such as Oracle VM to emulate the hardware environment for platform testing purposes.
7. Test the platform. If defects are identified, determine the root cause if possible. If the defect is caused by a product or (more likely) a combination of products, work with the vendor(s) to obtain fixes for the defects. If a single vendor is being utilized for the combination of products, it can make it easier to obtain a fix. (When multiple vendors are involved, one may blame another for any problems.) Once fixes are obtained, continue to test until platform is stable.
8. Finalize vendor price negotiations. Note that component selection may change at this point if price considerations are sufficient to cause a change of component.
9. Move the platform from development through test to production. Additional changes to the platform may result in response to new requirements and/or new incompatibilities discovered during this process.
10. Continue to make iterative changes when requirements change or when product changes, including the challenge of patching or updating the system of components over time (security patches, bugs, and so on).

The overall complexity of the list of steps is apparent after reviewing the list. Of course, different companies will have different versions of this list of steps. Many companies might not perform all of the steps, particularly in repetitive scenarios; however, the general pattern will be similar.

After reviewing the example list of steps, it becomes apparent that the original comparison of a box of parts of a model car to a preassembled model car does not truly capture the differences between traditional IT platform development and optimized engineered solutions. Instead, a more apt comparison may be the car that you buy from a car dealer versus the car that the protagonist assembled in the Johnny Cash song "One Piece at a Time."[2]

In the song "One Piece at a Time," the protagonist builds a car from parts selected from 1949 to 1973. The transmission is from 1953, the motor is from 1973, and there are many other mismatched parts. The protagonist ends up with two headlights on the left side of the car and one headlight on the right side. In some places, holes are missing bolts; in other places, there are no holes for the bolts. We may laugh along with the protagonist in the song at the resulting automobile, but we may shed tears when considering the results of traditional IT platform development. Some IT platforms are terminated after spending billions of dollars on development without ever providing value to the organization that paid for the system. That said, many traditional IT projects involve substantial local development and project management, and root cause determination for the cause of the failure is difficult. Purchasing a prebuilt platform such as Exalogic avoids the risk of a do-it-yourself platform.

One of the key advantages of optimized engineered solutions is that such systems enable customers to avoid processes such as those mentioned. Some organizations have hundreds of people involved in processes to develop standard platforms and test all of the components together. The staff resources spent on engineering the platform is far from free and doesn't provide competitive advantage. Engineered systems enable the enterprise of the future to skip the platform development process and instead focus on the applications that matter for the enterprise's core business. For these reasons and more, engineered and highly performance optimized systems are the future of the IT industry.[3]

Engineered systems yield many benefits, including the following:

- Removes the costs and risks of integration engineering
- Includes pretuned stacks for optimized performance
- Allows for single management and administration view
- Provides support from a single vendor

- Offers a consistent application to a disk patching mechanism provided by the vendor
- Reduces time to deployment.

TIP
If requirements do not mandate that you design your own application platform, you should consider an engineered system such as Oracle Exalogic Elastic Cloud. A prebuilt engineered system such as Exalogic will perform faster and will cost less to purchase, run, and maintain than a custom-designed application platform.

Exabytes of Information

Exadata and Exalogic received their names because of the exabytes of information that the data centers of the future will be required to handle. Our society is becoming increasingly inundated with digital information. Today, information is broadcast from satellites and transmitted over the airwaves, cables, fiber networks, and through other means. In 2004, monthly Internet traffic exceeded 1 exabyte, the equivalent of 1000 petabytes. In 2010, monthly Internet traffic exceeded 20 exabytes.[4]

An *exabyte* is a unit of information or computer storage equal to 1 quintillion bytes. One kilobyte equals 1000 bytes. One megabyte equals 1000 kilobytes. One gigabyte equals 1000 megabytes. One terabyte equals 1000 gigabytes. One petabyte equals 1000 terabytes. One exabyte equals 1000 petabytes. See Table 1-1 for a visual representation of the size of information from byte to exabyte.

Integrated Machines vs. Appliances

Some of the characteristics of engineered solutions, such as the preconfigured assembly of components, are similar to appliances. Appliances do not have configuration options and are designed to perform a simple task, such as a toaster that heats bread to a certain temperature. Exadata and Exalogic are not true "appliances," however. Instead, they are engineered platforms that have

Name (SI symbol)	Power of 10	Number of Bytes
Byte (B)	0	1
Kilobyte (KB)	3	1000
Megabyte (MB)	6	1,000,000
Gigabyte (GB)	9	1,000,000,000
Terabyte (TB)	12	1,000,000,000,000
Petabyte (PB)	15	1,000,000,000,000,000
Exabyte (EB)	18	1,000,000,000,000,000,000

TABLE 1-1. *Sizes from Byte to Exabyte*

been optimized to run application servers and databases. Hence, many of the traditional rules for managing application servers and databases continue to apply.

In the "One Day Installation Challenge" video on YouTube (www.youtube.com/watch?v=aWHPC188tus), Oracle demonstrates how to install an Exalogic machine in less than ten hours. In contrast, a company following the do-it-yourself platform development approach described earlier might spend more than six months to integrate and deploy a comparable platform.

With traditional IT systems, when a customer calls the support desk, before a problem can be diagnosed, a support team asks a customer numerous questions. These questions ask about the specific component vendors and versions used in the host environment, including operating system, hardware, network, storage, firmware, and patch levels. All must be determined before the actual process of diagnosing the problem can begin. Furthermore, the customer support organization for the specific component will not typically have access to an identically configured environment, especially if some of the elements are supplied by a different vendor. There are limitless combinations of how such components can be configured together. Indeed, during the issue resolution process, it may transpire that the root cause is actually related to a different element of the system belonging to a different vendor. When this occurs, the customer will need to begin this whole support issue process again, with the other product vendor, thus prolonging the issue resolution time.

In contrast, with integrated machines, you simply tell the customer support person the machine model you are using. Furthermore, the customer support desk has access to an identically configured machine, making it easy to replicate the problem that the customer has encountered and dramatically simplifying the troubleshooting process.

Oracle Exadata Database Machine

Although the focus of this book is the Exalogic machine, you'll also find it useful to have a basic understanding of its close cousin, Exadata. There is value in comparing and contrasting their architectures, and you need to keep in mind that both will often be used together.

The Oracle Exadata Database Machine is an integrated software and hardware platform that has been designed for data warehouses, Online Transaction Processing (OLTP), and database consolidation. The Exadata machine's InfiniBand fabric enables extremely fast input/output (I/O) communication between the storage server and the database server. (InfiniBand is also used within Exalogic for high performance I/O and is described in detail in Chapter 2.) Oracle Exadata includes intelligent storage server software that considerably increases the performance of data warehouses by reducing the amount of information that needs to be communicated between the data warehouse server and the storage server.

