Capacity Planning in Distributed Environments

Dr. Bernie Domanski
The College of Staten Island/CUNY

ABSTRACT

Capacity Planning has been a discipline practiced by IT since before the invention of the mainframe. It encompasses different measurement, analysis, modeling and reporting techniques and strategies. But with the advent of distributed computing, with hardware resources competing in what may be termed a commodities market, should Capacity Planning be treated with the same reverence as in the past? This paper will tackle this question head on.

Executive Summary

Capacity Planning has been a discipline practiced by IT since before the invention of the mainframe. It encompasses different measurement, analysis, modeling and reporting techniques and strategies. But with the advent of distributed computing, with hardware resources competing in what may be termed a commodities market, should Capacity Planning be treated with the same reverence as in the past?

This paper will tackle this question head on. We will discuss Capacity Planning's history, so that we understand and can sing the hymn “But That's How We've Always Done It.” But, more importantly, we want to point out:

- When it does make sense to do Capacity Planning and how it aligns with the more important concept of Service Delivery;
- That there are alternatives to complex tools that model down to the disk revolution.

The key message we’re trying to make is simple: Capacity Planning in a distributed environment should allow IT to make intelligent, cost-effective decisions regarding the resources required that will rapidly enhance the service given to its customers.

Capacity Planning's Beginnings

In 1965 Gordon Moore, who later co-founded Intel, predicted that the capacity of a computer chip would double every year. Moore had looked at the price/performance ratio of computer chips - the amount of performance available per dollar - over the previous 3 years and simply projected it forward. Moore himself didn’t believe that this rate of improvement would last long. But ten years later, his forecast proved true, and Moore then predicted that chip capacity would double every two years. To this day Moore's predictions have held up, and engineers now call the average rate of capacity increase - a doubling about every 18 months - Moore's Law.
Moore’s Law is likely to hold up for at least another 20 years. Storing all those bits shouldn’t be a big problem either. Consider 1983, the year that IBM released the PC/XT. This was the first PC with an internal 10 Megabyte internal hard disk. Customers who wanted to add the 10 Megabyte drive to existing systems needed to by a $3000 kit - which made the cost per Megabyte $300. Thanks to the exponential growth described by Moore’s Law, the end of 1996 had 5.1 Gbyte hard drives selling for less than $500! That comes to less than $0.10 per Mbyte! And holographic memory is a new technology that can hold terabytes of data in less than a cubic inch of volume - implying that all of the volumes in the Library of Congress could fit in a holographic memory about the size of a fist.

Communications technology, an essential part in distributed systems, stands to benefit from the same exponential improvements that have made today’s $2000 laptop more powerful than a $10 million IBM mainframe computer of twenty years ago.

**Service Delivery**

The key for business today is delivering services to customers as quickly and as economically as possible. Why? To provide customers with the information they need to make business decisions. Customers are not concerned with the underlying hardware, operating system, or application software that brings the information to them. Their only concern is getting timely, *useful* information. For capacity planners, the same philosophy should hold true - the key is to identify *useful* information, and then display it quickly and inexpensively. Useful information is the information that has the *global* perspective, rather than detail. Cost-effective delivery may imply auditing existing IT resources to determine the true cost of delivering business services and information. This must include not only the cost of hardware and software, but the cost of people as well. Buying a new piece of hardware may be cheaper than spending the money on having several people study it, and subsequently buying something that only is a tiny bit less expensive.

The cost of maintaining a Capacity Planning staff should be examined for today IT environments, if for no other reason than to understand the return on investment. In the past, if equipment were purchased that was incorrectly sized for an application, that mistake could cost tens of millions of dollars. Today, a sizing error for a single server, for example, would only cost several thousand dollars. Yes, large systems could require purchasing lots of servers. But if there isn’t enough capacity, very often the solution is to buy more servers or to increase the memory and disk capacities of the ones already purchased. The *difference for making similar mistakes is orders of magnitude different in price*. Understanding this concept and what it costs to maintain a full time Capacity Planning staff might make quite a difference on an organizations’ bottom line. Please don’t misunderstand me here - Capacity Planning...
Planning is most definitely needed where there is a large infrastructure in an organization, or if a truly giant application is being deployed.

