Solaris™ Containers — How Advances in Server Virtualization Will Simplify Service Manageability

---

A Technical Brief for IT Executives and Managers
February, 2002

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303
1 (800) 786.7638
1 (512) 434.1511

CONTENTS

Executive Summary **1**

Manageability — The Top Layer in the IT Hierarchy of Needs **3**

Server Virtualization **5**

Solaris Containers: The Next Step in Server Virtualization 6

The Evolution of Solaris Containers **7**

Example: The Dynamic Aspect of Solaris Containers 8

Example: Using Solaris Containers in a Clustered Environment 9

Sun Product Directions for Solaris Containers **12**

A Future View of Solaris Containers and Service Level Management **14**

Manageability — The Value of Solaris Containers **16**

Conclusion **18**

CHAPTER 1

EXECUTIVE SUMMARY

After nearly a decade of experience with large-scale distributed computing, businesses understand how to build scalability and availability into their IT infrastructure. Now, with the escalating costs of managing vast networks of servers with software components installed on thousands of nodes, the focus has shifted from increasing performance and availability to reducing the cost of IT infrastructure and better managing end-user service levels. Sun's vision is that businesses can accomplish this through a new approach that relies on *server virtualization*.

End-user application services are frequently comprised of components that are distributed across multiple servers. To reduce costs, businesses are eager to consolidate these applications onto fewer servers, but must be careful about maintaining isolation between applications. Advances in hardware and software have ushered in the idea of server virtualization, a concept that allows servers on the network to be flexibly partitioned into independent execution environments that provide isolation within the same server. Server virtualization allows data centers to be visualized and managed as a fabric of interconnected computing resources rather than as a room filled with individual systems.

Sun's next advance in server virtualization is a concept called *Solaris™ containers*. Solaris containers isolate software applications or services using flexible, software-defined boundaries. These applications can then be managed independently of each other, even while running in the same instance of the Solaris™ Operating Environment.

Solaris containers create an execution environment within a single instance of the Solaris Operating Environment, and provide:

- *Full resource containment and control* for more predictable service levels
- *Fault isolation* to minimize fault propagation and unplanned downtime
- *Security isolation* to prevent unauthorized access as well as unintentional intrusions

This environment also simplifies resource usage accounting so that granular and extensive tracking of resources can support capacity planning and advanced client billing models.

It is Sun's plan that Solaris containers will become the fundamental, ubiquitous management object in the Solaris Operating Environment, and will be used throughout Sun's entire product line. This common approach will simplify service provisioning and make it easier to consolidate applications onto fewer servers without concern about resource constraints, fault propagation, or security. The ubiquitous management model will also simplify service-level management by permitting end-to-end services to be managed as a single unified object. The service management environment will be flexible and dynamic, and yet easy to understand and manage.

The primary benefits of Solaris containers will be:

- *Reduced management costs* through server consolidation and end-to-end service-level management
- *Increased resource utilization* with dynamic resource reallocation between containers
- *Improved service availability* by minimizing fault propagation and security violations between applications
- *Increased flexibility* because these software-based containers can be dynamically reconfigured

Businesses can realize a significant reduction in total cost of ownership without sacrificing service levels through this intuitive management model.

Intended Audience

This paper describes a vision and an approach to resource management in a distributed computing environment. The intended audience is IT managers, architects, and executives in organizations that are implementing large-scale, distributed computing solutions.

CHAPTER 2

Manageability — The Top Layer in the IT Hierarchy of Needs

As IT systems have evolved over the years, so have the expectations of both end users and IT staff. With each advancement in technology, new expectations are generated, and previous breakthroughs are viewed as standard features that are soon taken for granted. This idea can be illustrated as a pyramid, where the latest innovations act as a foundation for new developments (Figure 1).

Early information systems were judged a success if they simply solved a business problem. It didn't matter if it required a team of programmers to maintain the system. As long as the business benefits could justify the cost, the systems were considered good investments. Figure 1 shows this as the bottom layer in the hierarchy of needs for IT systems.