Exadata uses dedicated servers, or nodes, for storage and processing. The ratio of storage nodes to compute nodes in the Exadata machine is based on an understanding of the business problems that Oracle's customers are attempting to solve. (Compute nodes for Exalogic will be described in detail in Chapter 2.) The Exadata machine that is intended for data warehouses has a ratio that differs from the Exadata machine that is intended for OLTP applications. Both types of Exadata machines rely on the InfiniBand fabric to enhance performance between the database server and the storage server.

The database servers can utilize Oracle Linux or Oracle Solaris as the operating system. (Both operating systems are available for Exalogic, and are described in Chapter 2.) The storage servers use only Linux, and the ZFS Storage Appliance runs Solaris. Oracle 11*g* Release 2 is the current version of the Oracle database that runs on Exadata.

Appliances and similar solutions have been involved with data warehousing for a long time. Exadata, however, has optimizations that existing data warehouse appliances do not have. Furthermore, Exadata also provides the first optimized solutions for OLTP.

Exadata performance enhancements have focused on eliminating unnecessary communication between the database server and the storage server and speeding up communication between the database server and the storage server.

Oracle Exalogic Elastic Cloud

Oracle Exalogic Elastic Cloud is an engineered hardware and software machine designed to provide a platform for a customer's entire application portfolio. Exalogic includes hardware connected by an internal InfiniBand network fabric. An Exalogic machine can be connected with additional Exadata and Exalogic machines. Oracle's market-leading Java Enterprise Edition (Java EE) application server, WebLogic Server, has been reengineered for deployment to Exalogic and uses specific performance enhancements when running on Exalogic. The InfiniBand network fabric offers extremely high bandwidth and low latency, which provides major performance gains with respect to communication between the application server and the database server, and with respect to communication between different application server instances running with in the Exalogic system. Physically, Oracle Exalogic Elastic Cloud can be viewed as a rack of physical server machines plus centralized storage, which all have been designed together to cater to typical high-performance Java application use cases. Figure 1-2 shows the front view of an Exalogic machine.

In summary, Exadata is the database machine for the data tier and Exalogic is the middleware machine for the application tier.

Why Would an Organization Adopt Exalogic?

Why would an organization adopt the Oracle Exalogic Elastic Cloud machine? Just because Exalogic is an extremely high-performing Java solution, it does not mean that it is the perfect solution for every organization. Some organizations have architectural considerations that counsel adoption of alternative solutions. Further chapters will dive into these architectural considerations in greater detail, but the next few paragraphs describe some of the areas for which customers may find great value in adopting Oracle Exalogic Elastic Cloud. The next step is to understand Exalogic's hardware architecture in greater detail.

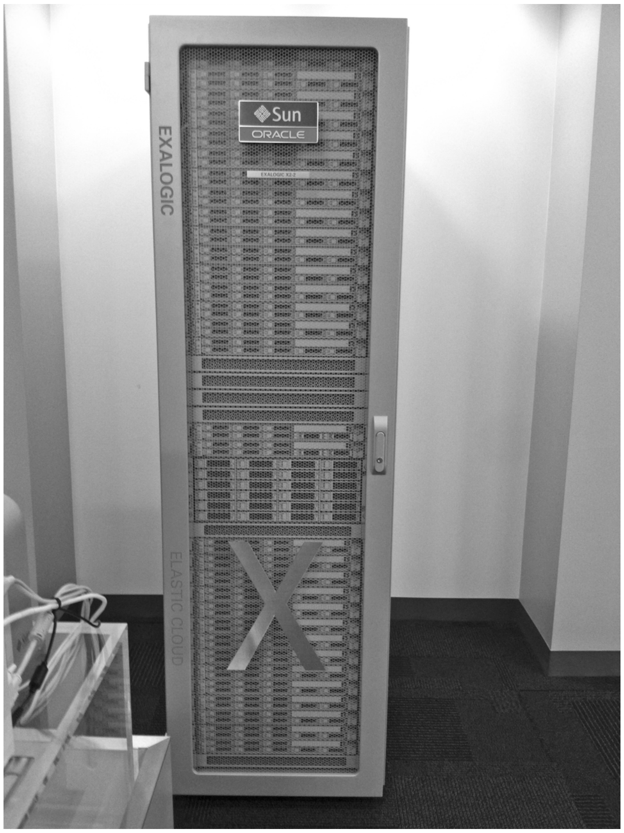

FIGURE 1-2. *Exalogic front view*

New Platform

If your organization is designing a platform for new applications based on Java technology, then Oracle Exalogic Elastic Cloud may provide great performance for a low cost. Cost savings with Exalogic come from a shorter time to production (the ability to install the system in a day), and the ease of managing the preconfigured appliance and maintaining and patching the system. Cost savings also come from the enhanced performance characteristics, which enable customers to reduce the amount of hardware and software that is necessary for a particular application. Exalogic is particularly relevant if the organization is interested in providing a private cloud from its data center (as discussed later in the section "Architecting Private Cloud Data Centers").

Performance Problem with Existing Platform

If the organization's current IT architecture is suffering from performance problems, migrating the existing applications to Exalogic may alleviate the performance issues without having to redesign and rewrite the applications. To a certain extent, this may be tempered by the specific versions of the applications that are suffering from the performance problems, as not all applications will benefit equally from deployment on to an Exalogic platform. Performance can be tremendously accelerated if the organization adopts both Exalogic and Exadata.

Hardware Refresh for Existing Platform

Most organizations refresh their existing hardware platform every three to six years with the next generation of hardware technology. If the organization is planning on replacing the existing hardware platform with next-generation hardware technology, this is an ideal time to consider Oracle Exalogic Elastic Cloud, and the dense high-performance computing hardware it employs, as the next-generation platform for the organization.

Oracle Workload

If the organization has already deployed Oracle technology in the data center, there will likely be significant performance benefits to moving the existing Oracle investment onto Oracle Exalogic Elastic Cloud, where Exalogic is explicitly tested, validated, and optimized for running such host applications.

TIP
If you currently use Oracle technology in your data center, you should consider adopting Oracle Exalogic Elastic Cloud the next time you do a hardware refresh. If you are currently suffering from performance issues, consider adopting Exalogic at your earliest opportunity. Your application performance should improve considerably while yielding a significant reduction in running costs.

Hardware Architecture

Oracle's Exalogic engineering staff conducted extensive testing on a wide range of hardware configurations to arrive at the optimal configuration for middleware type deployments. Design considerations included high availability, compute density, state-of-the-art components, balanced system design, field serviceability, centralized storage, and high-performance networking.

