

Right Architecture for the Right Workload:

The Application Tier

Tom Atwood

Enterprise Systems Products

Architecture Evangelist

Sun Microsystems, Inc.

Abstract

Most application services are being deployed in an N-tier environment. Tier-3 is the database tier, tier-2 is the application tier and tier-1 is the presentation or edge tier. When deploying application services users must decide on what type of architecture, horizontal or vertical (or scale out or scale up) they will use in deployment. Vertical architecture is common in tier-3 and horizontal is common in tier-1, however, what about tier-2, the application or business-logic tier? This paper looks at in detail at technical and business issues of tier-2 deployments using both horizontal and vertical architectures. The purpose is to examine several common and important applications such as SAP, PeopleSoft and Oracle Applications and make recommendations as to what server architecture, horizontal or vertical, is best for these applications or application types.

Introduction

IT managers today are trying to balance the cost of acquisition, performance and manageability in deploying key applications.

It is critical that they have good technical and financial information about various options. When deploying application instances there are many options including using a horizontal architecture of many smaller servers, a vertical architecture of one big server or a compromise of a few medium servers in a “diagonal” architecture.

This paper looks at the technical differences of horizontal, vertical and diagonal, architectures and analyzes the costs of acquisition. The goal is to provide information and recommendations that will assist IT managers in choosing the right architecture for their application tier deployment. Regardless of what type of architecture is chosen Sun can provide the solutions for any architecture. From a technical and financial point of view a diagonal architecture may be the best solution in most cases but both the horizontal and vertical architectures are very strong options.

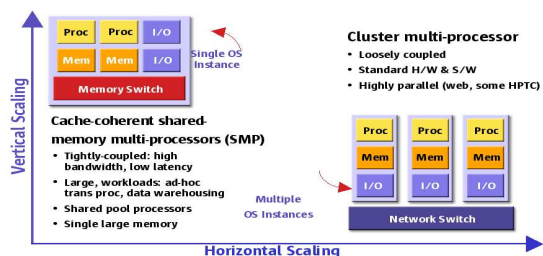
1. Architectural Definitions

It is important to understand the differences between horizontal and vertical architectures. This paper will touch briefly on the architectural differences focusing on those differences that are important to the application tier. For a more detailed description of the differences between vertical and horizontal architectures please refer to the SUPeR Berlin 2003 paper entitled “Attributes and Trends of Horizontal and Vertical Computing Architectures”.

A Sun whitepaper version of this SUPeR paper can be found at: <http://www.sun.com/servers/wp/search/search.cgi/>

Figure 1 illustrates the differences between vertical and horizontal architectures. Vertical servers are large SMPs (symmetric multiprocessing or shared-memory processing). There is only one instance of the OS covering the processors, memory and I/O components. In general all of these resources are contained within a single chassis or box.

Figure 1: Horizontal and Vertical



The interconnect is tightly-coupled (commonly a centerplane or backplane), low-latency, high-bandwidth and cache coherent (maintains info on location of all data regardless of cache or memory

location). More resources are added inside the box by adding system boards into the system. On vertical or SMP systems the memory is shared meaning that all processors and all I/O connections have equal access to all memory. Memory appears to the user as one large chunk.

Horizontal architecture refers to groups or clusters of smaller servers that are used to run one or many databases or applications. These servers, are connected together using external network interconnections. These interconnects are commonly standard network interconnects such as Fast Ethernet, GBE, SCI, etc. They are much lower bandwidth and higher latency than vertical systems. Resources are contained within “nodes” that commonly have from 1 to 4 processors. Each node has its own processor and memory and may have its own I/O or may share I/O. Each node has an OS instance. Resources are increased by adding more nodes and not by adding more resources within a node.

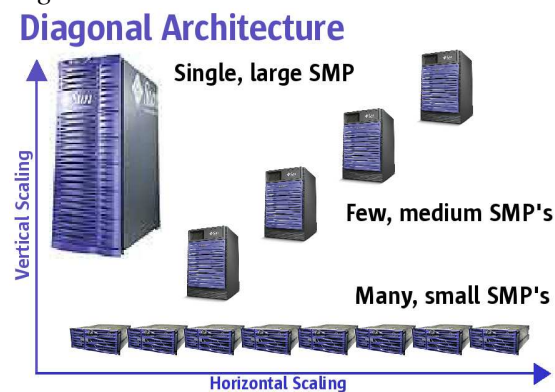
The memory in horizontal systems is distributed meaning that each node or blade has its own memory that is directly accessed only by the node's processor and I/O connections. Any access to these resources by other nodes is much slower than access by the node itself. Also, there is no cache coherency between nodes. Cache coherency must be handled by the applications if more than one node is needed for the application. Commonly the applications deployed on horizontal architectures are small enough to fit within a single node so cache coherency is not needed.

Application instances are normally deployed on the application tier in such a way that inter-node communication is not necessary. In that case the concept of vertical and horizontal architectures takes on a less

technical meaning. Specifically, for the application tier, horizontal architecture means that application instances are deployed on many small servers (4 processors or less). Each application instance is small enough such that it fits within a given node and there is no inter-node communication. The number of nodes is dependent on the size of the node (how many processors). The concept of vertical scaling stays basically the same where all instances of an application are deployed on one large SMP server.