As IT matured, more users were given direct access to systems, and managers began to depend heavily on computing to conduct daily operations. This is when availability and scalability (the middle two layers in the pyramid) became key concerns of IT managers. The net benefit of building availability and scalability into the system was measured by counting the lost productivity from either downtime or poor performance. Now, methods of implementing availability and scalability are becoming well understood. By replicating key software processes across a multitiered architecture of hardware servers, both availability and scalability can be incrementally expanded.

The current focus is on obtaining additional availability or scalability *in the most efficient way possible*, with users taking for granted that their IT services will deliver adequate performance and always be available. However, when more components are replicated throughout an IT architecture to give it greater resiliency and throughput, the result is a sprawling, complex network of systems that are costly and difficult to manage. For more information on how the Sun Service Point Architecture addresses these needs, please refer to the *Sun Service Point Architecture—Delivering Services on Demand for the Networked World*[1] white paper.

1. http://www.sun.com/software/whitepapers/wp-spa/wp-spa.pdf

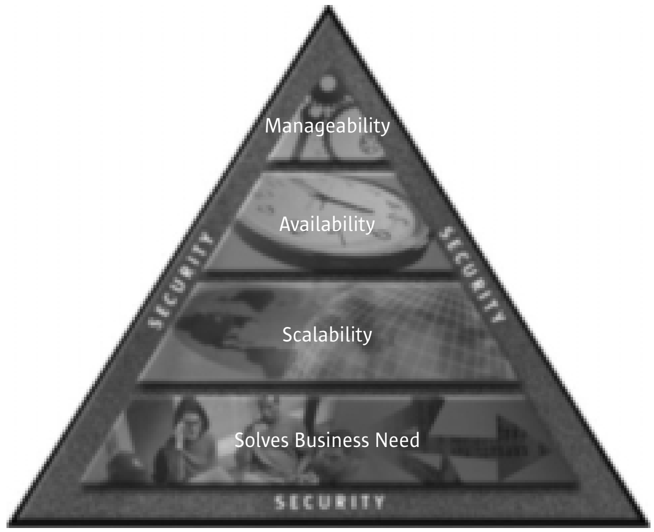

FIGURE 1 Hierarchy of needs for IT infrastructure

Businesses can decrease costs by improving resource utilization and lowering system management costs. Current capacity planning often includes extra capacity to handle occasional peak loads. For example, brokerage firms often design their systems to handle trading volumes far above those that have ever been experienced in order to capture as much revenue as possible if a spike in demand occurs. This means that normal trading activity levels may only require a small fraction of the total available capacity.

By allowing other applications to borrow resources from the trading systems when those resources are not being utilized, a much more cost-effective implementation can be realized. In the event of a spike in trading volume, resources would be dynamically reallocated so that the less-critical applications might experience delays, but the primary trading system would function as needed. Sharing resources in this way improves resource utilization, reduces total capital costs, and lowers system management expenses by reducing the total number of systems to be managed. The struggle to make IT operations more efficient has thus turned the focus of IT managers to *manageability,* the top layer of the pyramid.

In order for server consolidation to be truly effective, businesses must still be able to manage each application independently. This requires the ability to control resource utilization, isolate faults, and manage security between multiple applications on the same server. In other words, it requires the establishment of virtual server boundaries within the server.

CHAPTER 3

SERVER VIRTUALIZATION

The physical boundaries between individual systems are becoming less relevant. The service delivered to a client is increasingly based on applications that are distributed across multiple servers or are sometimes contained within a partition on a single server. It is these logical boundaries between applications that have become important. The resources available to an application can now be defined by management software that allocates system resources such as CPU, memory, and I/O to the application. Servers are, in effect, becoming virtual spaces in a fabric of networked resources. Each virtual space is an independent computing environment with logically defined security boundaries, fault isolation, and control over resources. This new flexibility will continue to change the way applications are deployed and how IT operations are managed.