Exalogic includes Sun hardware connected by an internal InfiniBand network fabric. An Exalogic machine comes in several supported rack configurations. Up to eight of these racks can be connected together with the internal switches that are provided; more than eight can be connected together with additional external InfiniBand switches. Processing is performed by 30 Sun Fire X4170 M2 compute nodes; each compute node has two 64-bit Intel processors with six cores each (12 cores in total per compute node). A full rack has 30 compute nodes, a half-rack has 16 compute nodes, a quarter-rack has 8 compute nodes, and a one-eighth rack has 4 compute nodes. Shared storage is provided by a built-in Sun ZFS Storage 7320 appliance, which is accessible by all the compute nodes. InfiniBand and Ethernet switches enable network communication. Figure 1-3 shows the rear view of an Exalogic machine, including the InfiniBand cabling.

Figure 1-4 shows the Exalogic Hardware architecture. Chapter 2 explores Exalogic's hardware architecture in more detail.

Exalogic is accompanied by a set of software tools, including the Exalogic Configuration Utility (previously known as OneCommand). This utility configures the networking for the system to be accessible via the customer's wider data center network. (For the YouTube video, Oracle product development engineer Ram Sivaram was videotaped unwrapping and installing an Exalogic machine in ten hours.[5])

InfiniBand is a networking technology similar to Ethernet and is used as an alternative for high-performance computing within a data center. InfiniBand Technology is a critical component within both Exalogic and Exadata. The InfiniBand Trade Association (IBTA) was founded in 1999 and is led today by Oracle, Intel, IBM, Mellanox, QLogic, and Voltaire. The goal of the IBTA is to standardize a communication link for high-performance computing systems such as Exadata and Exalogic. Applications can take advantage of the low network latency and high performance provided by

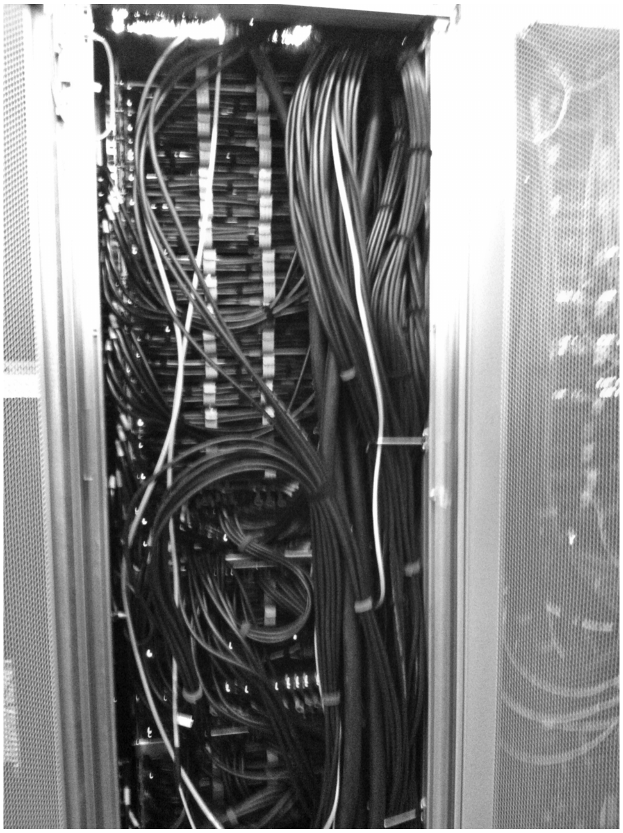

FIGURE 1-3. *Exalogic rear view including InfiniBand cabling*

InfiniBand (IB) through protocols such as Sockets Direct Protocol (SDP) or Internet Protocol over InfiniBand (IPoIB). "Exalogic-ignorant" applications work by virtue of implicitly using IPoIB, and IB-aware applications can explicitly use IB's native SDP to reduce latency further—an example of software engineered for specific hardware. Exalogic utilizes 40-gigabit InfiniBand technology to accelerate performance, in comparison to 1-gigabit

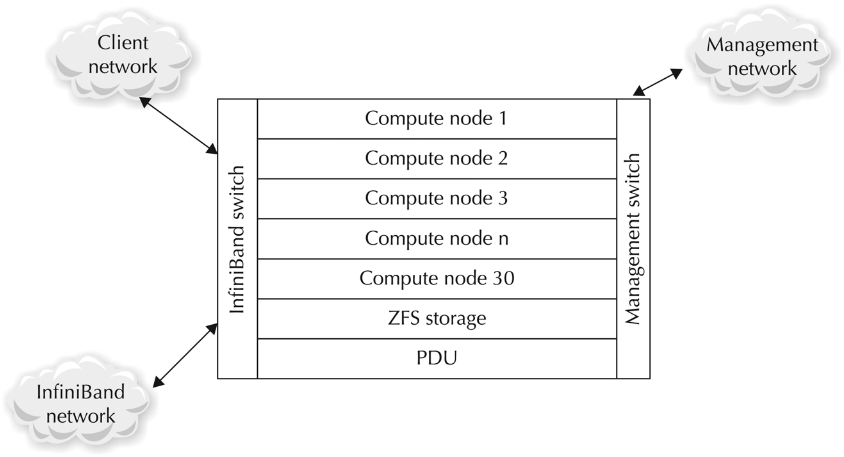

FIGURE 1-4. *Exalogic hardware architecture*

or 10-gigabit Ethernet networks used in typical systems. InfiniBand technology is continuing to evolve and future generations will be even faster.

Software Architecture

Several operating systems can be run on each compute node in an Exalogic system. Either Oracle Linux 5.5 or Solaris 11 Express, for x86-64 based processors, can be run on each compute node, with the restriction that all compute nodes must run the same type of operating system. Any vendor's software that is supported by the vendor for the above mentioned operating system is also supported to be deployed on Exalogic.

If you're running Solaris 11 on Exalogic, you can use operating system–level virtualization to help run applications on multiple instances of Solaris on a single compute node (including versions of Solaris such as Solaris 10). This virtualization technology in Solaris is called Solaris Zones. Refer to Chapter 4 for more details on virtualization.

Of course, just providing a enhanced operating system to run optimally on Exalogic is not enough. The real power of the engineered hardware and software combination comes from running Oracle's Exalogic-aware and

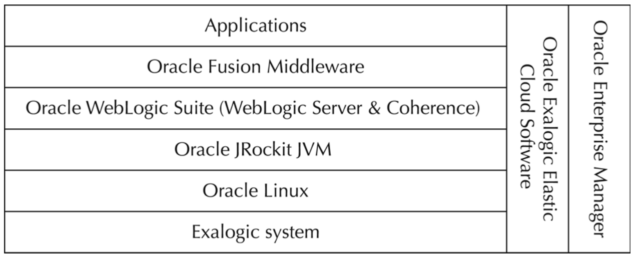
Applications
Oracle Fusion Middleware
Oracle WebLogic Suite (WebLogic Server & Coherence)
Oracle JRockit JVM
Oracle Linux
Exalogic system
Oracle Exalogic Elastic Cloud Software
Oracle Enterprise Manager

FIGURE 1-5. *Exalogic software architecture*

optimized middleware products on top of Exalogic's hardware and operating systems. The following sections introduce some of these key middleware components, including Oracle WebLogic Server, Oracle Coherence, plus the Oracle's management toolset for Exalogic, Oracle Enterprise Manager. See Figure 1-5 for a graphic representation of Oracle Exalogic Software architecture.