With regards to the application tier there is another architecture that is commonly used. This third architecture is called “diagonal scaling”. Diagonal scaling combines vertical and horizontal scaling to take advantage of the best of both architectures. In diagonal scaling applications are deployed on 2 or more medium to large SMP servers. For the purpose of this paper the minimum size server will be more than 4 processors. Figure 2 shows all three server architectures, vertical, horizontal and diagonal. Effective data centers will most likely have a mix of all three architectures. Sun’s binary compatible server product line that starts at one processor and continues up to 72 processors enables solutions using all three architectures.

Figure 2: Architectures



2. Application Tier Attributes and Requirements

Most data center application deployments use an N-tier architecture which commonly has 4 tiers. Tier 0, or the client tier, is the tier where the users reside. Client systems can include PDAs, desktop and laptop computers, point-of-sale devices, cell phones, etc. These are the devices that users use to access a given application. Tier 1, the presentation layer, is also called the “edge” tier. This tier is where the user “enters” the application normally via a web interface. This tier is where applications like web servers, authentication servers, proxy servers, etc. are deployed. This tier directs the users to the appropriate business logic that resides on Tier 2, the application tier.

The application tier in a three-tier data center environment is stateless meaning that little or no data is stored on the servers themselves. This layer has the business rules for the application service. Transactions are received by the presentation layer and processed by the application layer. When data needs to be read or written the transaction is passed to the database layer. Application servers tend to consolidate connections to the database. This is useful because too many connections to the database can affect performance. A notable exception to this is the Oracle Applications suite which does not consolidate connections.

In most cases the application server layer requires more cpus than the database layer for each application service. In the case of SAP R3 this ratio is about 3 processors for each database processor. So, an SAP R3

implementation that requires 20 processors for the database will require about 60 processors for the application layer. The question is does one deploy 30 2-processor servers, 1 60-processor server or something in between? What are the technical and financial ramifications of various deployment options?

The database tier, or Tier 3, is where the data resides in relational databases such as Oracle, DB2, Sybase and others. This tier is stateful, meaning that this is where the data of the application is found. Processing requirements and availability requirements are best met with single images of the database. For more information on deploying databases in Tier 3 please see the Sun whitepaper "*Attributes and Trends of Horizontal and Vertical Computing Architectures*" which can be found at: <http://www.sun.com/servers/wp/search/search.cgi/>

The application tier or Tier 2 has some important attributes that need to be considered when deploying applications. They can be a mix of OLTP and batch. They are commonly memory intensive requiring quite a bit of memory per "user". Application instances are generally not dependent on each other meaning that the instances running on various nodes do not need to communicate with each other. This removes a key issue that can affect scalability of applications deployed using a horizontal architecture in the database tier and that is the relative slowness of the network interconnects between nodes (servers). Many popular applications today scale well and can use the resources of larger vertical servers.

A successful deployment of the application tier must meet the following requirements:

Performance:

Users will have service level agreements (SLA's) that require that applications be able to support a given number of users, with a minimum amount of data and within a reasonable response time. Any architecture must be able to provide the performance required.

Availability:

Application availability is a must. The application must be up whenever the users need it.

Acquisition Cost:

The acquisition costs must be reasonable but must not compromise SLA's.

Management Costs:

The architecture must be straightforward to manage so that management is not too costly. At times there is a trade off between management costs and acquisition costs as well as availability and performance.

The goal is to find the best mix of all four requirements from both a technical and a financial perspective.

3. Application Tier Resource Utilization

As will be seen later in the financial analysis section of this paper improving resource utilization can reduce acquisition costs. Clearly if one can increase individual server

utilization fewer servers will be needed thus reducing acquisition costs. This section will look at some issues regarding resource utilization and how to improve it.

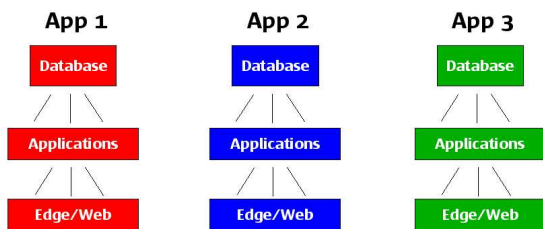
Application Tier Topology

A good way to improve resource utilization is to pool or share resources. The concept of one application instance or one database instance per server can lead to inefficiencies as each server must be configured for the peak load. But when the application is not at peak load many of the resources are under-utilized. Thus it is useful to figure out ways to consolidation more than one application instance on each server.

When deploying many applications such as PeopleSoft, SAP and Oracle Applications each module, such as PeopleSoft's Human Resources module, will have some components that run on tier 1, the presentation layer, tier 2, the application layer and tier 3, the database layer. One way to deploy many modules would be to set up separate database servers, application servers and webservers for each module with no sharing between modules (figure 3).

Figure 3: Complete Separation

Deployment Topology: Complete Separation

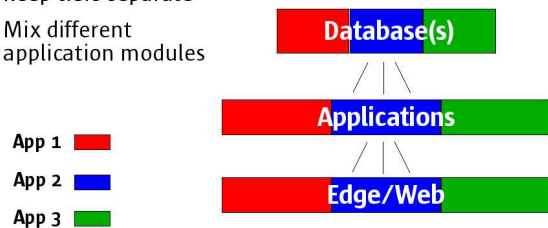


However, a more efficient way to use resources is to mix and match application tier instances and run more than one application tier instance on a server. If the application instances are matched properly per server resource utilization can be increased. Figure 4 shows consolidation of the applications by tier instead of by modules. This increases the likelihood of being able to run multiple instances of a given tier on given servers.

