COLORADO TECHNICAL UNIVERSITY

THE SIMALYTIC MODELING TECHNIQUE AS APPLIED TO CAPACITY PLANNING IN A MULTI-PLATFORM ENTERPRISE

A DISSERTATION SUBMITTED TO

THE GRADUATE COUNCIL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS OF

DOCTOR OF COMPUTER SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

BY

TIM R. NORTON

M.M.S., University of Texas at Dallas, 1977

B.A., The Colorado College, 1972

COLORADO SPRINGS, COLORADO

DECEMBER 1997

THE SIMALYTIC MODELING TECHNIQUE AS APPLIED TO CAPACITY PLANNING IN A MULTI-PLATFORM ENTERPRISE

BY

TIM R. NORTON

THE DISSERTATION IS APPROVED

Dedication

I would like to dedicate this work to my extended family and friends, for their support and encouragement, and to my wife, Beth, and children, Mike and Becky, for the years of support, encouragement and patience that made it possible.

Acknowledgments

I wish to gratefully acknowledge the guidance and insight provided by my committee chairman, Dr. John Zingg, the effort and guidance from Dr. Carol Beckman and the efforts of Dr. Charles Reddan who replaced another member at the last minute. In addition, I wish to gratefully acknowledge the help and support from many industry leaders, notably Dr. Jeff Buzen, Dr. Robert Goettge, Dr. Dan Menascé, Dr. Doug Neuse, Dr. Herb Schwetman, and Dr. Connie Smith, who provided valuable insight.

Abstract

Application designs are changing from single system to cross platform client/server designs utilizing the features of different types of computers, operating systems and networks. Planning the capacity of large computer installations using multiple systems requires an understanding of each of these areas and the inter-relationships between them.

The “Simalytic” (**Sim**ulation/An**alytic**) Modeling Technique[1] addresses modeling complex multiple-platform computer applications at an enterprise level for capacity planning. This technique uses a general purpose simulation tool as an underlying framework and an analytic tool to represent individual nodes or servers when predicting capacity requirements for an application across an enterprise. This technique combines both platform-centric tools (limited features but detailed platform information) and general purpose tools (rich low level features) to address today’s large heterogeneous enterprises. This methodology takes advantage of features in the different techniques (simulation vs. analytic queuing theory) as well as features in the different tools (platform-centric vs. general purpose).

“In theory, there is no difference between theory and practice. However, in practice, there is!”

Jerry Percell, CMG96

[1]*Simalytic*^{TM}, *Simalytic Modeling*^{TM}, *Simalytic Modeling Technique*^{TM} and *Simalytic Enterprise Modeling*^{TM} are trademarked by Tim R. Norton
All other trademarked names and terms are the property of their respective owners.

Contents

LIST OF ILLUSTRATIONS

LIST OF TABLES

LIST OF EQUATIONS

1. Introduction and General Background

Data processing has been one of the fastest, if not the fastest, evolving industries of the century. Applications that once would have been implemented as batch systems on a single computer are now multi-platform on-line transaction processing client/server systems with GUI (graphical user interface) front-ends on PWS's (programmable work-stations) attached to departmental servers and mainframe repositories. These new applications utilize the features and services of different types of computers (mainframe, mid-range, desktop) running different operating systems (MVS, Unix, OS/2, Windows, etc.) connected by a variety of communication network techniques (RPC, DCE, NFS, FTP, SNA, APPN, etc.) (Hatheson 1995; Wilson 1994).

As applications move into this new client/server world, the question is how to select the right systems at each level and, once selected, how to insure that those systems are the right size? If any one of them is too small the whole application will fail. If any are too big, the cost of running the application may exceed the revenue it generates. Neither is a very attractive situation.

The objective of capacity planning is to find the successful middle ground. Not too many years ago capacity planning meant watching a few metrics like processor utilization and the overnight batch window to determine when an upgrade would be needed. Today, planning the capacity of large computer installations with multiple systems requires an understanding of not only the operating systems, the platforms, the clients, the servers, the networks, the transaction systems, etc., but more importantly, the applications and the relationships between them. Once those relationships are defined and understood, the performance of the applications can be assessed against the business objectives and goals.

Projected business volumes are then modeled to predict the capacity required to meet those goals at future volumes.

There are many modeling tools and techniques that address both performance and capacity for each of the systems in today's multi-platform environment (Pooley 1995; Smith 1995). However, these tools either address a specific environment or provide for a generalized approach. They do not support performance analysis and capacity planning when applications span across multiple heterogeneous platforms. The Simalytic Modeling technique developed in this dissertation provides a bridge across these existing tools to allow the construction of an enterprise level application model that takes advantage of models and tools already in place for planning the capacity of each system.

1.1 Dissertation Topics

The major topics covered in this dissertation are:

- Section 1, *Introduction and General Background,* is an introduction into the overall area addressed and general background, which includes an introduction to capacity planning for transaction applications and a discussion of the importance of using transaction response times when modeling applications.
- Section 2, *Modeling Capacity Projections,* is an overview of modeling techniques for capacity planning, a discussion of the differences between platform-centric and general purpose modeling tools and a description of the problem area, including the problem statement and the hypothesis to be proven.
- Section 3, *Simalytic Modeling Methodology,* describes the methodology to be used, which includes a detailed description of the mathematical foundation behind the Si-

malytic Modeling technique, and a discussion of its application to the issues from Sections 1 and 2.

- Section 4, *Simalytic Model Development,* describes the detailed development of the Simalytic Modeling technique, including the detailed discussion of the construction of an actual model for a two server system.
- Section 5, *Investigations into Simalytic Modeling,* describes the results from example models of a three server system, a four server system and an eight server system constructed using the process discussed in section 4.
- Section 6, *Conclusion,* discusses the implications of the results from section 5, the application of the technique and areas of further research.

1.2 Comparison of Simulation and Queuing Theory

Both simulation and queuing theory have proven to be valid and useful modeling techniques for capacity planning (Domanski 1983). The choice as to which to use is often not easy and may be influenced more by the skill set of the modeler or the availability of the tool than by how well the tool addresses the current situation. When compared as to their suitability to the capacity planning task, each technique has strengths and weaknesses that make it more or less suitable given the particular circumstances of the individual situation. Although there are many points of comparison between simulation and queuing theory modeling techniques, the ones of interest here are those that are disadvantages in one technique but can be compensated for using the other technique as part of a hybrid Simalytic Model.

One of the main distinctions is how each technique addresses complex variation in the distribution of the interarrival times and service times. For example, the distribution of interarrival times might be exponential with a mean of 20 for the first two thirds of the modeling interval, but have a mean of 5 for the last third due to the nature of the system being modeled. Capacity planning simulation models preserve the complex variation generally by using system trace data to provide actual distribution timings. Queuing theory, however, assumes a single mathematical distribution, such as exponential or uniform, with a single average value for the entire model run, which might be 15 in the example above. Thus the queuing theory model would predict a response time that might be higher than the system average and still miss the unacceptable response time in the last third of the interval. This difference is a distinct advantage for simulation models if the distributions of the actual environment being modeled do not match a mathematical distribution or if the environment changes over the modeling interval such that any of the distributions are not accurately represented by a single average. Because distribution variation complexity tends to increase as the length of the modeling interval increases, simulation models also have the advantage of being able to model the environment for much longer periods of time. When the complexity in the actual environment is small and the system is homogeneous across the modeling interval, queuing theory models have been shown to be accurate predictors (Buzen 1984; Buzen and Potier 1977).