An alternative architecture based on Sun Solaris and the HotSpot Java Virtual Machine (JVM) is shown in Figure 1-6.

Version 2.1 and subsequent versions of Exalogic include Oracle Virtual Machine, as shown in Figure 1-7.

Chapter 3 will explore this software architecture in greater detail.

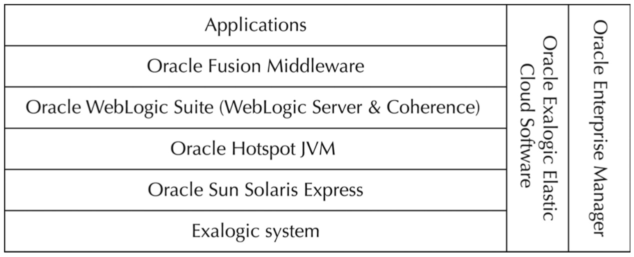
Applications
Oracle Fusion Middleware
Oracle WebLogic Suite (WebLogic Server & Coherence)
Oracle Hotspot JVM
Oracle Sun Solaris Express
Exalogic system
Oracle Exalogic Elastic Cloud Software
Oracle Enterprise Manager

FIGURE 1-6. *Exalogic software architecture*

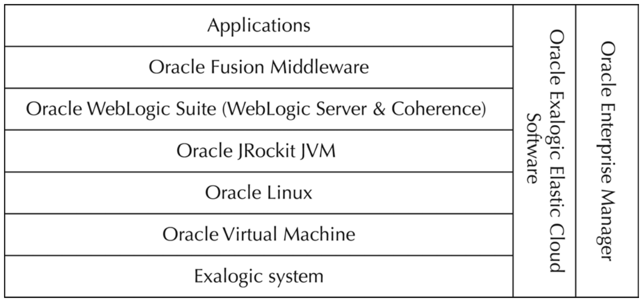
Applications
Oracle Fusion Middleware
Oracle WebLogic Suite (WebLogic Server & Coherence)
Oracle JRockit JVM
Oracle Linux
Oracle Virtual Machine
Exalogic system
Oracle Exalogic Elastic Cloud Software
Oracle Enterprise Manager

FIGURE 1-7. *Exalogic software architecture*

Oracle WebLogic Server

Oracle WebLogic Server (WLS) is the market leading Java EE Server. (Java EE is discussed in greater detail in subsequent sections and in Chapter 3.) WLS can be installed and run on many operating systems and hardware platforms. WLS is used to run mission-critical systems and applications, especially where reliability, availability, scalability, and performance are requirements.

WebLogic, Inc. (the company) was founded in 1995 with the goal of producing the world's first web application server (called Tengah). In 1998, BEA acquired the San Francisco startup WebLogic, Inc., and the application server was renamed WebLogic Server and became one of the main inspirations for Sun Microsystems' subsequent Java EE specification and industry standard. In 2008, Oracle acquired BEA, including WebLogic Server. Oracle has positioned WebLogic Server at the core runtime for its Middleware and Application products, Fusion Middleware.

Oracle is the market leader in the enterprise application servers market with more than 43 percent of the market. Gartner has published its application server market share numbers for 2010 based on total software revenues. According to Gartner, Oracle holds more market share than its four closest competitors and grew at a rate of 17.8 percent, faster than the industry average of 12.1 percent.[6]

Oracle WebLogic Server has set numerous world records for performance, including achieving the highest EjOPS/core of any SPECjEnterprise2010 result.

WLS also holds the world record for the jAppServer2004 results. More details on these specifications are available by consulting the references in the appendix.[7]

Oracle WLS 10.3.4 (and subsequent versions) contains the engineered optimizations necessary to leverage all the capabilities of the Exalogic hardware components, especially InfiniBand. If WLS 10.3.4 is running on a non-Exalogic system, it will be unable to take full advantage of these optimizations. Previous versions of WLS will run on Exalogic (if they are certified to run on Oracle Linux 5.5 or Solaris 11, generally) but will not be able to take full advantage of the Exalogic-specific software performance enhancements that appear in later versions of WebLogic.

Java Enterprise Edition

Java EE (formerly J2EE) is the industry standard for enterprise Java computing. Java EE is actually a collection of related specifications that have evolved over time. The Java EE standard adhered to by middleware vendors enables developers to create Java applications that leverage common capabilities while avoiding being locked into a specific vendor's Java EE product. Applications that rely upon the Java EE standard can be easily migrated to another Java EE platform.

Java is a programming language that was developed by Sun and released in 1995. It is an object-oriented language similar to C and C++, but with a simpler object model and automated garbage collection (memory deallocation). As a result, using Java typically results in less error-prone and more productive application development, compared with its programming language predecessors. Java employs the concept of a virtual machine that C++ does not, enabling developers to write Java code that is compiled into a file that could be deployed onto any platform that includes a JVM, regardless of what type of operating system and processor the platform runs on.

Sun's slogan of "write once, run anywhere" was based on this virtual machine concept. Soon after the initial release of Java, major web browsers added a JVM plug-in, enabling Java programs contained in HTML web pages (called *applets*) to run in the browser. Java became very popular because of the Java Applets technology; in the late 1990s, this was the first browser-based technology to support dynamic rich content. Figure 1-8 shows a graphic representation of Java client-side technology.

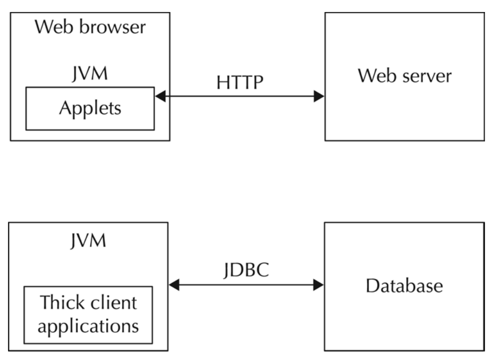

FIGURE 1-8. *Java client-side technology*

In the first example shown in Figure 1-6, a Java applet is displayed running in a web browser. As a security precaution, applets can communicate only to the web server they were downloaded from and cannot, for example, read or change the contents of files on the browser's local file system. In the second example, a Java thick client application running in a JVM on a client-side machine can communicate to a database server using Java's database connection API (JDBC).

However, the Java Standard Edition did not have all of the capabilities that enterprises sought for building long-running server-side applications. This led to the development of several technologies that were bundled together in the Java Platform for the Enterprise.

The original goal for the Java Platform for the Enterprise was to provide support for web applications where dynamic content is created on the server side (using Java Servlets and JavaServer Pages) and to provide support for running business logic components on the server (Enterprise Java Beans).