Figure 4: Consolidated deployment

Deployment Topology: Tier Consolidation

- Keep tiers separate
- Mix different application modules

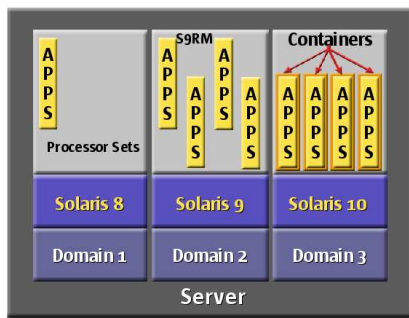


Resource Management Tools

To be able to effectively run multiple application tier instances (and multiple database tier instances as well) on one server it is necessary to have good resource management tools. Solaris™ tools such as processor sets, Solaris™ 9 Resource Manager (S9RM) and, new in Solaris™ 10, N1™ Grid Containers (containers) can improve resource utilization. Processor sets and S9RM partition processor resources. They make sure that a given application has at all times at least the minimum amount of resources necessary to meet performance requirements even if it is sharing a server or domain with other application instances.

N1 Grid Containers will improve greatly on S9RM and processor sets. Containers add security isolation and fault isolation between containers. Each container has its own IP address, own file system, own access privileges and can reboot separately from other containers. The size of each container can be smaller than a single processor. All of this isolation is provided on one copy of the OS so management is simplified as well. An OS update for many containers is as easy as updating the OS for the server or domain that contains those containers. Figure 5 shows domains, Solaris 9 Resource Manager and N1 Grid Containers.

Figure 5: Resource Management
Granularity at Every Level: Hardware, Operating System, and Application



Login Persistence and Load Balancing

When deploying many application tier instances in most cases they will be deployed on more than one server. In that case a critical factor in resource utilization is the the ability of the application to load balance the many users that login to an application.

In the case of SAP and PeopleSoft and Oracle forms-based modules of the Oracle Applications suite users are pointed to the least busy server when they login. Thus, if there are 10 servers in the application tier the first 10 users will probably be assigned one at a time to each of the 10 servers (figure 6). The next 10 users will be assigned to the least busy servers. This may lead to different numbers of users logged in to each server as some servers will be busier than others (figure 7).

Figure 6: Login Persistence
Login Connection Persistence

- Users connected to least-busy server

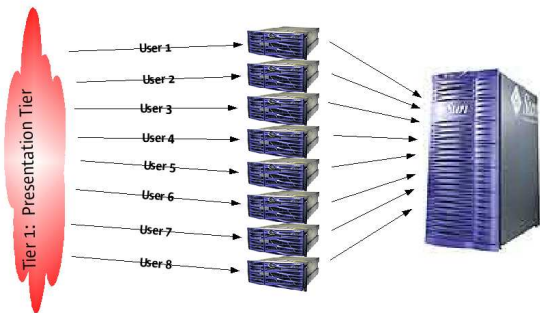
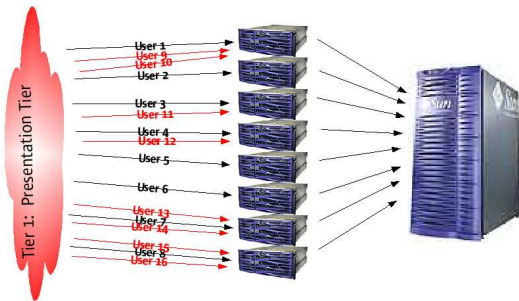


Figure 7: Login Persistence

Once a user is logged in to a physical server they stay connected to that server and cannot dynamically move from a busy server to a less-busy server. Not all applications have this issue but SAP, PeopleSoft and forms-based Oracle Applications cannot change servers dynamically. The user needs to logout and log back in. Oracle application modules that have an “i” in front can take advantage of java-based middleware that enables dynamic load balancing.

Login Connection Persistence

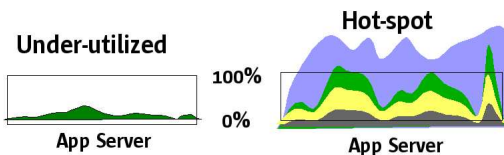
- Users connected to least busy serve
- Users remain connected until logout



Thus, with many servers it is possible to have some servers that are overloaded or are “hotspots” that affect performance while other servers are sitting idle or underutilized (see figure 8). To avoid server “hotspots” each server needs to be large enough to handle any possible peak load and the highest number of application users. Or there needs to be enough servers so that no one server ever has too many users logged in. This can lead to inefficiencies as the need to deploy larger numbers of servers can lead to lots of underutilized resources. The smaller the server the more are needed and hence the fewer users or the fewer application instances per server.