The lesson learned from mainframes years ago was that if an enterprise has application services with varying peaks and valleys of use, it is more efficient to aggregate these applications together and share resources. Through such consolidation, resources can be reallocated on the fly instead of having dedicated, but infrequently used, hardware to support each of these application services. Using this approach, resources can be moved to the point of attack when and where they are required. If done correctly, businesses can reduce the total expense of their computing infrastructure through reductions in the cost of capital equipment, maintenance, and system management.

Mainframes were also the first to divide server resources into partitions to isolate applications from one another. This enabled customers to protect production environments while system updates or revised applications were tested in an isolated workspace that could not affect the production system. In 1996, Sun took this idea one step further when it introduced Dynamic System Domains technology.

With Sun's Dynamic System Domains, each domain runs its own copy of the Solaris Operating Environment and provides fault isolation so that system or application errors cannot impact applications running in other domains. A key difference between these domains and the early mainframe logical partitions technology is that *domains can be dynamically partitioned*. In the event of a resource shortage in one domain, the system can automatically borrow resources from another. This adds great flexibility to the data center while maintaining security and isolation from faults in other domains.

Even higher levels of management efficiency could be achieved if these isolation capabilities could be obtained without requiring separate instances of the operating system that must be managed and monitored.

Solaris Containers: The Next Step in Server Virtualization

Today, Sun is taking the server virtualization theme one step further by allowing domains to be partitioned to sub-CPU granularity using Solaris containers. Solaris containers will provide the basis for a completely new approach to managing an IT infrastructure by allowing the data center to be treated as a fabric of interconnected computing resources that can be flexibly partitioned into isolated execution environments for application services. This will provide great flexibility in provisioning application services because the application environment will be transportable from one server partition to another with minimal management overhead. Application services will have an isolated execution environment wherever they are provisioned, so it will also be easy to consolidate applications onto fewer servers.

In addition, the use of a common model for partitioning will simplify service-level management of end-to-end application services. Sun expects to see development of high-level service management applications that enable management and monitoring of the end-to-end services and provide detailed resource usage accounting for both application components and end-to-end services. This new management paradigm is expected to achieve significant reductions in total cost of ownership by creating management efficiencies that reduce administration costs through automation, while at the same time providing greater control over end-user service levels.

A Solaris container is a complete execution environment for a set of software components. The container provides a virtual mapping from the software components to the platform resources and allows application components to be isolated from each other even though they share a single instance of the Solaris Operating Environment.

Solaris containers establish boundaries for resource consumption (such as memory or CPU time) and provide fault isolation as well as security isolation. As processing requirements change (for example, an unexpected world event occurs and causes a surge in hits against a news-oriented Web site), one or more of the boundaries of the container can be expanded to accommodate the spike in resource consumption. Fault and security boundaries are maintained when resource boundaries are updated whether the update is done by an administrator or through predefined policies that result in automatic updates when certain conditions are met.

CHAPTER 4

THE EVOLUTION OF SOLARIS CONTAINERS

A Solaris container is made real by application of resource, fault, and security management policies. Beginning with the release of the Solaris 9 Operating Environment, Sun resource management applications will have many of the features necessary to create resource boundaries. For example, resource management tools will provide the ability to allocate resources such as CPU, physical memory, and I/O bandwidth within a single instance of the Solaris Operating Environment. Thus, resource-oriented containers will be possible with the Solaris 9 Operating Environment, allowing businesses to achieve many of the benefits of containers while additional features are under development by Sun.

As Solaris containers continue to evolve, fault isolation and security isolation are expected to be added so that the containers can provide a level of protection similar to that of a Dynamic System Domain, but with finer granularity. If a fault occurs in a user-level process, the container boundary would then prevent propagation of the failure to other containers. A Solaris container is also expected to provide isolation at the system administrator level. A unique “root” user would exist for each container, and role-based access control would provide the ability to restrict administrators from crossing boundaries. For example, the system administrator for container A would not be able to see what is happening in container B.