The initial Java Platform for the Enterprise in 1998 included specifications for Enterprise Java Beans (EJB 1.0), Java Servlets (2.1), JavaServer Pages (JSP 1.0), Java's database connectivity API (JDBC 1.1), and Java Naming and Directory Interface (JNDI 1.0) API.

Java Servlets

Java Servlets is a technology designed to generate dynamic web page content on a server that is streamed back to the web browser as HTML pages, ready to be rendered by the browser in a human-readable form.

Servlets are a multithreaded technology intended to improve upon many of the drawbacks that other server-side HTML-generating technologies of the late 1990s suffered from, such as the Common Gateway Interface (CGI), commonly used in web servers of the time. Servlets are deployed in an application server container that provides services that the servlet can rely upon, reducing the amount of software coding that is required to create a web application. Figure 1-9 shows Java servlets.

An example servlet might be an application created for online banking. The servlet in Figure 1-9, for example, receives a request for a customer's account balance. The servlet looks up the customer's current balance information using JDBC and then dynamically generates a HTML web page to send back to the customer, showing the account balance and associated account information.

JavaServer Pages

The JavaServer Pages technology was created to simplify the process of creating a web application using Java servlets, to reduce the amount of coding required. A JSP page appears similar to an HTML page, except that it includes special JSP tags that are used to generate bits of content dynamically on the server mixed in between the fragments of static HTML. This processing might include retrieving information from a database to populate a generated web page, for example. The generated HTML page is then sent from the server to the web browser. The JSP page itself is never seen by the user of the web browser, only the resulting generated HTML is seen.

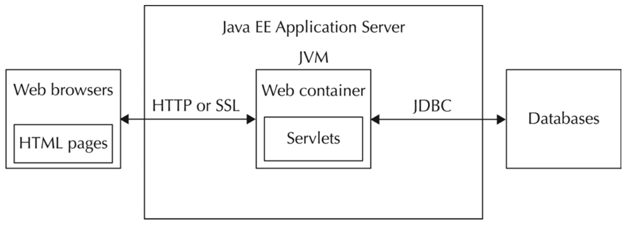

FIGURE 1-9. *Java servlets*

The advantage of JSP is that the technology separates the presentation logic from the business logic in a server-side application, thereby enabling different developer roles to handle the creation of different types of software logic. So presentation developers who understand HTML but do not understand Java can easily see how these pages will be displayed and can concentrate on getting the look, feel, and flow of a web page just right (unlike with traditional servlets, which are composed of pure Java code). Java developers can be responsible for the business logic, which is generated by special tags or in JavaBeans that are referenced from the JSP. Figure 1-10 shows how a JSP page returns a response page after receiving a request.

In Figure 1-10, a web browser sends a request for a JSP page. The web container (shown as the servlet/JSP container in the figure) recognizes that the request is for a JSP page and compiles the JSP page, generating a servlet. The servlet is then executed to generate the HTML response page. The HTML response page is then sent back to the web browser that sent the original request. Future requests to the same JSP will not incur the startup time of compiling the JSP and generating the servlet, as the JSP has to be compiled only once, regardless of how many different clients request it. Using the banking application example, the JSP version will have the same functionality as the servlet version, except that the bank can employ a user interface specialist who does not understand Java, but is more skilled in creating visually appealing and usable web pages, while freeing the Java developer to write the code for retrieving the customer account information from the database.

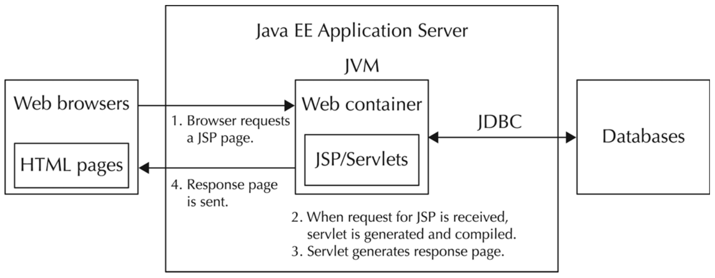
Java EE Application Server
JVM
Web browsers
HTML pages
1. Browser requests a JSP page.
Web container
JSP/Servlets
JDBC
Databases
4. Response page is sent.
2. When request for JSP is received, servlet is generated and compiled.
3. Servlet generates response page.

FIGURE 1-10. *Java Server Pages returning a response page after receiving a request*

TIP
We recommend separating the business logic from the presentation layer. JavaServer Pages are ideal technologies for separating the presentation layer from the business logic layer.

Enterprise JavaBeans

Enterprise JavaBeans (EJB) is a server-side component architecture for modular construction of the business logic for enterprise applications. The standard provides a pure-Java component architecture that is similar in concept to the Common Object Request Broker Architecture (CORBA) industry standard that emerged in the early-1990s and is in some ways an unofficial successor to CORBA, albeit for Java-based applications only. The EJB components are deployed into a container, in a similar manner to how Java Servlets and JSP pages are deployed into a container. However, EJB technology is far more complex than Servlet or JSP technology, and, with the benefit of hindsight, was deemed too complex. The EJB standard has been simplified in subsequent versions. Figure 1-11 shows an example of the EJB container along with the web container.

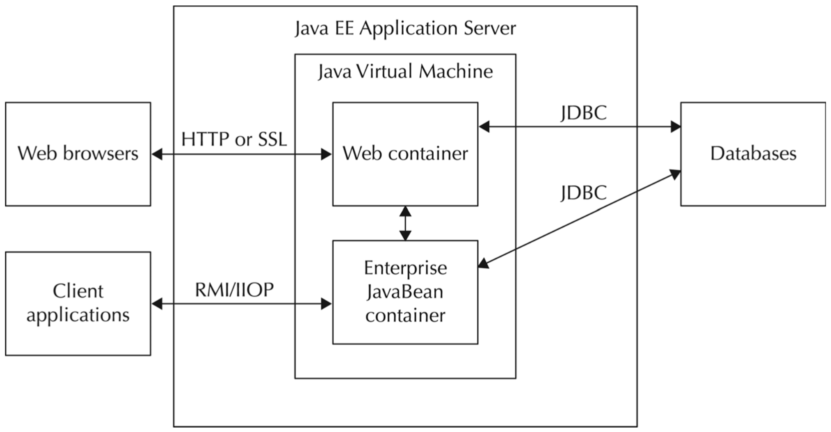

FIGURE 1-11. *EJB and web containers*

In a banking example, EJBs are used for transactions management and enabling easier persistence and retrieval of data from relational databases. Transactions management is important if the application is doing anything more than just displaying current data. For example, if the banking application allows the customer to change the account balance via a transfer, EJBs and transaction management should be used to ensure that a withdrawal from one account and the deposit to the other account is executed atomically as a single operation. Otherwise, during error conditions, money may go missing or duplicate deposits may occur.