However, Capacity Planning must be applied intelligently and in direct support of a cost-effective, Service Delivery function that IT is chartered to provide. Any other reason for doing Capacity Planning really makes no sense at all.

What's the Real Reason to do Capacity Planning?

If mission-critical business applications become overloaded, the poor performance that results could have a very serious consequence: revenue can be lost if dissatisfied customers move to the competition. This consideration, and this consideration alone, is enough for organizations to want to provide sufficient capacity for the mission-critical applications to perform well. It is interesting to note here that when IT is not perceived as providing adequate service levels to different lines of business, outsourcing becomes a very serious alternative. If you can't do it right yourself, pay someone else to do it for you!

Note, too, that the cost of providing too much capacity is just as critical a concern. What company can afford to tie up millions in hardware simply because this equipment might be needed at some time in the future? And does senior management really know how much it costs to obtain the benefits from their IT investments in hardware, software and personnel? IT managers today want to be assured that their sizable investments are not only under control, but are being responsibly managed.

When is Capacity Planning done? Simply, it’s when a business manager realizes that the new application (or new release of an older application) may have a negative impact on customers. If customers see that the performance of the new system is significantly worse than the old, they’ll want to go back to the old system! Why is Capacity Planning done? A successful business succeeds if they are responsive to customer needs. Thus, new functions/features of applications need to be added to keep the customer satisfied. The Capacity Planner, in close harmony with the Application Designer, are responsible for assessing the new functionality being built, and finding the appropriate software and hardware architecture that will work efficiently and as inexpensively as possible. And that also means assessing future application functionality and its impact on application design and system architecture so that the useful life of the application is as long as possible.

What about business vs. technical requirements? The business requirements demand functionality to keep existing customers satisfied and to attract new customers. The business needs this functionality to be available as soon as possible, and as cheaply as possible. But the technical requirements to build that functionality might be very demanding. That implies identifying a software and hardware architecture that might be very costly. It is precisely here where the capacity planner can offer some assistance. On a broad level, each proposed architecture should be assessed for its potential performance limitations (response time) and corresponding service level (amount of actual work done). With a model of the entire system, bottlenecks can be identified, and substitute components inserted and evaluated. At this level, it is not necessary to get very precise predictions of throughput, utilization, and response time. We only need to identify the broad capacity limits and compare them to the anticipated business volumes of work. That way, the business will know, with some degree of confidence, that the architecture being built will have sufficient capacity on day 1, and should be useable until the work volume increases beyond what was anticipated.

Scaleability and Compatibility

Bill Gates, chairman of Microsoft, said that success is a lousy teacher. It seduces smart people into thinking they can't lose. And it's really an unreliable guide to the future. What would seem to be a perfect business plan or the latest technology today, may soon be as out-of-date as an 8-track tape player, vacuum tube television, or the monolithic mainframe computer. Companies contemplating making large investments into IT will try to avoid repeating the mistakes made previously in the computer industry. History, it would seem, is a good teacher, and observing many companies over a long period of time can teach us principles that will help us with strategies for the years ahead. The key to understanding mistakes is the need to initiate rather than to follow trends. Let’s look at some actual history.

In the 1950s and first half of the 1960s, many companies were trying to establish themselves as leaders in the computer industry. What each company did, even within their respective product lines, was that each model had a unique design and required its own operating system and application software. Computers at different price levels had different designs - some were dedicated to scientific applications, others to business applications. It took a great deal of energy and time to get software that ran on one computer to run on another. But this was the trend … just keep building different machines and operating systems.
But the initiative that revolutionized the industry came out of seeing a real business need. Organizations did not want to keep re-inventing the wheel as their capacity needs grew bigger. And certainly they did not want to keep converting software so that they could say that they were at the “leading edge” or that they were at “the state-of-the-art.” The key, found by Tom Watson of IBM, was to develop a scalable architecture. All of the computers in IBM’s System/360 family, no matter what size, would respond to the same set of instructions. All of the models would run the same operating system. Customers would be able to move their applications and their hardware peripherals freely from one model to the next. IBM’s notions of a scalable architecture reshaped the industry.