High-level service management applications are expected to provide a user interface that enables administrators to manage and monitor complete end-to-end application services. Even when application services are distributed across multiple Solaris containers and multiple physical systems on the network, the service management application would present a unified view to an administrator. The detailed tasks of establishing and managing container boundaries through interaction with lower-level resource management applications would eventually become transparent to system administrators, and the high-level service management application would become the single management interface.

Example: The Dynamic Aspect of Solaris Containers

To clarify the concept of a Solaris container boundary, it's helpful to look at a simplified example. Figure 1 illustrates a Solaris container as a three dimensional box. The three axes in the graph represent the available processor, I/O (bandwidth), and memory resources for the container. In practice, Solaris containers can have many different types of boundaries and will be multidimensional.

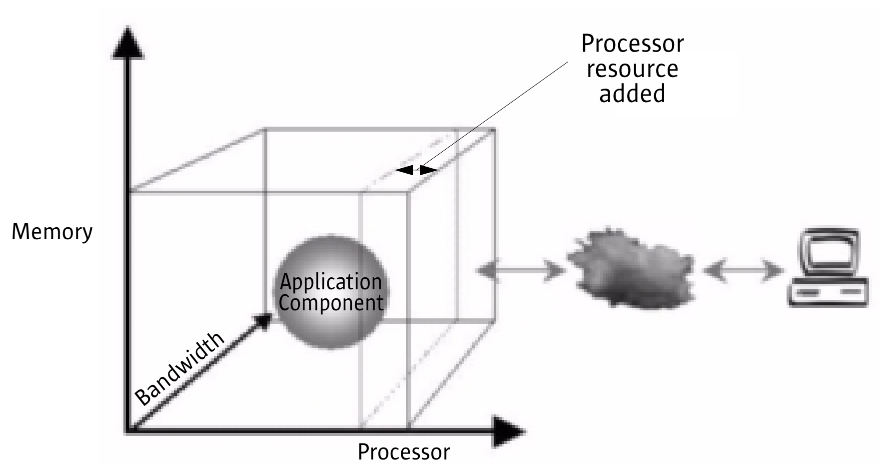

FIGURE 1 Expanding a Solaris container

In this example, the *processor* side of the container represents a specific number of processors or a specific number of shares of the total processor resource. An expansion of the available processor resources is shown by adding volume to the right side of this cube. This additional compute resource would be enabled by changing the limit on processor usage that was previously established using Solaris projects or Solaris 9 resource management software.

The *memory* axis of the Solaris container shown in Figure 1 represents a specific amount of physical memory and would also have been defined using Solaris 9 resource management software. The *bandwidth* axis represents a specific network bandwidth rate, or a portion of available bandwidth capacity.

It is important to note that manipulating one dimension of the container may impact an application service's resource requirement in some other dimension. For example a component may appear to be CPU bound because it uses all of it's processor shares in that dimension. However, it may be that this can be resolved through the increase of the physical memory dimension of the box to prevent memory paging from consuming the CPU. The resource management software would be aware of this and factor it into its dynamic resource allocation strategy.

In addition to resource control, fault isolation, and security, other requirements could also be defined for Solaris containers. For example, availability requirements, failover semantics, and maximum number of user processes would be defined in the container to permit automated control of the workload and its service availability. Defining failover semantics for an entire container offers a key advantage: a group of related processes could all be failed over together. The example below illustrates how Solaris containers can simplify resource management in a cluster failover scenario.

Example: Using Solaris Containers in a Clustered Environment

Solaris containers will be important in a clustered environment such as that provided by SunPlex™ systems. One of the benefits of clusters is in aggregating resources to provide a higher-level view of those resources to system administrators. Clusters can be thought of as a single system that has optimized and highly available resources and can manage failover processing for application services. SunPlex systems will provide a highly available platform for managing containers that host highly available services.

Figure 2 shows an example of a three-tiered application that includes a database service, an application server service, and a Web server service. These services are mapped onto seven separate Solaris containers that are layered across four separate domains in the clustered environment. The domains might be spread across two or three physical servers in a SunPlex system, but their physical layout is only a relevant with regard to establishing an appropriate level of availability and redundancy.