Figure 8: Server Hotspots
Implications of Login Persistence

- Users cannot migrate dynamically
- Some servers may very busy
- Some servers under-utilized
- Need to configure headroom to avoid “hot-spots”

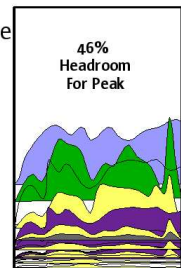


Whenever a server is configured it will be configured with some amount of headroom. Unfortunately, such headroom cannot be shared between servers in the application tier. Consolidating many servers into fewer larger servers consolidates headroom and allows it to be shared thus improving system utilization and requiring fewer total processors to be deployed. The more servers are deployed there is an increase in the amount of headroom resources that cannot be shared.

A way to improve load balancing and overcome some of the issues of login persistence is to use larger servers for the application tier and let Solaris do the load balancing. Solaris is very efficient at scheduling many processes among all the processors that exist in a given server or domain. Processes can dynamically migrate from one processor to the next based on workload. Thus, a large vertical server is essentially a pool of compute resources that are assigned as needed to the many users and application instances that reside on that server (figure 9). The ability to use Solaris to load balance can lead to the need for fewer processors and memory and lower acquisition costs.

Figure 9: Solaris Load Balancing
Load Balancing Via Solaris

- Solaris does the load balancing
- Logon persistence less of any issue
- Less chance of “hot-spotting”
- Better resource utilization
- Use S9RM and processor sets
- Consolidate and share headroom
- Fewer processors needed



A paper by Dr. Christian Tolkes of Sun's SAP Competency Center describes in more detail the possible reduction in hardware that is possible when using larger servers. The conclusion of this paper is that for the reasons discussed above it was possible to deploy SAP with 50% fewer processors using an Sun Fire™ 6800 vs smaller 2-way servers. For more info please see:

Consolidated SAP Solutions—A Review of Costs,” Dr. Christian Tolkes, <http://www.sun.com/servers/wp/search/search.cgi/>

64-bit or 32-bit Servers?

When looking at options for lowering server acquisition costs many look at 32-bit platforms as an option. However, many application tier instances have high per user memory requirements. Oracle applications, for example, has a per user memory requirement that ranges from 5MB up to 80MB with an average of 20MB. 64-bit systems, of course, can support significantly more memory per processor and per application process.

Thus, 64-bit systems are strongly recommended to meet the memory requirements of the application tier. While memory requirements can vary from one application to the next SAP, PeopleSoft and Oracle Applications are three examples of systems with memory intensive memory requirements. Regardless of what general architecture is used to deploy the application tier 64-bit servers are preferred.

Discussions with Sun's SAP competency center and Sun's Oracle Advanced Technology Center (SOATC) indicates that

64-bit servers of 4 processors or more are preferred based on the nature of those applications. Servers with 2 processors or less can hotspot easier (a single batch job for SAP can overwhelm a 2 processor server). It seems reasonable to assume that many other applications behave similarly to SAP, PeopleSoft, and Oracle and that such recommendations are appropriate for other applications.

5. Application Tier Architectures

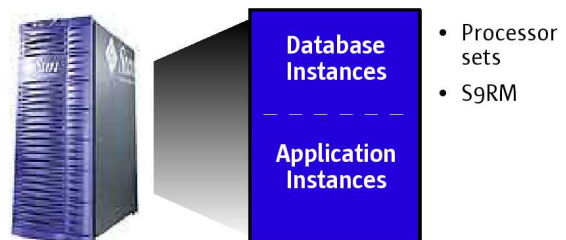
This section examines four different architectures that may be used in deploying application instances in the application tier. Each option will be described and then evaluated against the technical criteria previously discussed. The evaluation based on financial factors will be later in the paper.

Vertical 2-tier:

This option involves deploying the application instances and the database instances on a single server or domain of large, vertical servers (figure 10). This solution promises to have the fewest OS instances and hence the most centralized management which may lead to lower management costs. In general, the more OS instances that need to be managed the higher the management costs.

Figure 10: Vertical 2-tier

- Database and app instances in one server/domain



This option has very good performance as application instances and the database instances may be able to communicate with each other internal to the server instead of over an external network interconnect. Some industry-standard benchmarks such as the SAP 2-Tier benchmark, are run using this architecture. Resource utilization is also very good as Solaris can balance resources efficiently between the various application instances and the database instances. Resource management tools such as processor sets, Solaris 9 Resource Manager and, in the future, N1 Grid Containers, can be used to run these various instances in the single domain.

However, the availability of such an architecture is problematic. If that big server or that big domain goes down then all application instances and all database instances go down. Clustering can be used to provide failover capability to improve availability but that requires twice as much hardware and can increase complexity. So, while vertical 2-tier can provide some advantages in terms of management, resource utilization and performance its lack of high availability make using this architecture in production unlikely. Note that with the software fault isolation capabilities of N1 Grid Containers that are in Solaris 10 this option may become viable as the availability characteristics will improve. Figure 11 is a summary of the above technical analysis of all four architectures.

Figure 11: Technical Summary

Architecture	Performance	Availability	Complexity
Vert. 2-tier	Very Good	Marginal	Low
Vert. 3-tier	Excellent	Good	Low
Horiz. 3-tier	Good	Excellent	High
Diag. 3-tier	Good	Very Good	Medium

Vertical 3-tier

Like the vertical 2-tier option the vertical 3-tier architecture places all of the application and database instances on a single vertical server (figure 12). However, in this option the application tier instances separated from the database instances by placing them in different dynamic system domains. Dynamic system domains (domains) are hardware partitions that are available on mid-range and high-end Sun Fire servers. They provide resource, fault, service and security isolation and enable the deployment of different application environments on one system.