TIP
If there is a need for transactions management, consider using Enterprise JavaBeans.

A Little Java History

In December 1999, version 1.2 of the enterprise Java standard was released. Java had been renamed Java 2 in December 1998. The Java Platform for the Enterprise was subsequently renamed Java 2 Enterprise Edition (J2EE). J2EE version 1.2 added incremental changes to the previous components of J2EE (servlets, JSPs, EJBs, and so on) and also added support for transactions and messaging. Java Message Service (JMS) provided capabilities for asynchronous and synchronous messaging and support for publish/subscribe models. (JMS was a standard for messaging; the predominant proprietary technology that led to the development of JMS was Message Oriented Middleware and IBM's MQ Series product.) Another type of EJB was introduced, called Message-Driven Beans. Transaction support was provided through a container-based Java Transaction Service (JTS, based on the Object Transaction Service from CORBA, which itself was based on the X/OPEN standard) and a Java Transaction API (JTA) for writing code to interact with the JTS.

In September 2001, J2EE version 1.3 was released. In addition to including incremental changes to previous components of J2EE, this version added support for the Java Connector Architecture (JCA) and XML (Java API for XML Processing, or JAXP). JCA is a Java-based technology solution for connecting application servers and enterprise information systems. JDBC is used to connect JEE applications to databases, and JCA is a more generic architecture for connection to legacy systems. XML is becoming of increasing importance to Java development. Almost all enterprise Java programs involve XML at this point.

In November 2003, J2EE version 1.4 was released. This version added support for web services. J2EE 1.4 was delayed while waiting for the Basic Profile 1.0 web services to be completed. The Basic Profile was developed by a consortium that focused on interoperability conformance for the different web services standards.

In May 2006, Java EE version 1.5 was released. (Java 2 had been renamed Java, and hence J2EE had been renamed JEE.) JEE version 1.5 simplified the EJB persistence model.

In 2009, Oracle acquired Sun and committed to continue to support the Java EE standards. Subsequently in December 2009, Java EE 6 was adopted. This version is supported in the latest version of WLS. Java EE 7 and Java EE 8, which have not yet been released, are focused on adding additional capabilities for cloud computing into the JEE standard.

Java EE applications deployed on Exalogic may use all of the preceding technologies, as shown in Figure 1-12.

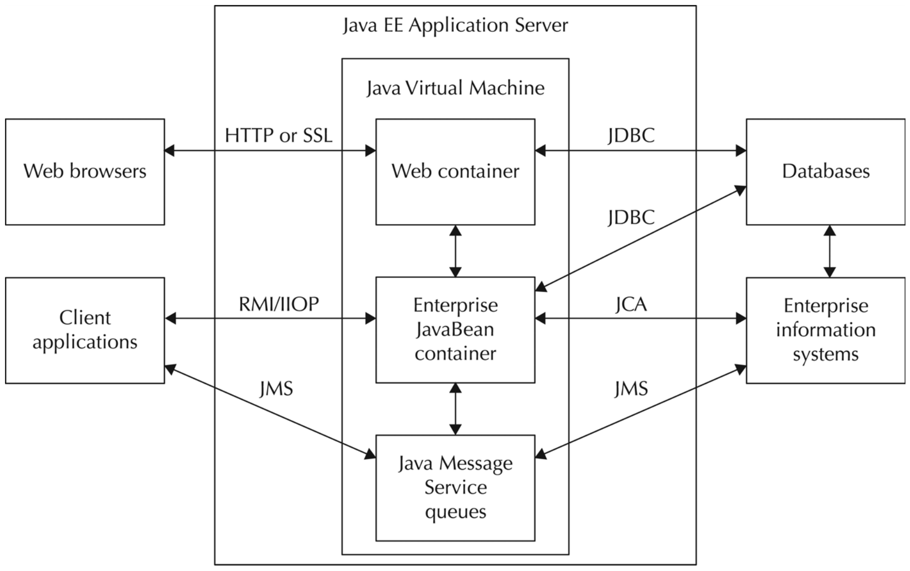
Java EE Application Server
Java Virtual Machine
Web browsers
HTTP or SSL
Web container
JDBC
Databases
JDBC
Client applications
RMI/IIOP
Enterprise JavaBean container
JCA
Enterprise information systems
JMS
JMS
Java Message Service queues

FIGURE 1-12. *Example Java EE application*

Oracle WebLogic Suite

Oracle WebLogic Suite is the name for a collection of related Oracle products. In addition to Oracle WLS Enterprise Edition, it also includes Oracle Coherence Enterprise Edition, Oracle WebLogic Real Time (JRockit), Oracle Virtual Assembly Builder, Oracle Web Tier, and two Oracle Enterprise Manager add-ons: Diagnostics Pack for Java and Management Pack for Coherence.

NOTE
Oracle Enterprise Manager: Diagnostics Pack for Java, which includes Advanced Diagnostics for Java (AD4J) and JRockit Mission Control, is discussed in the separate section that follows. Oracle Coherence and the management pack for Coherence are also discussed in a separate section.

Oracle WebLogic Real Time is a real-time solution for Java based on JRockit's real-time features. This solution is typically used by applications that have a requirement for deterministic scheduling with specified timing bounds. Java implements real-time solutions by providing deterministic control of the JVM garbage collection thread. Customers interested in real-time solutions include Wall Street banks (high-frequency traders, and so on), military command and control systems, and other applications where microseconds are critical.

Oracle Web Tier includes components that interact with end users through HTTP requests and responses. Capabilities include security gateways, support for a DMZ in front of the application server tier, load dispatching, and so on. Products include the Oracle iPlanet Web Server, the Oracle iPlanet Web Proxy Server, Oracle Traffic Director, Oracle Web Cache, and Oracle HTTP Server (an Apache-based server).

Oracle Virtual Assembly Builder (OVAB) is designed to help organizations configure multi-tier applications and provision them onto virtual machines. For example, a web server, an application server, and a database can be packaged and deployed as a single unit onto Oracle VM. OVAB structures the process of combining the components, deploying the components, and configuring the components. Virtualization enables organizations to separate the application from the underlying hardware infrastructure and operating system. Many organizations are investing considerable resources into virtualization to achieve greater agility and cost reductions.

Oracle Enterprise Manager: Diagnostics Pack for Java, which includes Advanced Diagnostics for Java (AD4J) and JRockit Mission Control, provides capabilities to perform JVM diagnostics. This package is included with WebLogic Suite and is an Enterprise Manager component. JVM Diagnostics is important whenever an administrator is trying to discover the root cause of a production problem.

Oracle Enterprise Manager (OEM) is Oracle's administrative console for the enterprise. OEM enables organizations to monitor all of their applications across the Oracle stack, from the application tier through the middleware and the database down into the hardware with OEM Ops Center. The particular Enterprise Manager packs mentioned so far are commonly required for enterprise Java deployments using WebLogic Suite.