During a similar period, the minicomputer industry was created by Ken Olsen when he founded Digital Equipment Corporation (Digital). He offered the world the first small computer - a PDP-1. Purchasers now had a choice: they could pay millions for IBM “Big Iron” System/360, or pay about $120,000 for a PDP-1. Not as powerful as the mainframe, it could still be used for a wide variety of applications that didn’t need all that computing power. In 1977, Digital introduced its own scaleable-architecture platform, the VAX, which ranged from desktop systems to mainframe clusters, and again, scaleability did for Digital in minicomputers what it had done for IBM in mainframes.

What’s the lesson here? Companies like IBM and Digital were successful then because they saw a need for business that business had … to fill incremental computing needs in different ways, without having to waste prior investments in IT. This same need is still with us today. If a company needs more computing power, they ought to be able to get more power so long as its mission-critical application software can still run.

Computers were once intentionally designed to be incompatible with those from other companies - the manufacturer’s objective was to make it difficult and expensive for existing customers to switch brands. Amdahl, Hitachi and other mainframe clone companies ended the mainframe monopoly IBM held. In addition, a cottage industry emerged in the storage arena where companies like StorageTek or EMC could supply completely compatible disk drives for the generic mainframe. Market-driven compatibility proved to be an important lesson for the computer industry.

This notion of market-driven compatibility extended into software and operating systems. While UNIX™ was once only the darling operating system of the academic community, it became embraced by many hardware manufacturers including Digital and HP. With its proliferation on many machines, even IBM could not ignore its presence. We see today that MVS, the proprietary operating system of the mainframe, now includes many functions and features to make communication with other UNIX-based systems seamless. IBM dominated the PC market at its beginnings; then many PC-compatible makers emerged to take part of the pie - as long as their clone PCs ran PC-DOS or MS-DOS applications, they had a chance. Even Apple had to give in and build software that would allow DOS and Windows applications to run on their hardware. Market-driven compatibility is partly responsible for the exponential growth and acceptance of the Internet. Killer programs like Mosaic and its successors, Netscape Navigator and Microsoft’s Internet Explorer, allow organizations to share information across different hardware platforms, each running different operating systems.

Perhaps the most critical business IT problem has been solved too - software portability across platforms - by Sun’s JAVA. Efforts like IBM Systems Application Architecture (SAA) and consortia efforts like those from the Open Software Foundation (OSF) tried to define infrastructure common to all. But all of these efforts failed miserably. With JAVA, an object may be defined on a Sun SparcStation, clipped to a Web page on an HP 9000, cached on NT, and fired on a Mac or Network Computer. JAVA makes dynamic distributed systems possible, where we can readily move objects around for optimal placement during development time, deployment time, and even run time. Perhaps compatibility across platforms is really here.

Scaleable architectures and market-driven compatibility are concepts that drive capacity planning for distributed systems. The key here is the network - the glue that connects the seemingly different components of the architecture. Thus, capacity planning becomes less of a function of say, counting available MIPS, and becomes more of a function driven by anticipated new business that has to be processed:

- we want to scale our applications up to process more work;
- we want them to run on the new hardware we acquire;
- we need to connect applications (i.e. data) that currently exist on different platforms; and
- we don’t want to re-invent or convert anything, if we can help it, to keep our costs down and our productivity up.

Bill Gates summarized the status of IT recently - “It’s a little hard to appreciate how far we’ve come from the good old days where just to get the sales report formatted in a nice way, you might wait nine months!”

Capacity Planning in Distributed Environments - B. Domanski - CMG97
…we’ve really gone way beyond anything that ever happened on the mainframe. … you really will be able to do simple, multiser ver applications. Just sit down, write a few lines of business logic, and boom - connect all that up.” Capacity Planning is going through a re-engineering process. And Capacity Planning’s “cousin”, Performance Management, is also going through a similar transition. Sam Greenblatt, Senior Vice-President of Advanced Technology at Computer Associates, recently said “Integrating application, system and network management is helpful only if it yields useful business information. Nobody cares whether or not a system is down if it doesn’t impact their business.” Again, the key - Service Delivery - is that Capacity Planning is a methodology used to ensure that maximum service is delivered to customers.