Simulation models also have the advantage of preserving the identity of individual transactions across the entire model. Attributes can be assigned to the individual transactions to influence model behavior and increase the level of detailed results collected for analysis. Complex transaction routing techniques, such as transaction attribute based or fork/join, can be implemented in simulation models. Detailed results can be collected for both end-to-end response times and the variation in response times by transaction attribute. Queuing theory models cannot implement fork/join routing and they calculate end-to-end response times based on calculations using average response times and routing probabilities. Simulation models provide better results for environments with high transaction variation and queuing theory models provide acceptable results for environments where all transactions are very similar and are represented accurately by the average transaction.

The author's experience has shown the main advantage to queuing theory models is execution speed because simulation models must make many calculations for every transaction but queuing theory models make one set of calculations regardless of the number of transactions represented by the modeling interval. In addition, simulation models must be run many times to establish an acceptable confidence interval which greatly increases the time and resources required. Only a single execution is required of a queuing theory model because the average values of the results are accepted as an implication of the initial assumptions that are the basis of queuing theory models.

Because the focus in this dissertation is on the use of commercial modeling tools, it is appropriate to also note the differences in model construction effort. The author has found that queuing theory models are generally implemented using a type of tool referred to in this dissertation as platform-centric modeling tools, which are discussed in greater detail in section *2.3.1 Platform-Centric* on page 40. These tools include more specific information about the environment being modeled and allow the modeler to work at a higher level of abstraction than when building a model with most simulation tools. Therefore, within this context, queuing theory models are generally easier and quicker to build than a simulation model of the same environment.

1.3 Advantages of Simalytic Modeling

The advantages of using the Simalytic Model Technique are:

- Rapid Analysis
- Spiral Methodology
- Reuse
- Distributed Model Development
- Applicable Tools

Rapid Analysis: A Simalytic application model can be quickly constructed with minimal effort for each node model. Analysis of the application using this high level model allows additional effort to be applied only to the nodes in the critical-path areas, saving time and effort by avoiding areas with minimal impact on application responsiveness.

Spiral Methodology: Simalytic Modeling provides a mechanism to promote the exchange of information between the users, the developers and the modelers. As more information becomes available, the application model is refined. Early assumptions can be replaced with the results from more sophisticated tools as more details become available. This spiral approach allows modeling earlier in the design phase as well as after implementation.

Reuse: Simalytic Modeling provides for the direct incorporation of existing tools and techniques into the high level model. The investment in time, effort and money expended for training on the tools used for existing system models is preserved. It is much easier to get someone who has built a number of models of a system, using a tool they are well trained on, to do another model, than it is to start over with a different tool.

Distributed Model Development: Simalytic Modeling provides easy distribution of modeling activities to multiple people or organizations. Because of the reuse of existing modeling techniques, node models can easily be "sub-contracted" to the modelers most familiar with those tools and systems.

Applicable Tools: The ability to use the most applicable tool for each node of the Simalytic Model increases the speed of model construction through reuse and improves the accuracy of the application model through the platform information inherent in the submodels without reducing the overall model responsiveness to arrival distribution variation. For example, the network component of the application can be assumed constant or modeled with greater detail if the modeler determines that the network is a major component of response time. If the simulation tool does not provide the level of complexity needed, then a specialized network modeling tool can be used to create network nodes for the whole Simalytic Model or just for a critical part of the network between two servers. In either case, the simulation framework of the Simalytic Model provides a mechanism to address any unusual arrival distributions if that is important to the overall model.

1.3.1 Merging of Methods

The Simalytic Modeling Methodology merges the advantages of the two traditional modeling techniques, simulation and queuing theory, such that the advantages of one are used to offset the disadvantages of the other. Section *1.2 Comparison of Simulation and Queuing Theory* on page 3 identified the major areas of comparison as: distribution variation, transaction identity, execution speed and ease of construction. Implementing the model framework as a simulation model provides the modeler access to transaction routing and identity facilities. Implementing the nodes with queuing theory models reduces the construction effort and execution time. In addition, the quality of the results should be improved because the vendors of the platform-centric queuing theory modeling tools generally include in the modeling tool large amounts of specific information about the platform being modeled. This type of information is generally not included in a simulation model. Either it is not available to the modeler or the cost of including it in the model is too high because of the level of research required to find the information or the effort required to implement it in the model.

It is the author's opinion that the relatively small amount of research into the combination of simulation and queuing theory modeling techniques is due to the high degree of knowledge required in both areas to construct such a hybrid model. The assumption appears to be that such a hybrid model must be constructed by implementing the algorithms for both techniques in the same computer program. Although not stated, this assumption is implied by the use of programming language segments as examples, as seen in papers such as (Schwetman 1978). This is further supported by the fact that Dr. Schwetman is no longer active in the area of hybrid modeling and sells a commercial simulation tool, CSIM from Mesquite Software, Inc. (Schwetman 1996). This assumption is the principle motivation for using a combination of commercially available tools for the development of the Simalytic Modeling Methodology. By combining the features, functions and advantages of commercial tools, the modeler is no longer required to implement complex simulation or queuing theory algorithms but can still benefit from results of such implementations.

1.3.2 Suitable Application Types

Although the Simalytic Modeling technique addresses the complexities of client/server environments, it is not suitable for all situations. The Simalytic Modeling technique uses the strengths of the simulation framework to model the relationships among the parts of an application that span several systems. The applications most suitable for modeling with the Simalytic Modeling technique are those composed of relatively small units of work (transactions) somewhat evenly distributed across several heterogeneous computer systems.

Those applications that are implemented predominately on a single system would gain little from the additional effort to create the simulation framework for a Simalytic Model. These applications, such as batch and interactive workloads, are generally composed of a single process, or of a series of processes, that remain on the same system once they start execution. Even if they do switch between systems, the responsiveness of such workloads is generally dominated more by the performance of the predominate system than by the interrelationships. An application that has little or no interaction among different systems is generally adequately predicted by current single system tools. Such applications are well contained within a domain addressed by a platform-centric tool and are best modeled by that tool.

The most suitable applications exhibit a relatively high degree of interdependence of relatively small units of work among different systems. These applications are generally transaction based workloads implemented with multi-tier client/server designs. It is also important that the workload can be correlated directly with measurable business functions. Applications with either high workload variability or transactions that provide service for multiple workloads are not suitable. Workload characterization problems such as these will impact any model using any technique. Currently, the characterization of the level of system interdependence is very subjective. An apparently rich area of future research is the development of techniques to quantify the variability of multi-tier client/server applications and to determine the suitability for Simalytic Modeling based on that quantification.

Although Simalytic Modeling is generally not suited to single system applications, the possible exception would be to use the technique to model very long intervals of single system transaction applications. The technique used in the Simalytic Function, such as a rolling average, could adapt to changes in the arrival rate rather than use a single average for the entire interval, which is generally the case with queuing theory tools. This application of the Simalytic Modeling technique should also provide an interesting area for future research.