Figure 12: Vertical 3-tier

- Tiers 2 and 3 in domains on single server
- Database instances tier in one domain
- Application instances in other domain(s)



- DR used to resize domains
- More than 1 domain for HA
- Mix apps in domains

By deploying the application instances in different domains and separate from the database instances the vertical 3-tier architecture overcomes the availability drawbacks inherent in the vertical 2-tier. At the same time the performance and resource management advantages of deploying many

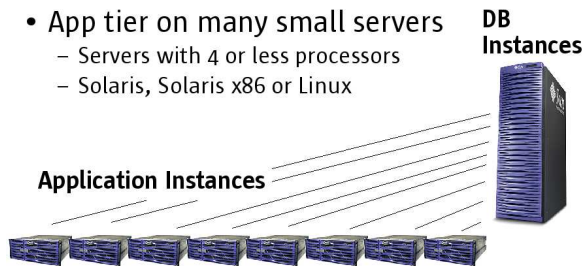
application instances on large instances of Solaris are realized. From a technical perspective the vertical 3-tier is a very strong architecture.

Horizontal 3-tier

The horizontal 3-tier architecture involves deployment of many application instances on many relatively small servers (figure 13). These small servers are normally from 1 to 4 processors in size. As mentioned previously it is best if these are 64-bit servers. Availability is very high with this solution. While each node may not have high RAS features the loss of one node is not detrimental to the production environment because of the redundancy of having many servers.

Figure 13: Horizontal 3-tier

- DB tier on vertical server
- App tier on many small servers
 - Servers with 4 or less processors
 - Solaris, Solaris x86 or Linux



Because the nodes are relatively small there may be some resource usage issues as explained earlier in the section on load balancing and persistent login. Performance itself is generally good as production environments tend to deploy as many nodes as are necessary to meet workload requirements. Management cost can be an issue, however, as the use of small nodes means that there are many instances of the OS to manage and this can become a challenge. These management challenges

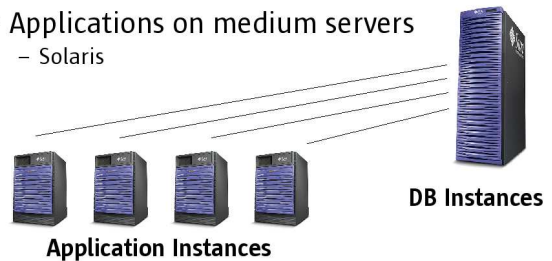
may be lessened in the future when Sun's N1™ Grid Provisioning Server becomes available for data center servers. Currently the N1 Grid Provisioning software is available only for Sun's blades products.

Diagonal 3-tier

The diagonal 3-tier architecture is very similar to the horizontal 3-tier but uses larger servers as nodes. Instead of many 1 to 4 processor servers the diagonal 3-tier architecture uses fewer but larger servers (figure 14). These servers commonly are from 8 to 24 processors. At least two servers are used for availability so the size of the node is dependent on the total number of processors required for meeting required service levels. Since the nodes are larger than the horizontal architecture the resource utilization is generally better than the horizontal option. As the application deployment gets larger (more total cpus) the diagonal option gets stronger as the node count increases and the node themselves get larger.

Figure 14: Diagonal 3-tier

- Database on vertical server
- Applications on medium servers
 - Solaris



Since there are fewer nodes than the horizontal architecture complexity is less and management are likely to be costs lower in the long term. With larger nodes capacity increases can be done within the server nodes themselves (more processors or faster

processors) or by adding more nodes. Note that there is a variant that can be called the “mixed diagonal” architecture where the nodes are of different sizes. The deployment may start with 12 processor nodes but move to 4 processor nodes if small increments of capacity are needed or could increase to larger nodes if big capacity increases are needed.

Application-specific Issues

There are some application-specific issues that should be considered when looking at the above different architectures. Oracle does not consolidate connections between the application tier and the database tier. There is a limit of about 8000 database connections per server. That may limit the size of the a server in the application tier.

6.Financial Analysis

This section looks at the various deployment architectures from a financial point of view. The financial analysis will be limited to the cost of hardware acquisition because this is easy to quantify and can be generalized. A more important aspect of cost, total cost of ownership (TCO), is out of the scope of this paper but is recommended for customers seriously investigating what architecture to deploy in the application tier. Sun and its partners can provide detailed TCO analysis for IT departments that need them.

TCO studies are best done on the customers specific environment and are hard to generalize. In general, however, the fewer number of servers the fewer OS instances to manage and the lower the management costs over time.

For more information regarding the financial impacts of using the a horizontal or vertical architecture for the database layer please see the Sun whitepaper “*Attributes and Trends of Horizontal and Vertical Computing Architectures*”:<http://www.sun.com/servers/wp/search/search.cgi/>

In order to compare the cost of hardware acquisition for various options a tool called the “app calculator” was created. This tool is a simple spreadsheet whchi calculates the acquisition of two configurations at a time. The calculator compares configurations that use small servers (called the horizontal option) and servers that are medium to large in size (vertical option). However, it should be noted that all options are “horizontal” in some way in that they include more than one server or domain (except for the vertical 2-tier architecture). The key difference between the horizontal and the vertical options in the tool is the size and number of servers. The inputs include:

Number of 1.2 Ghz US-III processors required to meet the workload.