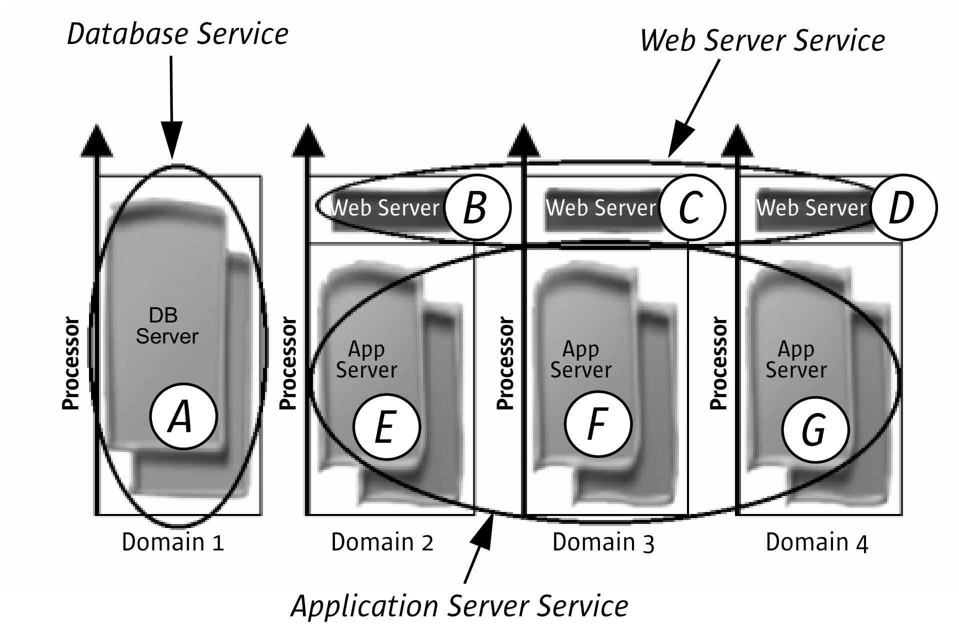

FIGURE 2 Using Solaris containers in a clustered environment

The *database service* consists of a primary database instance running in a single Solaris container shown as *Solaris container A* (marked by a circled A) in the diagram. Solaris container A happens to map exactly onto a single domain where the operating environment has been optimized for the database server. Thus, all of the domain's resources are allocated to a single Solaris container. Since the database server is not spread across multiple domains, the only way to add resources to Solaris container A is to add capacity to the domain in which it runs. Depending on the configuration, capacity could be borrowed from another domain through Automatic Dynamic Reconfiguration (ADR) in the event of a decrease in database performance.

The *Web Server Service* and the *Application Server Service* are each comprised of multiple instances of their respective software servers. These software components are mapped onto individual Solaris containers that are spread across three hardware domains in this example. These containers could be resized through dynamic borrowing of resources from other containers, or through addition of physical resources (such as CPU, memory, etc.) that are allocated to the container. It would also be possible to provision new containers by carving out resources from other existing containers.

Given the flexibility of this environment, there will be many ways to handle a failover. An example is shown in Figure 3.