1.4 Capacity Planning

What is capacity planning in the context of today's computer systems? Capacity is the maximum amount that something contains, but amount of what? Planning is making decisions based on the forecasting of future events using current and historical information. But the decisions to be made are sometimes contradictory and the information incomplete.

The capacity of a system can be measured many different ways, depending on the business the system supports. Generally, the way a system is measured centers around the performance of one or more of the applications. The system "has enough capacity" if everything is getting done when it is needed. This may sound like a simplistic statement, but the key to understanding the capacity of a system is the definition of the performance objectives. Without some goal at the business level there cannot be any meaningful statements about the capacity of a computer system as long as it continues to run. Without goals, a system is “out of capacity” only when it becomes so overloaded that it deadlocks or when some devices are so over-utilized that data is lost. Performance might be very poor, but we only know it is unacceptable if it is worse than the goal (Domanski 1995 13; Rosenberg and Friedman 1984; Wicks 1989; Wilson 1994).

Therefore, capacity planning is making decisions about the resource requirements of a given computer system based on the forecasting of future application performance using the goals and expectations of the business. That is a very broad and general definition, but it captures the objectives behind capacity planning. Capacity planning can be viewed as all of the activities required to answer the question, “What do we have to buy and when do we have to buy it to make sure that the applications that run the business perform at the level required to insure the business succeeds?”

1.5 Transaction-Based Applications

Although there are still many important batch applications, this discussion centers around transaction-based applications for several reasons. First, multi-platform client/server systems are generally focused toward small real-time units of work such as transactions rather than large long-running units of work associated with batch. Second, large batch applications generally have long execution times and points-in-time when all work must be completed and moment-by-moment responsiveness is seldom an issue. It just doesn’t matter when a third of the paychecks are printed; they *all* must be ready when the time comes to distribute them. Third, batch workloads have a much lower arrival rate, often once a day or less, and tend to be serialized. Fourth, transaction-based applications are much more sensitive to the demands of momentary peak loads where batch based applications are more sensitive to scheduling issues and interference from higher priority workloads. Therefore, the traditional single system view will continue to provide adequate performance and capacity planning for batch workloads while transaction workloads require an enterprise view to understand the relationships between responsiveness as a whole and the individual platform resource requirements.

What is a transaction? Transaction processing systems, often referred to as OLTP (On-Line Transaction Processing), allow the end-user to enter a relatively small, independent unit of work into the system and receive some information as a response in near real-time. Transactions, which can be defined from different points of view, include entering an order at a terminal (business transaction), an SQL command (database transaction), or some keystrokes followed by a carriage-return (interactive transaction). In one sense, each keystroke a user types in a text editor is a transaction because a small unit of work (the keystroke) is sent to the sever, acted upon, and information is returned to the user (the keystroke is echoed). A transaction might be defined as messages received from, and replies sent to, 3270 terminals (which were really early PWS's) by an OLTP system such as CICS or IMS or as logical units of work marked by "commit" commands by database systems such as Oracle and Sybase (BGS 1996).

The concept of a transaction is important because it is meaningful to the end-user. Transactions can be counted to establish load (e.g. arrival rate) and measured to establish performance (e.g. response time). The responsiveness of the transactions associated with an application determine if that application meets the needs of the business. A customer service representative can answer more inquiries more effectively when the requested information is presented in one or two seconds than if it is presented in ten or twenty minutes. The business question is to determine where between these two the responsiveness becomes unacceptable. The business objectives determine the required responsiveness for any given application. Modeling techniques can then be used to predict application responsiveness at higher transaction rates to insure those objectives will be met in the future.

1.6 Hypothetical Company Example

The importance of looking at transaction processing is evident when we look at a hypothetical company as its data processing systems evolve over many years. The XYZ Company starts with a centralized batch environment where all business functions use punch cards to enter data into the system and printed reports run the business. Special purpose servers are introduced to improve the productivity of some of the departments and, over time, XYZ has a decentralized environment. Different business functions on different servers cause communications and training problems, so the applications are merged together at the end-user's PWS using various techniques that preserve and hide the legacy applications while providing a single common interface for everyone.

The next three sections (and associated figures) describe the evolution of the XYZ Company from the perspective of the different computer platforms used in the company. Which operating system was chosen for each of the business functions was completely arbitrary and does not represent a recommendation. These figures represent one of many scenarios that could result in a situation where a company must plan for growth in a multi-platform environment. The intent of the figures is to show how the business functions can be initially strongly associated with a single platform and, over time, become dependent on the entire enterprise environment.

1.6.1 Centralized Environment

A general overview of the centralized batch environment is shown in *Figure 1 The "Old" System*. Capacity planning in this environment is focused on getting the reports delivered in the morning. Much of the real work of the company takes place “off-line” and the performance of the different applications is not a major issue. The business will continue to function for hours, or even days, using the last reports printed.

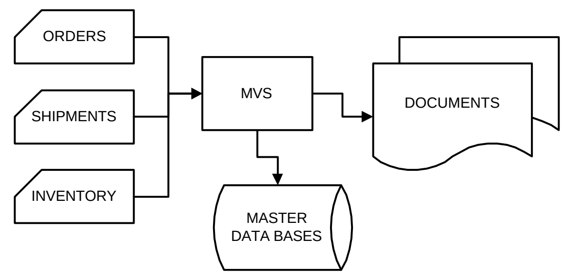

Figure 1 The "Old" System

A capacity planning model of this system consists of a function that maps input volume to run times. If it takes three hours to process 1000 orders, then we can project it will take 4.5 hours to process 1500 orders. This model is easy to build and validate. Historical data provides many points on the function curve and the curve can then be projected to higher input volume. This technique will work with processor time, device utilization or almost any other measurable resource.

1.6.2 Decentralized Environment

At some point one of the department managers realizes that he could eliminate data entry costs and get more up-to-date information by using an on-line system. As XYZ Company does not have any policy or direction about distributed systems, the manager installs a turn-key system where both hardware and software are provided by the vendor. Another department manager sees the benefits and installs an on-line system for that department. Of course, the second department needs different software which comes from a different vendor who uses different hardware. After a while, XYZ Company has a number of systems and looks something like *Figure 2 The "New" System*.

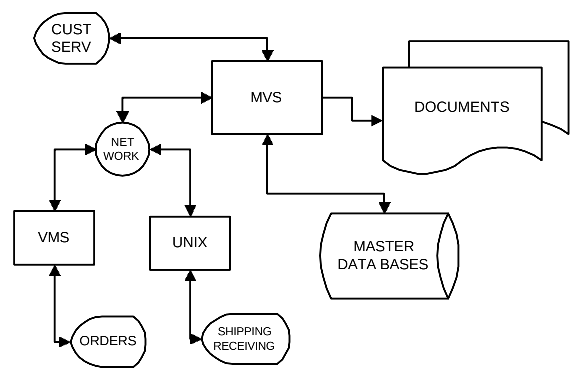

Figure 2 The "New" System

Capacity planning in a decentralized environment is still fairly straight forward. Even though each system sends the old card data to the MVS system for nightly processing, each system is an island to itself. If the Order Entry workload outgrows the VMS system, the Order Entry users will see a performance delay, but the Shipping and Receiving users will not be impacted. As long as the nightly batch runs get done and the reports printed (which now also means downloaded to the departmental systems), the business continues to function.