Simply stated, today’s Capacity Planning efforts need a broader focus with less precision. We’ll explore this a bit further.

**Is Capacity Planning a Checkoff Item?**

While many organizations would consider their mainframe capacity planning efforts as mature, these same organizations are at the earlier stages as far as planning for distributed environments are concerned. Perhaps because budgets were departmentalized, and because of the dropping cost of hardware, individual department managers had gone to a vendor, and made the “buy” decision for client/server gear. Here, the traditional capacity planner was not involved in the process. The next few years may see the planning function return to central IS management in many organizations.

In recent times, we must ask why the planning function was not housed in central IS? Primarily cost; the hardware has, for the most part, become a commodity, like bread or eggs in the grocery store! A 16 Megabyte upgrade to the mainframe required a 2-month justification study; today, we can buy 16 Megabytes of memory for a PC at our local computer store for about $100! So rather than burden the planner with commodity shopping, users took on that responsibility.

This notion of hardware being a commodity has thrust capacity planning under the microscope - do we even need CP any more? With prices dropping and technology advancing, it might be easier to just buy new gear when you need it, period, and not do any Capacity Planning at all! Consider, too, the cost of doing a Capacity Planning study - people with specialized analytic skills are needed; the study will take some serious time before it’s completed; and dedicated machine time may be needed to collect measurements from specific performance experiments. In today’s commodity market, Capacity Planning may not always be a viable option.

Again, we need a broader focus with less precision. It no longer is critical to obtain the exact utilization of each server in the network. What becomes more important are questions like “is the network providing adequate performance? What should we get/do if it isn’t?” We need to understand distributed applications and, more importantly, how to grow distributed applications; that is, how to build distributed applications that are scaleable. How many more users can be added, while preserving acceptable performance, i.e. response time. These questions force the capacity planner to seek a new perspective - one that is more closely tied with application-specific measurement.

Capacity Planning is not done for the same old reasons anymore. Capacity Planning is done for business reasons. The key to success for a business is ample delivery of necessary IT services, of which scaleability and compatibility are critical components. So the traditional capacity planner must now become more application savvy … and must understand the network. (S)he must focus attention on identifying the parts of an application that won’t scale up well … and then offer solutions. (S)he must always keep compatibility across platforms at the forefront of their thinking. And (s)he must be able to anticipate bottlenecks - either in a server, the network, or a client - and propose alternative components that will avoid broad performance problems. Deploying new applications on a specific architecture may be wonderful today, but may become disastrous tomorrow if that architecture becomes a dinosaur and new/faster/cheaper gear is available.

**Adding Small Pieces of Capacity**

Organizations sometimes find themselves in a bind - additional work has to be processed because of a higher volume of business, but their in-house IT resources are at or near their processing capacity. At times like this, organizations can do one of two things: they can scrutinize the system carefully to identify and tune resources that could be used better, or, they can go out and buy more resources. Tuning, while important to delivering good service to customers, very often yields very small increments in capacity - for example, perhaps a 1% or 2% buy-back in CPU utilization on a single processor. While there are numerous stories in the capacity planning folklore of changing one or two system parameters that yield incredible improvements, the prudent capacity planning should not count on finding those magic tuning parameters - especially if the system and its critical applications are constantly being
reviewed. Buying more resources obviously can yield tremendous capacity increases - but for more money!

If you consider history once again, the mainframe was, for the most part, a single system to manage. And it was a stable environment for production workloads. Mainframe vendors provided a rather thin set of tools for systems management, and this resulted in an entire cottage industry to be born: companies that made systems management and capacity management tools. Tools that emerged from third party companies included software monitors (e.g. Candle's Omegamon and Landmark's Monitor), performance databases/archives (e.g. Merrill's MXG and Legent/Computer Associates MICS), and queuing-based modeling packages (e.g. BGS' BEST/1 and Metron's Athene). In addition, many of these mainframe tools rely heavily on raw measurement data being processed by SAS (SAS Institute). Purchasing licenses for these tools would give the Capacity Planner an arsenal of functionality that would be used to predict future IT resource needs at specific service levels. But these tools are/were not cheap! Typically, a software license for any one of these tools could range between $20,000 and $120,000 (MXG being the exception).