Oracle Coherence

Oracle Coherence is an in-memory distributed data grid designed for clustered applications and application servers. Coherence provides distributed data caching on top of a scalable peer-to-peer clustering protocol. The advantage of using Coherence is that the data is located close to the processing applications while the Coherence data grid handles the data persistence functionality. This offers the benefit of scalability, with additional servers being able to handle an increased capacity of data without impacting response time.

Oracle Coherence is useful when you need to reduce the amount of time it takes to retrieve persistent data from a data source. Instead of a client application needing to communicate across the network to the data source and wait for the data source to process the request and compute a result, the result information is cached in the Coherence data grid for quick retrieval. This performs especially well when the data cache is on the same server with the client application. Coherence maintains the relationship between the data in the distributed cache and the original data source.

Coherence can reduce latency for application servers by reducing the time it takes to generate responses. Coherence provides high availability by transparently failing over the clustered services and redistributing cached data when necessary. When a new server is added, it automatically joins the cluster and Coherence fails back services to it, transparently redistributing the cluster load. Figure 1-13 illustrates the Coherence data grid.

As shown in Figure 1-13, several application servers are utilizing data stored in a clustered Coherence grid. The application servers are located near the data cache that they are utilizing.

Some optimizations for Exalogic have already been added to Coherence. For example, Coherence includes an Elastic Data feature that increases performance when used in conjunction with the local 100GB solid state drives (SSDs) in the Exalogic compute nodes. The Elastic Data feature has been optimized to prioritize RAM availability to store the primary copies of data, by moving backup copies to or from RAM to flash. The feature also triggers its internal garbage collection as its remaining RAM capacity approaches zero; alternatively, when RAM is plentiful, the internal garbage collection feature will be less likely to trigger. Other optimizations include increased concurrency when overflowing to flash devices, and more are planned for future releases.

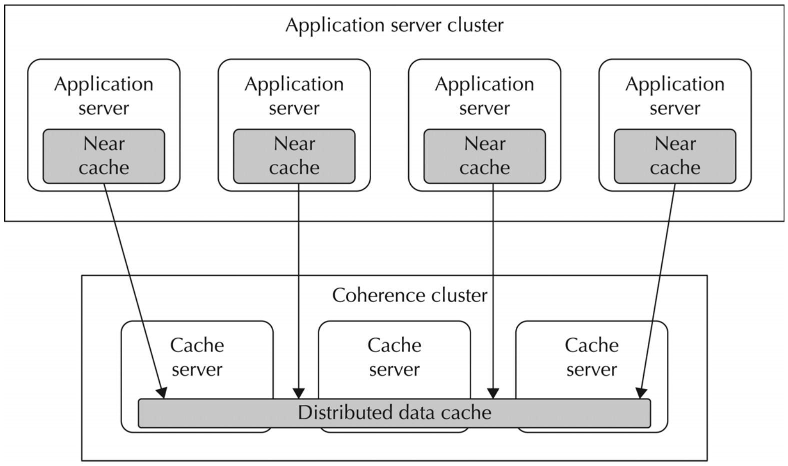

FIGURE 1-13. *Coherence in-memory data grid*

Oracle Fusion Middleware and Fusion Applications

Many of the Oracle Fusion Middleware 11*g* components run on WLS and can therefore take advantage of the InfiniBand-based and other performance enhancements that are present in WebLogic when running on Exalogic.

Oracle Fusion Middleware includes best of breed products such as Oracle WebLogic Suite, Oracle WebCenter Suite, Oracle SOA Suite, Oracle Identity Management, Oracle Service Bus, Oracle JDeveloper, and so on. These middleware products provide capabilities for business process management, enterprise and legacy systems integration, content management, data integration, business intelligence, event driven architecture, SOA governance, business transaction management, and transaction processing.

Oracle Applications includes applications such as PeopleSoft, JD Edwards, Siebel, ATG, Flex, E-Business Suite, and a new suite of applications known as Oracle Fusion Applications. Oracle Fusion Applications are architected to run on Oracle Fusion Middleware and will be able to take advantage of the capabilities of Exalogic. Oracle Fusion Application areas include customer relationship management; financials; governance, risk, and compliance; human capital management; procurement; project portfolio management; supply chain management; and other areas. Advantages of running Oracle Fusion Applications on Exalogic are discussed in greater detail in Chapter 8.

Oracle Enterprise Manager

OEM is designed to be a single management console that can manage all the levels of today's complex IT infrastructure, from the application layer to the hardware layer. As organizations move into cloud computing, they'll find it no longer sufficient to use management tools that are designed for a single layer or a single vertical application. Instead, they will need access to a tool set that can be used to manage the enterprise. Oracle Enterprise Manager is intended to offer that capability.

Oracle Enterprise Manager is composed of two core management tools: Grid Control and Ops Center. Grid Control is used to manage and provision software. Ops Center is used to manage and monitor the hardware and the network. Other optional management packs add capabilities to Oracle Enterprise Manager.

Exalogic Performance Enhancements

Exalogic performance enhancements enable applications to take full advantage of InfiniBand's fast networking infrastructure. In addition to IB-based enhancements, there are also enhancements to the JDBC driver, JVMs, and other features. These performance enhancements fall into three general categories: enhancements to increase communication performance between the application server and the database server; enhancements to increase communication performance among application servers; and general enhancements to WLS. In addition to performance enhancements to WebLogic, performance enhancements are being made to many other products in the Oracle Fusion Middleware platform, including Oracle Coherence. Layered products running on SOA Suite, WebCenter, and so on, will then implicitly take advantage of these optimizations.

Communication between WLS instances on Exalogic and database server instances on Exadata can take full advantage of performance enhancements provided by InfiniBand. There are also improvements to integration between WebLogic and Oracle RAC.

Application Server cluster communication is improved by InfiniBand. For example, web applications can keep track of the current state of end user interactions using an HTTP session object managed by the application server. WLS can automatically replicate session data from the primary server to a secondary server in the cluster each time session data changes. However, session replication incurs a performance cost if a large amount of user data is being held. For Exalogic, WLS's replication mechanism is improved to utilize the 40 Gb/s I/O bandwidth provided by InfiniBand for interprocess communication between servers.

WLS (and the underlying JVM) includes additional general enhancements, such as network request handling, memory, and thread management. These enhancements yield better response times, better throughput, and other benefits that are described later in this book. These enhancements are specific to Exalogic because they alleviate bottlenecks that would otherwise materialize on dense computing resource environments.

Architecting Private Cloud Data Centers

The term “cloud computing” dates back to the days in which a cloud was drawn on a whiteboard to abstract away the network details. Today, cloud computing means much more than just an abstraction to simplify concepts on a whiteboard.