The introduction of transaction processing complicates modeling performance in this environment, but the transaction and batch workloads remain generally isolated by shift. Modeling the daytime transaction processing workload requires more sophisticated queuing theory or simulation tools, but the overnight batch is forecasted much the same as before because it is generally a sequential process. If the performance of a database deteriorates with volume, then the function that maps input volume to run times from the example in *1.6.1 Centralized Environment* on page 14 might change from 4.5 hours to 5.2 hours to process the 1500 orders. The number of orders changes relatively little from one batch run to the next so the model can be validated and revised at intervals. However, an increase in the number of transactions from Customer Service will be disproportionately concentrated during the daily peak. Although the traditional single system view of capacity planning is still valid, the much more complex nature of OLTP requires modeling tools to take into account the bursts of transaction arrivals and the increase in response times due to queuing in different parts of the system.

1.6.3 Client/Server Environment

The major change happens when the company realizes that the large number of different systems and user interfaces are causing problems for employees to communicate and making cross-training much too difficult. The company takes advantage of the existing LAN infrastructure and provides the users with a single common GUI based environment as shown in *Figure 3 The "Client/Server" System*.

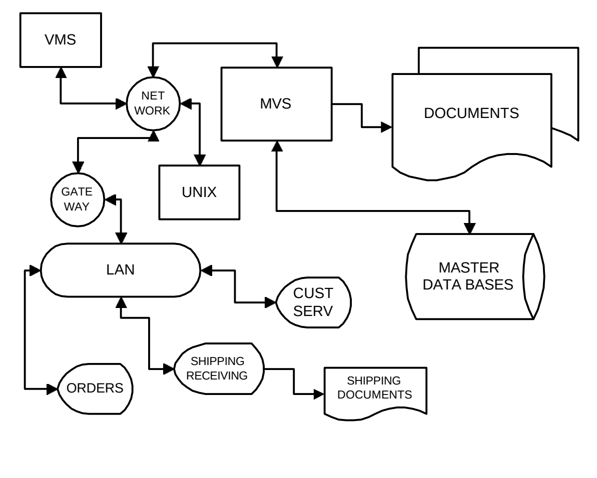

Figure 3 The "Client/Server" System

There are many different techniques and products to accomplish this and a discussion of these is beyond the scope of this paper. Which are chosen really doesn't matter, but what is important is understanding that any of the client applications on any of the PWS's can, and will, send transactions to several of the legacy applications to provide the end-user a screen of complete and interrelated information. For example, the Order Entry user may type in the name of an existing customer and get not only the customer's address but any pending or past orders and the status of the account. This may provide better service, but it also causes transactions to be sent to each of the other systems.

Capacity planning in a client/server environment is much harder because of the inter-relationships between systems. The systems are no longer isolated and independent. In the example above, if the Shipping workload outgrows the Shipping system, it can impact the responsiveness of the Order Entry transactions that are sent to that system. In addition, increased volume of the Order Entry transactions that are orders from existing customers will impact the Shipping workload because those orders require processing on the Shipping system.

Modeling in this environment is truly a challenge. Each of the systems requires a different knowledge base and expertise (Gunther 1995; Hatheson 1995). Each of the systems cannot be modeled independently because the transaction arrival rate for one system may depend on the response times of the others. The client software on the PWS may issue transactions to several servers (request everything about customer #123) or it may have to wait for one response before sending the next (what is Jones' customer number; then request everything about that number). While the former situation will cause the instantaneous peaks to synchronize on all of the servers, the latter will slow everything down as one of the servers becomes overloaded and its response times increase. “While it is important to be able to model specific UNIX or NT hardware, the problem we face is modeling the environment that has a diverse collection of hardware, operating system, database management system, and network hardware.” (Domanski 1995).

1.7 Planning the Future

The evolution of applications from centralized batch environments to client/server environments has introduced many problems for capacity planning. Many of the tools and techniques currently used to plan the future resource requirements of applications cannot accommodate the complexities of client/server environments. Section *2 Modeling Capacity Projections* examines these problems as they relate to one of the key techniques used by capacity planners: modeling the performance of applications to predict capacity requirements.

1.8 Problem Area

The problem addressed by Simalytic Modeling is at the intersection of several areas: capacity planning, hybrid modeling (both simulation and queuing theory), client/server systems (transaction processing) and commercial tools (both general purpose and platform-centric). No longer is the corporate data processing center a single large mainframe computer with ‘dumb’ terminals. The rapid increase in the client/server type of system has created a demand for capacity planning but the current tools and techniques do not yet address the total enterprise. The intersection of many problems has been synthesized into a single problem statement to identify both the direction and importance of the research in this dissertation.

1.8.1 Problem Statement

No modeling tools exist today that provide all the required features and functions necessary for planning the enterprise level capacity for applications that utilize complex multiple-platform computer systems. These tools are either platform-centric or general purpose (see sections *2.3.1 Platform-Centric* on page 40 and *2.3.2 General Purpose* on page 41), neither of which alone can provide both the required level of detail and the enterprise scope.

1.8.2 Hypothesis

It is possible to develop a viable modeling methodology that will use a general purpose simulation modeling tool as an underlying framework and utilize the results of an analytic modeling tool to represent individual nodes or systems when predicting the capacity requirements of an application at the enterprise level.

1.8.3 Proof Approach

The proof of the hypothesis is the evidence that the Simalytic Modeling methodology produces similar results to both simulation models and analytic models of a reasonable number of simple single-device multiple-server models and that it produces similar results to simulation models of a reasonable number of complex multiple-device multiple-server models designed to be representative of realistic client/server application situations. Because the arrival rate provides the bridge between the two types of models used to create a Simalytic Model, the average response times of the three models are studied for a range of arrival rates for each situation. Simalytic Modeling is shown to be a valid technique for solving client/server application modeling problems by producing results consistent with simulation models over what is assumed to be the entire range of practical situations. The technique is also verified to produce clearly delineated (i.e. precise) predictions, that is, predictions within a narrow range for the given input values, for the same range of practical situations.

2. Modeling Capacity Projections

The use of models to assess and predict the performance of computer systems is not new. A number of different modeling techniques can be used for capacity planning. The choice of which to use varies according to the needs at the time. Modeling is but one of the tools the capacity planner has to use, but it is a tool that has proven to be very effective. The following examples illustrate how modeling is an integral part of the capacity planning process.