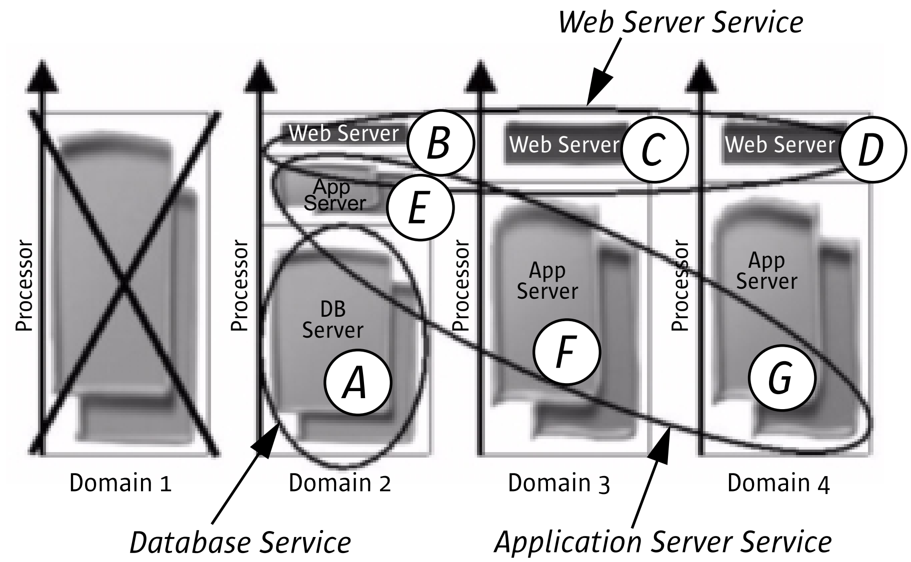

FIGURE 3 Cluster example after failover of the database server

In the event of a failure in the database domain, for example, the SunPlex system will be able to failover the database container (container A) from Domain 1 to Domain 2. To make room for the failed over database server instance, the other two containers in Domain 2 (B and E) would be resized. A service management application might also continue to monitor and adjust each of the containers to minimize bottlenecks and maximize overall throughput in support of predefined service-level policies. Throughout this activity, fault and security boundaries would be maintained with each container no matter where it is physically hosted.

CHAPTER 5

SUN™ PRODUCT DIRECTIONS FOR SOLARIS CONTAINERS

Sun has chosen the Solaris container approach because it changes the management paradigm and establishes an interconnected fabric of compute resources on the network. This allows for an integrated and comprehensive approach to service-level management, and is expected to provide efficiencies in management as well as offer greater flexibility in provisioning application services.

Because Solaris containers isolate multiple execution environments in a single instance of the operating system, they extend the concept of server virtualization down to a sub-CPU level of granularity without unnecessary performance or management overhead. Some other vendors are approaching the need for containment by running separate copies of the operating system on top of a virtual machine. This means that each instruction executed from a contained application must pass through the virtual machine as well as through the operating system, which creates unnecessary system overhead and adds to management complexity by requiring administration of extra copies of the operating system.

In contrast, multiple Solaris containers can run in a single copy of the Solaris Operating Environment to achieve isolation without the performance overhead of a virtual machine and without adding complexity to the management environment.

Sun's first step toward Solaris containers comes with the planned release of the Solaris 9 Operating Environment. Solaris 9 software will provide detailed control of system resources as discussed in the Solaris container examples in this paper. Following that, Sun's intention is to integrate the Solaris container concept throughout its software and system management product lines and to enhance the Solaris Operating Environment to provide fault isolation and security isolation for Solaris containers.

Sun™ Management Center software will continue to be enhanced with the intent to provide a high-level service management facility that automates many of the container management tasks and removes the administrator from management details. Future versions may allow resource management policies to be set from within a service management console that also enables administrators to define Solaris containers and assign application services to specific containers as part of managing an end-user service. Sun's vision is that Sun Management Center software will provide oversight of other Sun management tools to establish the necessary Solaris container boundaries so that these details will be transparent to the administrator.

Sun intends to enhance all of its products to support this new management paradigm while continuing to provide application programming interfaces (APIs) that promote open standards. Through support of open standards, Sun promotes further progress in its rich portfolio of powerful third-party management solutions.

CHAPTER 6

A Future View of Solaris Containers and Service-Level Management

Creating boundaries that limit an application's resource consumption is not a new concept. In fact, existing resource management tools provide methods to establish these boundaries. What is new is the idea that containers can provide multiple independent execution environments within the same instance of the operating system, resulting in a dynamic, predictable, secure, and highly manageable IT infrastructure.