Sometimes referred to as a “Cloud in a Box,” Exadata and Exalogic provide the necessary application platform and information management capabilities that can be the foundation for a private cloud. While platform capabilities for information management and application platforms are a necessity, these capabilities alone are an insufficient condition for a private cloud. Also required for a private cloud is the ability to perform provisioning of resources, metering, and charge back for services that are used. Oracle Enterprise Manager can provide the necessary management capabilities to create a private cloud leveraging Exadata and Exalogic.

NIST Definition of Cloud Computing

Although there are several competing definitions of cloud computing in the industry, the National Institute for Standards and Technology (NIST) has provided the most accepted and comprehensive definition. NIST defines cloud computing as “a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction” (NIST Draft Definition of Cloud Computing v15).[8]

The NIST model includes five essential characteristics:

- **On-demand self-service** Consumers want to be able to access cloud resources without having to ask IT for access.
- **Resource pooling** To provide resources that can scale up to meet the consumer self-service demand, resources should be pooled.
- **Rapid elasticity** Consumers should be able to use the resources they need when they want them; resources should scale up and scale down as necessary to meet demand.

- **Measured service** Because resource usage will scale up and scale down as driven by demand, resource usage can be unpredictable. Therefore, resource usage should be measured to provide the ability to charge based on usage. Measured service proposes a metering system for IT (similar to how utilities bill for electricity or water usage) as opposed to the traditional approach where you pay for the system regardless of how much or how little you use it (the industry average for server utilization is under 30 percent).
- **Broad network access** To serve a wide range of consumers with a wide range of needs, it will be necessary for cloud providers to offer broad network access to cloud resources.

The NIST model describes three service models:

- **Infrastructure as a Service (IaaS)** IaaS providers typically offer virtual machine images that are either a base operating system or perhaps with some additional software deployed on the image. IaaS enables organizations to abstract out and enable allocation of compute resource, storage, network, and so on, on demand through the use of virtualization. Dynamic provisioning enables organizations to scale up and down as necessary. Typical IaaS providers include Amazon.com (public IaaS), Oracle VM (private IaaS), and other providers that offer virtualized machine images.
- **Platform as a Service (PaaS)** PaaS providers typically offer a platform upon which cloud consumers can build applications; this platform typically has an API and provides a variety of services to support the applications. Examples could include middleware as a service, database as a service, identity management as a service, and so on. A platform providing these services could then be dynamically provisioned on demand depending on the user's requirements. Typical PaaS providers include Oracle Exalogic Elastic Cloud (for private PaaS), Google App Engine (for public PaaS), Force.com (public PaaS), and other platforms on which cloud consumers can create applications.

- **Software as a Service (SaaS)** SaaS providers typically offer a multitenant application that can be configured to meet the application user's requirements. With multitenancy, multiple users (or organizations) share the application, but the application is configured so that user or organization has access only to their own data. Typical SaaS providers includes Oracle On Demand, SalesForce.com, Google Apps, and other applications that are provided in the cloud.

Finally, the NIST model includes four deployment models:

- **Public cloud** Public cloud providers offer their cloud resources to the general public over the Internet. Amazon, Google, and SalesForce.com are examples of public cloud providers.
- **Private cloud** Private clouds are built for a single organization and provide cloud resources to their internal departments and lines of business. Private clouds are built primarily when an organization wants greater controls than public clouds can offer (including performance guarantees, location of data, security, and so on). Numerous large organizations are building private clouds for their internal customers.
- **Community cloud** Community clouds are related to private clouds, with similar security and governance considerations that can leverage each other's resources to scale elastically. An example would be multiple private clouds developed for the U.S. Department of Defense that could leverage each other's resources to scale elastically while meeting Department of Defense security requirements.
- **Hybrid cloud** Hybrid clouds are a combination of private clouds and public clouds. In this case, an organization builds a private cloud for the control that the private cloud offers but makes use of public clouds when there is a need for greater elasticity than the private cloud can provide. An example would be an organization that needs to scale up resources on Super Bowl Sunday to handle a peak in consumer response after showing a Super Bowl ad.

Oracle Exalogic Elastic Cloud is an ideal building block to enable an organization to create a private cloud with a PaaS capability. Exalogic and Exadata can be combined in a PaaS offering, with Exadata providing Database as a Service capabilities and Exalogic providing Application Serving as a Service.

OEM can provide the capabilities for on-demand self-service and measured service, with Exalogic and Exadata providing the capabilities for resource pooling, rapid elasticity, and network access.

TIP
If your data center is planning to offer private cloud–based application resources and database resources, you should consider a private PaaS with Exalogic and Exadata as the foundation for your cloud data center. OEM can offer provisioning, metering, and charge-back services for your private cloud.

Public Clouds and Private Clouds

In August–September 2010, the Independent Oracle User Groups (IOUG) surveyed its members regarding enterprise cloud initiatives.[9] As of the fall of 2010, only 13.8 percent of respondents were using public clouds. However, the introduction of the Oracle Cloud and other public clouds designed for enterprise customers should increase this adoption rate.

Private clouds are of primary interest to many enterprise customers today. According to the 2010 IOUG survey referenced above, 28.6 percent of respondents have private clouds that are either in production or in a pilot. There is a two to one ratio when comparing the percentage of respondents who indicated that their organization is pursuing a private cloud to the percentage that is pursuing a public cloud (28.6 percent versus 13.8 percent).The disparity between private cloud adoption and public cloud adoption is driven by several factors, as indicated in the survey. Security concerns were the number one reason why respondents selected a private cloud effort over a public cloud. The number two issue was quality of service concerns: the ability to define an enterprise Service Level Agreement (SLA) for a private cloud instead of being forced to accept a public cloud's mandatory terms for an SLA. As the survey shows, private clouds in 2010 were still an immature technology and a number of issues remained to be resolved. That said, private clouds do avoid some of the key issues of public clouds—namely the security and SLA issues.

Why Use Exalogic in a Private Cloud Data Center?

Oracle Exalogic Elastic Cloud helps organizations achieve the opportunities and benefits that private clouds offer. It provides the high-performance computing power necessary for a private cloud. Exalogic, when combined with OEM's management capabilities, makes an excellent private cloud platform.

OEM helps provide the necessary capabilities that organizations require for provisioning server and storage capacity. OEM also provides support for metering and charge-back to enable organizations to create the business case and funding model for their private cloud. OEM also provides capabilities for managing service levels and providing visibility and control over applications and other resources.

Future versions of Exalogic will offer even greater support for private clouds.

Summary

If you are running Oracle middleware or applications, Oracle Exalogic Elastic Cloud is an extremely fast engineered system for running Oracle middleware and Oracle application workloads. Anyone who is considering improving performance for an existing system, refreshing a current hardware platform, or considering a new project should seriously consider choosing Exalogic as the platform of choice.

The value of engineered systems to cloud data centers was described in this chapter. It was also the first technical chapter in this book and served as a technical introduction to the rest of the book, as many of these topics are described in greater detail throughout.