In the mid-to-late 1980's, Rosenberg discussed one of the "Great Myths of Capacity Planning": the managers' view that modeling and capacity planning are one and the same (Rosenberg 1986; Rosenberg 1988). He addressed the misconception, along with several others, by clarifying the overall capacity planning process and the role modeling has in that process. There are several important observations that can be made from these works. First, by this time, capacity planners had accepted modeling as an accurate and effective technique for predicting the future requirements of a system. So much so, in fact, that management (and even some capacity planners) had begun to associate this single output from the capacity planning process with the concept of capacity planning. Second, understanding the input parameters to a model, such as the workloads involved and how those workloads will change over time, is a much larger part of the capacity planning effort than building the model to predict the impact of the changes. Third, because understanding the workloads is crucial to capacity planning, many of the modeling *tools* have included facilities to assist with collecting and analyzing data. This further associated modeling with capacity planning because the tools are involved in much more of the process, even though the final model may not be used. The model is only one of the latter steps in the process. And finally, there is a distinction between forecasting the workload volume and workload performance. Workload volume is an input to the model and describes the expected changes to the environment in business terms, such as transactions per second or orders per day. Workload performance is the prediction of the responsiveness of the application given the predicted volume and hardware configuration, which is generally assessed against a business related service objective.

An earlier example, from the late 1970's, is the work done by D. C. Schiller at the IBM Washington Systems Center to develop a modeling methodology called SCAPE (System Capacity And Performance Evaluation) which used simple queuing theory to develop system maps to describe the current and future systems and workloads (Hersh 1978). SCAPE system maps were graphical representations of current and future performance of either a workload or the entire system projected into the future some number of months. The goal of this, and the other research since, was to quantify workload behavior in terms of business expectations and to identify the limiting factors to achieving those expectations.

Capacity planning has always relied to some extent on modeling because of the need to predict future requirements. The relationship between the two is interesting in that either can be, and often is, done without the other. A capacity planner can analyze the workloads and make predictions based on experience or simple trending. Models can be constructed to understand how an application functions without any intention to predict future performance. The area of interest here is the intersection of the two fields: models used to predict capacity requirements based on performance expectations. Although it is easy to see how these models can be extremely simple or very complex, the issues are determining which of the different approaches to capacity planning can be used in a given situation and how to maximize the benefits of the selected approaches while reducing disadvantages.

2.1 Approaches to Capacity Planning Modeling

The approaches to capacity planning range from the application of rules-of-thumb to full scale benchmarks of the application or system (Brunetto 1984; Gilmore 1980; Hanna 1988; Mills 1991). *Figure 4 Capacity Planning Approaches* shows the relationship between these approaches. (This figure is not to scale; no meaning should be inferred about the absolute level of either axis, only that the relative position of an item implies some amount more than an item to the left or below.) Business analysis, rules-of-thumb, trends and linear projections rely on historical analysis and the assumption that future performance is a direct extension of past performance. Benchmarks can be the most accurate because they actually implement the applications, but at the greatest cost. Modeling is the middle ground between the high cost and effort of benchmarks and the low prediction ability of trends.

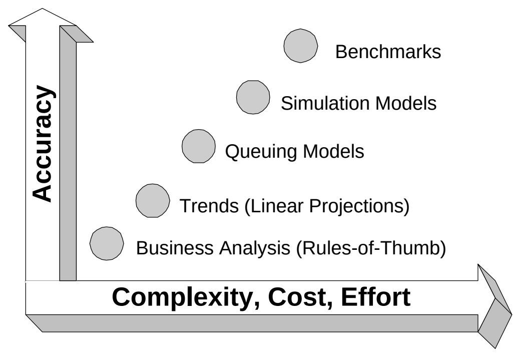

Figure 4 Capacity Planning Approaches

2.1.1 Business Trends

Many organizations have used rules-of-thumb to plan for business and application growth. Rules-of-thumb allow the capacity planner to watch resource related metrics, such as processor busy and disk I/O’s per second, and determine that additional resources are required when the metric exceeds the rule-of-thumb (Mullen 1988; Zimmer 1987). If the problem was not corrected (generally measured in complaints per hour), then other resources were added until everything was below the rules-of-thumb. The fact that there could still be unhappy users caused planners to look to other techniques such as forecasting.

Some of the simple forecasting techniques, such as historical trending or using Business Forecast Units (BFU), require a much lower initial effort and often produce an acceptable result. Just how acceptable depends on the required level of precision of the particular business (Harrison 1990). The BFU technique relies on linking some measurable business metric to the resources required to process all stages of the business included in that metric. For example, if the metric used was the number of invoices mailed, the resources associated with one invoice would include those resources required to enter the order, query the status when requested, ship the order and process the payment as well as print and mail the invoice. The assumption is that the ratio between these business activities will remain constant. If the number of invoices mailed doubles, then the number of each of the other activities will also double. It also assumes a linear relationship between resource consumption and the metrics, that is, if the number of invoices doubles, the required resources will double (this has not always proven to be a good assumption in actual systems). BFU's would then allow the required resources to be calculated based on the projection of future business (e.g. the number of invoices to be mailed per day next year). Harrison points out that using BFU's requires an understanding of how the business uses those resources (e.g. the peak-to-average ratio over a day) and how sensitive the business is to approaching the capacity limits.

Ted Keller, speaking at the 1989 International Conference on Management and Performance Evaluation of Computer Systems in Reno, Nevada, discussed the need to understand the nature of the business to be able to understand how business changes will impact resources (Anonymous 1990). He emphasizes the relationship between the past, in terms of historical data and trends, and the future, in terms of predicting complexity and growth.

2.1.2 Benchmarks

Benchmarks produce the best and most accurate predictions because they implement the applications as either a synthetic workload or as a test version of the actual application (Brunetto 1984; Ferry and Richards 1989; Harbitter 1990; Mohr 1979). In the truest sense they are not really predictions because they execute the application on the actual target configuration. The benchmarks being referred to here are application oriented as opposed to the more common hardware orientated benchmarks. Application benchmarks, such as (Carrasquilla 1991), are focused on predicting how a specific existing application will perform on different hardware and thus avoid many of the problems with hardware oriented benchmarks. A hardware oriented benchmark is designed to quantify the differences between processors that can be applied across a wide variety of workloads and therefore use very generalized workloads. Examples of hardware benchmarks are those from Specmark and the Transaction Processing Performance Council (TPC). A synthetic workload is a set of programs, jobs, scripts or any other components of the application carefully designed to have the same instruction and I/O mix as the real application but that do not actually do any productive work. A synthetic workload will usually produce very close to the same performance results each time it is executed (Harbitter

1990). Benchmarks require both the ability to create the workload and the actual target resources to run the benchmarks, which limits the ability to speculate about target configurations (Senson 1985). Current capacity planning research, such as (Carroll and Cool 1994) tend to dismiss benchmarks as too costly for general use.

Specialized benchmarks are often used to identify the performance characteristics of a type of hardware. An example is the work of Artis to establish what he refers to as the performance envelope of disk subsystems (Artis 1994a). This work does not predict a response time from a given workload, but establishes a matrix of input parameters (such as I/O rate, record size, locality of reference and read-to-write ratio) and the corresponding device response time. The user must determine how well the workload being analyzed matches the sample points in the matrix and how to extrapolate between points if that is required.