Let's point out a couple of characteristics of this set of tools. First, the predictive tools never had perfect accuracy. Response time predictions were considered acceptable if they were within 15-25% of reality. Second, no single tool stood out as being able to predict workload growth based on past history or anticipated business volumes. And third, and most important, these tools were not designed to understand distributed environments. Software monitors could not capture and report on end-to-end response time. Modeling products often did not take the network into account. The message here might be to stay away from fancy tools that have a single system focus if you’re planning for the capacity of a true distributed environment.

So what does the distributed system’s capacity planner do? Measurement is the key to proper use of these tools in an economic way. The measurement architecture should be able to populate a performance database with distributed workload data. The capacity planner must look for tools that have a more global perspective. For example, we would want a software monitor to be able to display the status of every IT resource in the distributed environment. And, if some resource is performing poorly, then we want to be able to isolate and drill-down on that device to get more detailed information. In the performance database area, we need to have archives that contain summarized data across systems. We need cross-system data to look at trends. HP’s OpenView PerfView/MeasureWare Agent (MWA) and GlancePlus are examples of distributed systems tools that monitor, archive and allow drill-down of specific IT resources connected on a network. And in the modeling area, we likely have to use simulation-based modeling tools such as NetArchitect, QASE, or SES/Workbench to understand the network along with the client and the server and each one’s impact on end-to-end response time. Such a model would have to have a greater understanding of the application as well, to identify poorly designed software components. The objective is to be able to provide planning information about the entire distributed environment, and how applications run in that environment. By having a measurement architecture that doesn’t populate tools with distributed workload data, then even with the best predictive tools, we still have to guess on how to populate the model with behavioral data.

Thus, Capacity Planning tools (monitors, collections of performance data, and models) can and should be used to find when to add the right pieces of capacity - call it Just-In-Time-Capacity. If resources are added at the right time, there will be no interruption in the quality of service delivered, and there will be no waste with respect to paying for resources before they are really needed.

Multi-System Focus

Capacity Planning efforts today must focus on the multi-system aspect of distributed systems. Figure 2 above is meant to illustrate how mainframe based systems evolved to client/server architectures. The diagonal line depicts the network; functions below the network line are functions that are performed on a client; functions above the line are performed on a server. The evolution shows how distributed systems have emerged by placing more emphasis on the client...
as it became more intelligent. The term Remote Management refers to managing the data and processing resources that exist at a remote server. The term "distributed" is used to indicate that a key function, e.g. "presentation" or "processing" or "data management", is performed on several machines. Thus Distributed Presentation refers to systems where the client has enough intelligence to offload some presentation functions; Distributed Processing implies that the client can offload presentation functionality and some of the processing chores; and Distributed Management indicates that the actual management of data occurs on both clients and servers.

Today, we have the intelligent client that can perform some significant functions on its' own; we have the network, whose bandwidth can define the system-wide performance; and we have the server (or servers) that often contain the data being sought for analysis and presentation on the client. This is a multi-tiered architecture, and is fundamentally different from old mainframe systems in that (1) there were no intelligent clients, only dumb terminals, (2) the network only connected machines of the same type, all using the same communication protocol, and (3) the one single server contained all of the data necessary on it and it alone.

Today’s systems have a great challenge to conquer: they often are asked to pass information along in a 3-tiered computing architecture: data often resides on a large mainframe; midsize machines often house smaller, but key, databases; and even smaller “client/server” machines often characterize a LAN. Capacity planners must face this reality of multiple systems connected via one or more networks, where the systems are of different sizes, running different operating systems, and containing different database management systems. Again the key to capacity planning must be to deliver useful information in a cost-effective way. Thus, the capacity planner must examine how applications will be using these different tiers. Common questions would include (a) getting many small servers vs. few large servers, and (b) where should specific processing functionality reside: on the client or on a server?