Baseline server utilization assumed in determining the number of 1.2Ghz US-III processors.

Horizontal node price.

Horizontal node hardware discount.

Number of processors in the horizontal nodes.

Vertical node price.

Vertical node hardware discount.

Number of processors in the vertical nodes.

Relative processor performance vs 1.2GHz US-III processors.

Processor utilization for horizontal nodes.

Processor utilization for vertical nodes.

The tool adjusts the number of processors and servers needed using the relative processor performance input parameter. A processor faster than a 1.2GHz UltraSPARC® III processors will require fewer processors to meet the application requirements. The tool takes the adjusted server capabilities and calculates how many servers of that type of server are needed to meet the requirements. The cost of acquisition for both the horizontal and vertical options are calculated using the number of adjusted servers times the per server cost and less the hardware discount. These systems costs are listed in the results chart as the “Total Uncorrected Cost”.

The tool makes further adjustments based on the server utilization. The tool compares the server utilization of each server type with the baseline or “assumed” server utilization. If the server has a utilization that is different than the baseline utilization the number of processors needed to meet the requirements is adjusted. This cost based on adjusted server utilization is called the “Total Corrected Cost”.

A server utilization figure better than the assumed or baseline utilization number results in fewer processors needed to meet the application performance requirements. If the server utilization is the same as the baseline there will be no difference between the “Total Uncorrected Cost” and the “Total Corrected Cost”.

In the application tier analysis done for this paper all of the servers are Sun servers to

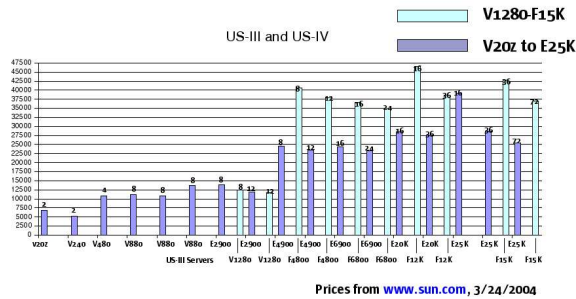
reduce the number of variables in the comparison. It is fairly straightforward to compare the relative performance of the systems thus giving reasonable adjustments. All of the system prices are U.S. list prices and were downloaded on March 24, 2004 from the www.sun.com website.

When comparing systems the 1.2GHz UltraSPARC® IV processor was assumed to be twice the performance of the 1.2GHz US-III processors. Benchmarks run by Sun and reports from early customers are indicating from 1.9 to 2.0 speedup in most cases but, of course, such speed up is application dependent. 2X speedup is used here partly for simplicity. The smallest server used was the Sun Fire™ v480 based on discussions earlier on the advantages of nodes with 4 processors or more. Servers in the survey used included only those that have been announced or are shipping and have prices available on www.sun.com.

In general smaller servers will have a lower per processor price. Figure 15 shows the per processor cost for a variety of Sun servers from the 2 processor Sun Fire™ v20z to the 72 processor Sun Fire™ E25K. The per processor cost for the US-IV Enterprise servers (Sun Fire™ E2900, Sun Fire™ E4900, Sun Fire™ E6900, Sun Fire™ E20K and the E25K) has been adjusted for the fact that those processors are 2x the performance of US-III processors. The exception is for the E20K and the E25K which have been announced with 1.05GHz US-IV processors and is about 1.75x faster than a 1.2GHz US-III.

**Figure 15: Adjusted Per Processor Cost
Per Processor Costs**

- Per processor price jump at E4900
- US-III processor cost is higher than US-IV servers



It is clear that the smaller servers have the lowest price per processor when server utilization and discounts are not applied. The Sun Fire™ v240 is the lowest followed by the v20z but both of those servers are only 2 processors. The v480 to the E2900 servers have an average per processor cost of about ½ the average per processor cost of the E4900 to the E25K and about 1/3 the cost of the US-III servers. Clearly the servers that use US-IV processors have an improved per processor cost.

The remainder of this analysis will include only servers with 4 processors and above for the technical reasons previously discussed. The analysis will focus on the cost of acquisition of various architectures where the costs may be affected by server utilization and hardware discounts. For the v480 and the E2900 the hardware discounts applied are 20% and for the E4900 and larger servers the hardware discounts are 40%.

Based on the technical discussion earlier server utilization for larger servers is higher than for 4 processor servers. For the v480

the utilization is set at 30% for all cases. For servers greater than the v480 the server utilization is set at 50%. What is more important about the server utilization numbers is not their actual percentages but their relative value. In other words, do larger servers have better server utilization than smaller servers? In general, yes, and for the financial analysis part the larger servers will be assumed to be about 2/3 more efficient than the smaller servers.

App Calculator Results

Several scenarios were run through the app calculator tool to determine the relative acquisition costs of each architecture.

The app calculator tool is designed to compare two different architectures. To simplify this analysis one of the architectures in each case is always the horizontal 3-tier. The vertical 2-tier was not tested because of the issues with availability. Thus the financial analysis will compare horizontal 3-tier against vertical 3-tier and diagonal 3-tier.