The results of a benchmark can be better predictors than a very well calibrated analytic model and could also be used to provide the node level response times in a Simalytic Enterprise Model of an application. Although the use of benchmark results are not addressed in this investigation into Simalytic Modeling, this appears to be a rich area for further research. Other studies, such as (Harbitter 1990), have investigated the use of benchmark results in conjunction with other modeling techniques. Harbitter's hybrid technique uses the benchmark results to validate the other modeling tool, but, as he assumes them to be correct, they could just as well be used in a larger framework model.

2.1.3 Modeling

Modeling provides more flexibility without increasing the complexity, cost and effort of the analysis beyond the means of most organizations. Capacity planning models are seen in a variety of different forms. The trade-off between these forms is the difference between the effort to produce the models and the accuracy of the resulting predictions. Hanna discusses the objectives and approaches to capacity planning in terms of how all of the different processes, including rules-of-thumb, forecasting and modeling, fit together (Hanna 1988). The general consensus seems to be that, as shown in Figure 4, modeling fits between rules-of-thumb/trending and benchmarking, with simulation being more costly than queuing theory (Brunetto 1984; Gilmore 1980; Hanna 1988; Mills 1991).

Even within the area of capacity planning modeling there are different approaches and techniques to address different capacity planning objectives. The modeling effort can be in response to a wide variety of questions: a consolidation effort that merges systems (Schwartz 1995); a concern for responsiveness as the business grows (Dwyer 1984); or a design tool when implementing a new and different environment (Biswas and Pietras 1988). Throughput (batch) versus responsiveness (transactions) also influences what type of model is best suited to the problem.

2.1.4 Summary of Approaches to Capacity Planning Modeling

The approaches to capacity planning modeling include analysis using rules-of-thumb, trends using linear projections, queuing theory models, simulation models and formal benchmarks. With any of these techniques there is a direct correlation between the level of accuracy of the results and the complexity, cost and effort to produce those results. The approach selected for any given capacity planning effort should reflect a balance between the desired accuracy and the level of effort to achieve that accuracy. In addition, new variations of these techniques attempt to increase the accuracy without a proportionate increase in the cost. Simalytic Modeling is one such variation that uses the more costly techniques only in the parts of an application model that require the increased level of accuracy.

2.2 Response Time Modeling

The key to the capacity planning methodology discussed so far is the ability to predict the performance of a future workload given a desired system configuration. As applications move more towards being transaction-based, the definition of application performance becomes centered around transaction response time. For this reason, modeling the transaction response time of an application is crucial to the ability to predict the future performance of that application. In addition, an understanding of the available modeling techniques is crucial to the ability to model transaction response times.

There are two basic modeling techniques used for computer performance modeling: simulation and analytic queuing theory (Kobayashi 1981; Menascé, Almeida, and Dowdy 1994). A third technique, hybrid modeling, is the combination of both simulation and analytic techniques in a single model (Kobayashi 1981). Although a full discussion of these techniques is beyond the scope of the paper, the following sections will provide the background required for the later discussion of Simalytic Modeling.

Either simulation or analytic modeling will build a model that represents the major components of the computer system to be modeled. Although there are exceptions, generally the resources of interest, when looking at the response time of transaction-based applications, are processor and disk. *Figure 5 Simple Transaction Model* shows a graphical representation of this model. Transactions enter the system from the left, wait in the processor queue and are served by the processor. At this point the transaction will either leave the system or move to the disk queue and then the disk server. Then the transaction goes back for processor service.

Figure 5 is a very common diagram of the life of a transaction *inside* the OLTP system, where reliable response time measurement data is available (Buzen 1984). This view is common because a transaction always starts and ends with some amount of processing, even if only enough to do a disk access or terminate the transaction. The transaction response time is a measure of the time from when the transaction enters the system until it leaves (Buzen 1984). In a system with a very low arrival rate there will be little or no queuing. When the system gets busier and the time between some transactions (the interarrival time) is shorter than the time to service the prior transaction, then queues will form for one or more of the servers. Even though the mean interarrival time formally defines the interarrival time of a queuing theory model, when exponential interarrival times are assumed (as with an M/M/1 model), the server can be saturated for part of the modeling interval because of the distribution of arrivals. Thus the nature of the mean tends to hide the peaks, both in terms of the arrival rate and the resulting response time. This phenomenon is exacerbated when the actual workload changes over the model interval and the mean no longer accurately represents the model interval and further hides the peaks. This situation is addressed in a simulation model using trace data because the queue time is calculated for each individual transaction. A queuing theory model, however, assumes a consistent mean for the entire interval. Although the

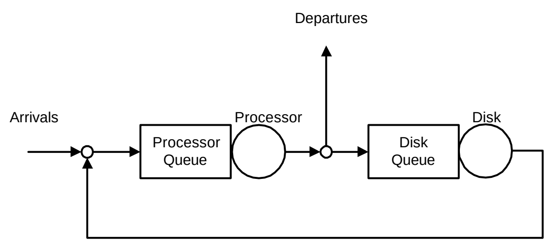

Figure 5 Simple Transaction Model

situation of short-term server saturation is accounted for in the queuing theory formula calculation of the average response time, the actual distribution of response times over the model interval is not identified. When the server is part of a larger application environment, this phenomenon can impact both the arrival rate and responsiveness of the workload at the other servers. Large variations in the actual workload arrival rate will make the interval unsuitable for modeling with a queuing theory model.

As applications become more distributed, the network becomes a larger component of the transaction response time. The central server view of an application tends to focus on the response time inside the server but the client/server view must include the network time. Network delay is not regarded as part of the transaction response time when planning the capacity of a system for two reasons. First, it is controlled by network hardware and communications lines, which are generally unaffected by processor or disk upgrades. Second, the network delay for a given transaction depends on the path through the network the transaction takes, which may vary greatly from one end-user to another or even from one execution of the transaction to another. In addition, client/server applications tend to group multiple OLTP transactions together for what the end-user sees as a single business transaction. This means that a model of the application that spans multiple systems must include network delays to provide an accurate representation of interdependencies between the systems in the enterprise (Domanski 1995; Linthicum 1997).

2.2.1 Analytic Queuing Theory

> "Analytical models capture key aspects of a computer system and relate them to each other by mathematical formulas and/or computational algorithms." (Menascé, Almeida, and Dowdy 1994, 45).

Analytic models can be implemented many different ways, from paper-and-pencil to spreadsheets to advanced commercial products. One analytic technique, queuing theory, plays the most dominate role in the area of capacity planning because capacity and performance problems are most often related to queuing delays caused by contention for resources within a system (Kobayashi 1981).

These models are generally more efficient to execute than simulation models, but they are also often less accurate because the mathematical formulae normalize all activity for each server in the model. For example, building a simple analytic model from actual transaction performance data would result in a single workload based on the average transaction arrival rate, the average processor time and the average disk time, with each following some probability distribution. If there is very little difference between the actual transactions and these distributions, then this model will provide very good results. However, the much more common situation is that the variation in the transactions will result in a model that is difficult to validate and poor at prediction because most of the actual transactions are not represented by the average calculated in the model.