And let’s not lose sight of networks! The sensitivity of the backbone network of multi-system configurations could easily be the single resource that dominates poor service by applications. The network bandwidth and speed is sensitive to choice of protocols. If an organization is considering using a Wide-Area Network, this often will require some study - perhaps involving a network simulation! The available/useful bandwidth of the public network, too, is extremely sensitive - and is less controllable! Router capacities must be carefully considered as well. A common performance/validation technique is to place a sniffer at some point on the network to collect network traffic statistics from which response times can be calculated.

**Costing**

Presumably, a great advantage of distributed systems is being able to buy needed capacity in small increments. Small systems have exhibited this characteristic; this allows a more accurate sizing of necessary equipment to business needs. All of this translates into less excess capacity and therefore lower overall cost - or so one might think.

As we’ve said before, scaleability is key for capacity planning for distributed systems. Economically, scaleability is a primary contributor to reduced cost. The theory is you buy enough capacity to do your processing now, and if additional capacity is required in the future, it is acquired at a reduced unit-cost because of the constant improvement in price/performance ratios.

Is there a fallacy in this thinking? Consider what happens during the entire life-cycle of equipment - especially client/server equipment. You buy what you need today - and incur both acquisition and installation costs. Over time, you also incur an operational costs (licensing fees, support personnel, and maintenance). But is that all of the costs? What happens when additional capacity is needed, i.e. one server needs more capacity. Yes, you acquire a bigger server, but what do you do with that old server? Do you throw it out? Most companies would rollover the server to a new place - that is, the old server is likely to replace an even smaller machine somewhere else in the organization. And as this may cause a cascading effect, consider, too, that there are costs that have to be incurred when installing each old machine in a new place, e.g. installing new software, testing, support personnel costs, etc.

In the mainframe environment, these rollover costs were seldom encountered. Processors were generally exchanged or added. But in the distributed environment, a processor swap can cause multiple rollovers. If the rollover costs become significant, it may become uneconomic at some point to do the rollover! Leilani Allen at a meeting of the Financial Management for Data Processing Association said "By the year 2000, it will cost more to keep old technology than to upgrade."1

---

While this may be surprising, the situation does point out that we should understand the actual magnitude of rollover cost in building a financial model over the life span of a distributed system. James Cook\(^2\) proposes such a financial model; namely that the life-cycle cost equals the initial acquisition cost, plus the operational cost, plus \(N\) times the installation cost, where \(N\) is the total number of swaps in the rollover series. The net impact of a first processor swap may cause an increase of nearly 50\% to the original acquisition cost, 100\% for the second swap, and 150\% for the third swap! Note, too, that at some point, rollover costs will consume any savings that may be gained on cheaper MIPS being available in the future. Thus, spending a little more initially on capacity may actually avoid a processor swap (and its costs) in the future.

The bottom line: the focus of financial management strategies for IT has long been on acquisition. But the realities of the life-cycle of equipment in distributed systems dictate that ongoing operational costs (that address rollover) demand more attention. Old, conventional wisdom just doesn't apply to distributed systems.

### Service Levels

The capacity planner should attempt to base a capacity plan on projected workloads that each receive an acceptable degree of service - service levels. Workloads should, ideally, be specified in business terms - sometimes referred to as natural forecast units or business transactions. One advantage of basing the capacity plan on business transactions is that a clear correlation exists between the capacity plan and the business plan, and should the business plan change, the effect on the capacity plan will be immediately obvious. Remember, the focus of the capacity planner should always be on business needs.

But before service levels can be managed, they have to be specified and agreed on. Application designers, capacity planners, and performance analysts need to agree on the answers to questions like:

- What is the relative importance of each application to the business?
- What is the required availability of each application?
- What is the required performance, i.e. response time, of each application?
- What are the current workload volumes for each application?
- Are the workload volumes or relative importance expected to change over time? If so, how?
- Are new applications planned, and what are their relative importance and workload volumes?
- What financial constraints exist on the acquisition of computer resources?
- Can IT costs be charged back to the business units that use the applications?

Service level measures, specifically availability, response times and workload volumes, should be reported by business unit and application. When compared to service level requirements, we can see whether goals are being met, and based on priority, determining that impact on the business. Re-planning capacity requirements may be necessary if the stated service level requirements are adversely affecting the business.