Vertical 2-tier

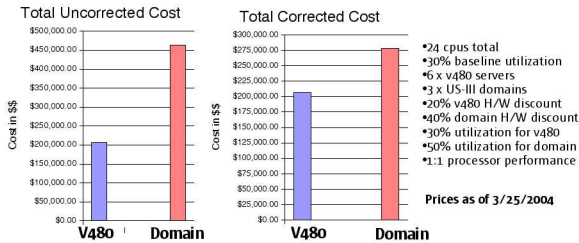
The vertical 3-tier results are shown in figures 16 and 17. The first analysis was for a domain with US-III uniboards (processor and memory boards). An 8-processor domain was used in this cost analysis because it is assumed that either the customer already has a US-III server with empty slots to create new domains or the large server was purchased primarily as a database server. Adding the application tier is a secondary consideration.

This scenario should also be of interest to IT managers that already have larger US-III servers that have some empty slots and are

considering deploying new applications in a couple of medium-sized domains or deploying those new applications on a few smaller servers.

Figure 16: US-III Domain

- V480 and 8 processor US-III domain
- 24 US-III equivalent processors



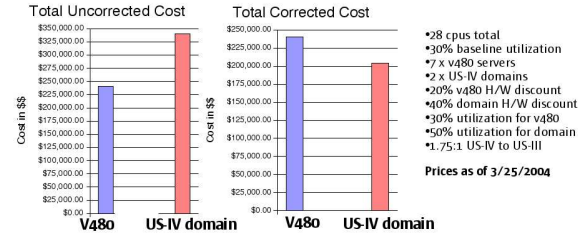
The target number of cpus was 24 which required 6 v480's or 3 8-processor US-III domains. The results from the app calculator shows that even with a better server utilization the US-III domain has about a 25% higher acquisition cost. This difference is small enough so that IT managers that are concerned about floor space, are charged by the server in outsourcing contracts or prefer the extra management and RAS features of larger systems can give this option a serious look.

The same analysis was done using domains with US-IV processors. It has already been shown that the per processor cost of US-IV-based systems is better than US-III-based systems and this analysis supports that. In fact, when applying the previously-mentioned hardware discounts and server utilization figures the 2 8-processor US-IV domains are less expensive to acquire than the 7 v480's and are less complex. Clearly the vertical 3-tier is an option assuming that the customer either has open slots in a larger server or a larger vertical server has already

been chosen as the database server. In general, the per processor price of larger vertical servers declines with more processors purchased as is shown in figure 15.

Figure 17: US-IV Domain

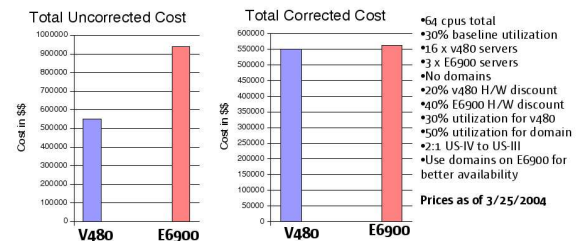
- V480 and 8 processor US-IV domain
- 28 US-III equivalent processors



The results in figure 18 can also be considered as an analysis of a vertical 3-tier architecture. In this scenario 16-processor E6900's were compared against 4-processor v480's. The total number of processors was 64 US-III's. This could be handled by 16 v480s or 3 E6900's with 16 processors. This is an example of diagonal scaling but if the extra 8 unused processors in the E6900's are assumed to be dedicated to the application database then this can also be an example of the analysis of the cost of the application tier portion of a vertical 3-tier architecture. In this case the corrected or adjusted costs for the two architectures are about the same and both are viable options.

Figure 18: 16-Processor E6900

- V480 and 16-processor E6900
 - 64 US-III equivalent processors
- "Corrected Cost" reflects "utilization" parameter



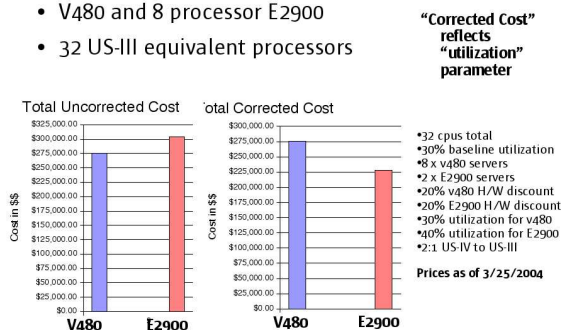
Diagonal 3-tier

Three different analyses were done comparing a horizontal 3-tier architecture to a diagonal 3-tier architecture. The difference in each scenario is the size and type of the server used in the diagonal 3-tier architecture. The horizontal 3-tier server was a 4-processor v480 in all cases. For the diagonal 3-tier the servers were an 8-processor E2900, a 12-processor E4900 and a 16-processor E6900.

Figure 19 shows the results when the diagonal server is an 8-processor E2900. The costs that are not adjusted for server utilization differences show that the E2900 option is close in acquisition cost to the horizontal option. The target number of US-III processors was 32 and it took 8 v480's and 2 E2900's. Server utilization for the E2900 was 40%.