Determining how closely the average transaction matches the actual transactions, and then modifying the model to improve that match, is model calibration. When a model cannot be calibrated it is often because the workloads being modeled are not homogeneous. The workloads must then be restructured in the model to reduce the variation between the average transaction and the actual transactions. This is generally referred to as workload characterization. Although workload characterization is a very important preliminary step to any modeling activity, it is beyond the scope of this paper. The interested reader will find a full discussion in (Domanski 1995; Menascé, Almeida, and Dowdy 1994).

Equation 1 Analytic Response Time Formula

$$T = \frac{S}{1 - l\ S}$$

The mathematical formula for the average response time (*T*) of transactions in a simple single-server analytic model is shown in *Equation 1 Analytic Response Time Formula* from (Menascé, Almeida, and Dowdy 1994, 108) where *S* is the average time for the server to complete its function (service time) and l is the average arrival rate of transactions. A detailed description of this formula and its application is also presented in (Buzen 1984).

Queuing systems are described using a notation invented by David Kendall. It has the form A/B/c/K/m/Z where A is the interarrival time distribution, B is the service time distribution, c is the number of servers, K is the capacity of the system (maximum number of customers allowed or maximum queue length), m is the total source population and Z is the queue discipline (Allen 1990, 257). The model generally used for predicting the capacity or performance of transaction systems is an M/M/1 model (exponential interarrival time distribution, exponential service time distribution, a single server, K and m are assumed infinite and Z is assumed FIFO (first-in, first-out)). Although there are many different distributions for A and B, generally M (exponential) is used because it more closely approximates the actual behavior of transaction systems (Buzen and Potier 1977).

Exponential interarrival time distributions (also referred to as Poisson arrival distributions) have properties that also make them very useful for client/server models. It has been shown that the arrival rate at a server from multiple sources is the sum of the arrival rates generated by each source. Also, the arrival rate at one of multiple servers from a single source is the original rate times the probability that server will be selected. Exponential service time distributions are also useful because of the Markov or “memoryless” property (the service delivered to one customer does not predict the service that will be delivered to the next) (Allen 1978; Allen 1990, 255; Kobayashi 1981, 103-5; Pooch and Wall 1993, 342-3). Because the mean and the variance of an exponential distribution are equal, it is relatively easy to analyze the interarrival times of real transaction data to determine if an M/M/1 model is appropriate (Pfeiff 1993). The key advantage to using an exponential distribution is that it simplifies the model because the analyst only needs to estimate a single parameter, the mean interarrival or service time. Other distributions require two or more parameters, such as mean and variance for a general model. Even if the actual distribution is not exponential, early work has shown that the error when modeling non-exponential servers with an exponential model is acceptable and that real-world systems tend to limit the variability such that they also limit the errors (Buzen and Potier 1977).

Some work has been done to understand and predict the impact that variations in arrival time distribution have on the results of queuing theory models. An example is Whitt’s analysis to understand the effects of large fluctuations in queue arrivals and departures and the relationship to different service disciplines using class-dependent service times (Whitt 1993). Whitt found unstable behavior (queue lengths going to infinity) with the FIFO (first-in, first-out) discipline when systems approach the heavy-traffic limits as traffic intensities approach one. Related works investigated the lag between the peak arrival rate and the peak server congestion (Eick, Massey, and Whitt 1993) and the time-dependencies of workload processes in M/G/1 (exponential interarrival time distribution, general service time distribution, single server) and M/M/1 queuing models (Abate and Whitt 1994). These investigations attempt to develop techniques to compensate for the difficulty queuing theory models have with heterogeneous distributions and workloads, something simulation models address much more effectively. For example, Buzen and Shum examined the application of queuing theory models in conjunction with the IBM Parallel Transaction Servers (Buzen and Shum 1994) without fully addressing the impact of workload variation. That analysis was expanded to include an understanding of work-load variability using a simulation model in (Norton 1995).

2.2.2 Simulation

> "The simulation model describes the operation of the system in terms of individual events of the individual elements in the system. The interrelationships among the elements are also built into the model. Then the model allows the computing device to capture the effect of the elements' *actions* on each other as a dynamic process." (Kobayashi 1981, 221).

There are many types of simulation models and techniques. Trace-driven simulations are of more use in capacity planning and performance modeling because they remove one major issue in model construction: transaction arrival distribution. Self-driven simulations generally assume some distribution pattern such as normal, Poisson, exponential or Erlang distributions, which may or may not represent the actual distribution of application transactions (Kobayashi 1981). Regardless of the underlying technique used in the simulation tool, there are two important characteristics of these tools. The first is the ability to maintain the identity of each transaction, and its associated attributes, throughout the entire model and have the model react to these attributes. The second is the ability to preserve the specifics of the interarrival times between individual transactions. Even though the Poisson distribution is accepted as generally the most representative in models of these types of systems (see the discussion in section *2.2.1 Analytic Queuing Theory* on page 30), other distributions can be used if analysis of the trace data shows another distribution to be more appropriate.

The mathematical formula to estimate the average response time (T) of transactions in a simple single-server simulation model is shown in *Equation 2 Simulation Response Time Formula* from (Menascé, Almeida, and Dowdy 1994, 108) where T_i is the response time of the i^{th} transaction and n_t is the total number of transactions that visited the server during the simulation.

Equation 2 Simulation Response Time Formula

$$T = \frac{\sum_{i=1}^{n_t} T_i}{n_t}$$

Simulation is a very broad topic and cannot be addressed here in detail. However, it is important to understand the main idea behind simulation: to represent some physical or "real-world" situation with a tool that allows for greater analysis than can be done with the original events (Fishwick 1995). Different techniques are appropriate to model different aspects of the same situation. Fishwick uses an example of a Petri net to simulate the throughput of the manufacturing line. He then points out that the Petri net would not be suitable to analyze the stability of the robot arm controllers.

Understanding the throughput or performance of a system given an increased future load (its capacity) is one of the very common uses of simulation models. An interesting use of simulation, considering this paper's focus on capacity planning, is the discussion of Simulation Based Planning in (Fishwick, Kim, and Lee 1996). This system allows military strategists to analyze the impact of different "moves", much the same as the capacity planner analyzes the impact on workload performance of different business projections.

Simulation models can be either continuous or, of more interest here, discrete. Within discrete simulation there are different types of models depending on the main focus of the representation and how the events being simulated are synchronized. *Event orientation* focuses on the identification of events, activities associated with events and the system state changes as events occur. *Activity orientation*, or *activity scanning*, is used to model situations of continuous change and the state of the activities is checked against predefined conditions with every small simulation time increment. *Process orientation* sees events as being grouped together into a process where events are related in time. It is most useful for queuing systems and includes transactions or *transaction orientation*. (Pooch and Wall 1993)

This description is meant to provide the general idea of the types of simulation models and is not intended as a comprehensive discussion of simulation techniques. Other sources use somewhat different terminology. For example, one source refers to *flowchart orientation* instead of *transaction orientation* (Anonymous 1996) and there are some differences from (Pooch and Wall 1993) in the definition of *process orientation*. (Schriber and Brunner 1996) only discusses the “transaction-flow world view” in conjunction with discrete-event simulation and implies the other world views are not worth discussing. It is expected that any simulation approach selected would match the requirements of the environment being modeled. Any of the techniques discussed are acceptable if they meet the criteria set forth in section *3.2.1 Methodology Assumptions* on page 54. Schriber and Brunner point out how the internal designs of different tools impact models. Their examples show the importance of selecting the correct tools for the environment to be modeled with illustrations of current common simulation tools that each produce different results for the same situation (Schriber and Brunner 1996).