But there are several obstacles to hurdle before managing service levels across distributed environments. The nature of client/server applications is such that the application architecture distributes some portion of its processing to the client. This introduces additional points of potential failure or bottleneck, and also complicates the definition of a transaction. In the network, heterogeneous systems coexist using different communication protocols and application-to-application session protocols (e.g. SNA APPC, OSF's DCE, etc.). Thus, managing end-to-end service levels becomes very complex because of the different computing and communications device in a path. And there is a relative lack of maturity, as compared with the mainframe, of measurement, monitoring, and applications management tools - but the technology problems in building similar tools are different for distributed systems. There is hope - base measurement data, like utilization and traffic, is available from most intelligent networked devices. Standards have been proposed for system performance data across platforms. Agents have been and still are being developed for database instrumentation.

The problem remains one of intelligently correlating the various component measurements into an end-to-end picture of the service provided to application users. To obtain performance at an individual transaction level will require code instrumentation within the application. Application-specific measurements would be collected and fed to an enterprise manager, which would allow the end-to-end correlation of performance.

HP's MeasureWare Agent Transaction Tracker is an example of a tool that can be used to manage IT service levels. User-defined applications enable you to group processes together by user or application name for better tracking of an individual's or application's

---

impact on your resources. MeasureWare Agent is consistent across multiple platforms - HP-UX, Sun SPARC, IBM AIX, Bull, and NCR WorldMark - using a common interface and standard metrics.

To create application specific measurements, the joint efforts of HP and Tivoli have produced ARM - Application Response Measurement. ARM is an API / software developer's kit that allows application developers to add appropriate performance instrumentation to their code easily. Applications deployed using C/C++, Microsoft Visual Basic, MicroFocus COBOL, and Delphi are currently supported under all of the popular PC platforms.

Summarizing, we note that as distributed applications are deployed, and as management responsibilities for these distributed applications shift to the central IT organization, service level management becomes increasingly important, complex and difficult. Many of the management tools required to effectively manage service levels across the enterprise are now first coming into their own. The first step toward implementing enterprise service level management is to approach systems management from the end-to-end application workload perspective, rather than viewing the environment as a collection of physical components.

Summary & Thoughts

Capacity Planners have some very critical questions to consider including:

- What type of servers should be deployed to support specific applications?
- How many servers are needed? How much bandwidth does the network need to provide acceptable service levels?
- How big should the servers be to handle the application volume?
- Can we define service levels for different distributed applications?
- How can we tell if the servers are optimized for the network operating system and communications software that is in place?
- Are applications properly configured for a mixed-platform environment?
- Are appropriate measurement sources available?
- What are the true costs of incremental additions to capacity?
- Is effective service level management possible for distributed systems?

Capacity management for distributed systems faces many challenges. The one single driving challenge is to constantly ask whether Capacity Planning is helping IT deliver the best service possible to its customers, now and in the future. IT managers must always address building scaleable architectures that have market-driven compatibility. Evaluating a “checkoff” item capacity planning philosophy for your environment may prove to be a real cost saver - especially when taking into account the cost of maintaining a full-time capacity planning staff. And we shouldn’t forget to include the true costs associated with adding incremental capacity.

Instrumentation across platforms is key - especially for new applications. Applications should be designed from the outset with the goal of being able to provide application-specific performance measurements so that we understand the quality of service delivered. Without application-level instrumentation, service-level management across the enterprise may not be possible.

Data reduction/summarization/reporting software will be required to manage the volume of customer data across platforms, and must address the network, as well as the client and the server. Modeling tools must become graphical to allow network topologies to be easily defined, and must then address modeling heterogeneous combinations of hardware and system software platforms. Modeling should be able to provide predictions of IT usage and service from a global perspective as well as a detailed focused perspective.

It is our hope and desire to see these challenges addressed in the near future with the development and application of new techniques.

Acknowledgements

We would like to thank those CMG referees and Editorial Review Board members for taking the time to read and review this paper. The comments and suggestions provided were invaluable in improving the quality of the work.