Figure 19: 8-processor E2900

- V480 and 8 processor E2900
- 32 US-III equivalent processors

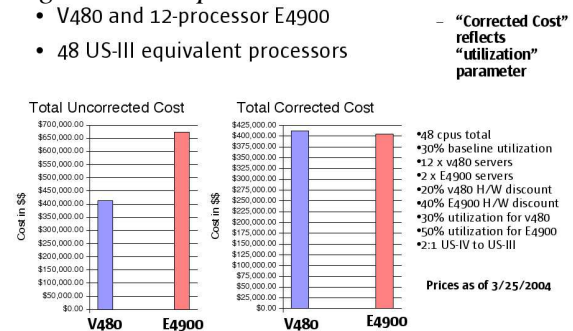


The corrected costs shows that the E2900 option is less expensive assuming that the E2900 has better server utilization. These results are as expected because as figure 19 shows the per processor cost is similar for the v480 and the E2900 before any adjustments.

48 US-III processors is the target for the scenario for the 12-processor E4900 compared to the v480. 2 E4900's and 12 v480's were required to meet the performance requirement. The server utilization for the E4900 was 50% reflecting the larger capacity of the server compared to the v480 and the E2900. The results show that the costs corrected for server utilization are about the same for both options (figure 20). Both options are technically viable and both have similar hardware acquisition costs. The horizontal option may have somewhat higher availability given the greater number of servers but that also means greater complexity and possibly greater management costs. If the E4900 is partitioned into 2 domains per server the redundancy increases and the availability increases as well.

Figure 20: 12-processor E4900

- V480 and 12-processor E4900
- 48 US-III equivalent processors



High availability technologies found on the E4900 and larger servers including hot-swap for on line service, memory and processor off-lining and proactive diagnosis combined with complete hardware redundancy of the E4900 and larger servers should serve to close any availability “gap” due to fewer

servers deployed in the diagonal architecture.

The results using the E6900 as the larger server are shown in figure 18 and have been previously discussed with regards to vertical 3-tier architecture. Using E6900's as servers in the diagonal tier for larger deployments is financially reasonable. Given the greater utilization rate that is achievable due to the nature of larger servers/domains the acquisition costs are even for the E6900 option and the v480 option. To increase the availability the E6900, like the E4900, can be partitioned using dynamic system domain technology. The 3 E6900 servers should be less complex and costly to manage than the 16 v480 servers.

Summary of Financial Analysis

An analysis of the four possible architectures shows that if the server utilization is the same for the smaller horizontal servers and the larger vertical or diagonal servers then the horizontal option is less expensive to acquire and at times is almost 1/2 the cost. However, if the server utilization of the larger (8 processors or more) vertical and diagonal servers is greater than the server utilization for the smaller horizontal servers (4 processors or less) the acquisition costs will be about the same. In that case the fewer servers in the vertical and diagonal architectures may lead to lower management costs. Regardless, all architectures appear to be viable economically.

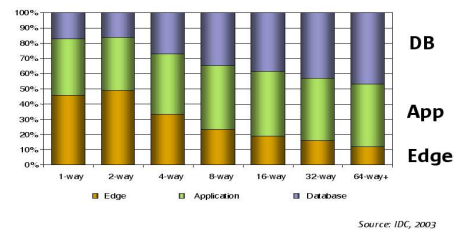
7. Industry Trends for the Application Tier

Data published in in 2003 by IDC (figure 21) shows that the IT industry is deploying

the application tier equally on systems that range from 1 processor up to 64 processors. The percentage of each size of server dedicated to the application tier ranges from 35% for 1-processor servers up to 40% for 64-processor servers indicating that IT departments are deploying a variety of architectures including horizontal, vertical and diagonal. Thus no one architecture dominates current deployments.

Figure 21: Industry Trends--IDC

Server Usage by CPU capacity



8. Summary

This paper has looked at four different architectures for the application tier. The horizontal 3-tier, vertical 3-tier and diagonal 3-tier architectures meet the performance, availability and cost requirements for successful deployments. The acquisition costs appear to be similar for all three architectures. The complexity of the diagonal and vertical architectures seem to be less than the complexity of the horizontal architecture due to fewer servers in the solution. That greater number of servers gives the horizontal architecture an edge in availability;. The vertical 2-tier option is not considered a viable option because of its lack of availability but that may change with the availability of N1 Grid Containers when Solaris 10 is released.

Given the similarity in acquisition costs other factors such as standardization on a single server, data center space, outsourcing contracts, and management expertise in a given server type may be the deciding factors in what architecture is chosen. However, the diagonal architecture seems to have slight advantage over the other architectures given its competitive acquisition costs, low complexity and high availability. Regardless of what application tier architecture IT departments decide to use Sun can provide the solution with its binary-compatible line of 64-bit servers that range from 1 processor up to 72 processors.

References

“*Consolidated SAP Solutions—A Review of Costs*,” Dr. Christian Tolkes, <http://www.sun.com/servers/wp/search/search.cgi/>

“*Attributes and Trends of Horizontal and Vertical Computing Architectures*”. <http://www.sun.com/servers/wp/search/search.cgi/>

Acknowledgments

I would like to thank members of Sun's SAP and PeopleSoft competency centers, SOATC, RABM and the MDO group: Scott Livezey, John Edward, Robin McDonald, Rey Perez, Hugo Rivero, Brad Carlile and Glenn Fawcett.