A uniform container environment could be provided across all Sun systems so that the hardware platform becomes transparent to software components in the containers. A software component could be just as easily deployed on a workstation as on a large server as long as the environment in which it is deployed supplies the appropriate resources for the container. This hardware independence in the context of a networked environment would provide a new level of flexibility, enabling system managers to more easily provision, migrate, and consolidate applications. They could manage service levels without having to worry about security, faults, or resource allocation for individual processes.

The possibilities for future capabilities with Solaris containers are numerous:

- Management and accounting features could allow tracking of complete services that span multiple Solaris containers. The system could compare performance of a service against desired goals and dynamically reallocate resources as needed to maintain service levels. If additional resources must be added to the system, the system operator can be alerted.
- When a system administrator creates a service container for a Java™ 2 Platform, Enterprise Edition (J2EE™) environment, a Solaris container could be automatically created at the same time. This would not only create efficiency in systems management, but also would simplify billing and charge-back by enabling detailed use statistics for individual software components as well as for end-to-end services.
- Applications could be completely independent of storage devices. The layer of abstraction provided by a virtual namespace and virtual file system would make this possible. Just as in a SunPlex environment today, once an application is installed anywhere in the framework, it could be executed on any other node in the framework.

- With detailed accounting of resources for end-to-end services, a utility model for software licensing is possible. For example, software could be priced based on how much CPU time is used by an end-to-end service, even if that service spans multiple Solaris containers. No longer will it be reasonable to license software based on the size of the target system. The software may not have access to the entire system when Solaris containers are in use.
- Provisioning tools such as Solaris Web Start Flash software will no longer need to copy and replicate entire system images, but instead will be able to replicate individual Solaris containers.

CHAPTER 7

MANAGEABILITY — THE VALUE OF SOLARIS CONTAINERS

Solaris containers will have a significant impact on total cost of ownership for IT systems. They will enable businesses to better manage large-scale systems through consolidation without concern about resources, faults, or security. This will result in greater systems management efficiency and improvements in resource utilization. Solaris containers will also change the management model to be more service-centric and allow more standardized automation of underlying resource and system management tasks.

The primary benefits of Solaris containers include:

- *Reduced management costs* — Applications will be more safely consolidated onto fewer servers, resulting in reduced management complexity and reduced management costs. Furthermore, the intended standardization on the Solaris container model throughout the Sun product line will simplify service provisioning and reduce training and reconfiguration expenses.
- *Increased resource utilization for lower cost of operations* — Dynamic resource reallocation will improve resource utilization by allowing unused resources to be shifted to other containers as needed. In addition, with fault and security isolation of Solaris containers, poorly behaved applications will no longer require a dedicated and underutilized system, but may be consolidated with other applications.
- *Improved service availability* — Fault isolation and security isolation protect applications from error propagation as well as intentional or unintentional intrusions that can affect performance or availability. The greater consistency and simplicity of this environment also reduces the chance of human errors that might impact availability.
- *Increased flexibility* — Solaris containers will make it safer for businesses to implement server consolidation with less worry about applications being affected by resource constraints, faults, or security issues. In addition, Solaris containers will be easy to reconfigure, and will greatly simplify the task of migrating applications from one set of hardware to another.

As an example of the flexibility of Solaris containers, consider that a four-CPU Solaris container on a Sun Fire™ 3800 server could look identical to a four-CPU Solaris container on the larger 106-processor Sun Fire 15K server. Thus, an administrator could freely move an application from one system to the other while knowing that all of the properties of the Solaris container would move with the application.

CHAPTER 8

CONCLUSION

In order to prosper, businesses must not only maintain high service levels to satisfy end users, but also must reduce their IT expenditures. These seemingly opposing objectives require a more elegant approach than just watching the budget. Solaris containers provide a leap forward in manageability that will enable businesses to reduce costs through improved efficiency and better resource utilization. This new approach will enable cost savings while continuing to meet the increasing demands of end users for high service levels.

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303

1 (800) 786.7638
1 (512) 434.1511

sun.com/solaris

February 2002