2.2.3 Hybrid

> "A hybrid synthetic model consists of both subsets of the real workload and specially constructed components." (Menascé, Almeida, and Dowdy 1994, 44).
>
> "*Hybrid modeling*: a combination of analytical procedures and simulation. So long as the interfaces between different levels or submodels are clearly established, the mixing of analytic and simulation techniques should present no technical problems." (Kobayashi 1981, 20).

The word 'hybrid' means "Something, such as a computer or power plant, having two kinds of components that produce the same or similar results." (Anonymous 1992,1994) and it has been applied very liberally to describe modeling techniques. A hybrid model combines two or more different techniques for a variety of reasons, such as complexity reductions, performance improvement or analysis flexibility.

There are many examples of hybrid models across a wide range of disciplines that illustrate both variety and creativity at combining different techniques:

- One hybrid technique for workload analysis uses either analytic or simulation modeling to represent identical workloads on different levels of detail (Lehmann 1984).
- A technique for determining if adding workloads to a system will cause excessive paging uses a two stage approach where the output from the first stage (performance measurement analysis) is input to the second stage (closed queuing network model) (Place 1986).

- A hybrid technique in CAD/CAM design systems uses a combination of explicit and parametric modeling to improve flexibility and provide the capability to import legacy or outside supplier data (Taylor 1994).
- (Elmqvist, Cellier, and Otter 1996) identify hybrid models as those that contain both continuous and discrete parts. They determined that different techniques were appropriate to model these different parts and discuss approaches to address discontinuous behavior in a reusable object-oriented fashion.
- A hybrid model to solve the distributed environment file allocation problem uses simulation to model query-by-query communications delays and an analytic model for average communication delay (Ghosh, Murthy, and Moffett 1992).
- (Sobh and Benhabib 1996) discusses hybrid modeling as modeling hybrid systems, which are systems, such as robotics, with interaction between digital and analogue devices and sensors over time.
- (Liu, Park, and Miller 1996) propose a hybrid synthesis approach to the constructing of Petri nets by developing them both top-down and bottom-up simultaneously.
- A very interesting approach from Lockheed Martin Advanced Technology Laboratories is to model hardware and software together (Schaming 1996). Schaming refers to this as a multi-domain simulation because different types of simulations or algorithm level data flows are used to model system components.

- Thomasian and Gargeya use a hierarchical technique to improve the performance of their models of memory-constrained timesharing systems (Thomasian and Gargeya 1984). They showed that a hybrid approach could be used where a simulation replaces the large set of linear equations for the corresponding Markov chain.

These examples of hybrid models show some commonality in relying on submodels and problem decomposition. They develop a technique that allows the best features of each tool to be focused on the problem by categorizing the problem into different parts. The concept of submodels, similar to subroutines in programming, allows each part of a model to be represented by a different technique. Submodels are supported by most of the commercially available modeling tools, but with varying abilities to utilize completely different techniques in the submodel. Submodels are a key concept because they allow some part of the model to be replaced with a different model as long as it provides appropriate functionality and results, similar to the FESC (flow-equivalent service center) decomposition technique discussed in (Menascé, Almeida, and Dowdy 1994). The specific uses of hybrid models in capacity planning and computer performance will be discussed in greater detail in section *2.4 Applicability of Hybrid Modeling* on page 42.

2.2.4 Response Time Modeling Summary

The three modeling techniques discussed above, simulation, queuing theory and hybrid, provide the basis for the development of the Simalytic Modeling Technique. Both simulation and queuing theory models have advantages and drawbacks when modeling transaction-based workloads. The application of hybrid techniques uses the advantages of one technique to offset the drawbacks of the other technique.

2.3 Modeling Tools

In addition to the choice between analytic and simulation tools, the capacity planner or performance analyst has the choice between platform-centric and general purpose tools. The basic difference between these two groups is the problem set the tools were designed to address.

2.3.1 Platform-Centric

Platform-centric means that the tool contains detailed information about the platform, but does not allow more than one platform to be modeled at a time. Examples of platform-centric information include the number and type of processors for each model of a system built by a given vendor (e.g. IBM 9021-982 = 8 processors @ 60 MIPS each), the speed and transfer rate of disk devices by manufacturer (e.g. EMC 5500-128 Read (cache hit) = 1.5 ms @ 4.5 MB/sec, Read (cache miss) = 13.5 ms @ 4.5 MB/sec, Write = 1.5 ms @ 4.5 MB/sec; all for a 4K block of data), and operational issues based on the level of the operating system (e.g. MVS/SP 1.3, MVS/XA 2.2, MVS/ESA 5.1).

Platform-centric models are generally easier to build because they are made of “building blocks” already defined to the tools and the relationships between them are fully understood by the model. However, these tools cannot be used to model an environment not built into the tool. For example, a tool with the above “building blocks” could not be used to model Unix running on an HP system. Although many platform-centric tools allow the user to define new servers with new performance characteristics, they generally do not provide large libraries of device and system definitions dramatically different from the intended platform. Platform-centric tools are *generally* implemented using analytic or queuing theory modeling techniques and process performance and configuration data collected from existing running systems.

2.3.2 General Purpose

General purpose means that the tool allows users to model anything they would like to build, but with little or no “built-in” understanding of any given platform. These tools are used to model more than just the hardware, including such things as the design of an application, traffic flows, and communications networks. Using the example above, the model builder would have to understand the design of the IBM 9021-982, how the eight processors communicate, and what is involved in completing a unit of work within the operating system. A platform-centric tool might contain a variable for the delay caused by dispatching a task on a different processor and set the value of that variable when the model starts based on the system vendor and model, the operating system version and type, and the current configuration (such as memory). The modeler using a general purpose tool would be required either to build a sub-model to simulate the underlying architecture or to determine a delay value to use whenever the event happened. Simulation of the architecture can be much more accurate but also much more difficult and time consuming.

Although many general purpose tools provide libraries of sub-models for a variety of systems and devices, they generally do not provide the required level of granularity, being either too general or too detailed for the situation. Modification of these submodels, if allowed by the tool vendor, can be time and effort intensive. Building the relationships between the submodels is generally part of the overall model construction and may require an in-depth understanding of all of the submodels used. General purpose tools are *gener-ally* implemented using simulation modeling techniques.

2.3.3 Modeling Tools Summary

The distinction between platform-centric and general purpose modeling tools is based on the set of problems they are designed to address. Platform-centric modeling tools are focused on in-depth knowledge of a single platform using relatively large prede-fined components while general purpose modeling tools are designed to address a larger variety of problems using relatively simplistic primitive functions and controls. Predefined components can represent devices, computer components, or even entire subsystems, in-cluding large amounts of knowledge about the behavior of those components. On the other hand, primitive functions and controls allow the implementation of models at a much lower level of granularity and detail. The selection of the type of tool is a compromise that tries to fit the tool to the majority of the requirements of the problem. Most problems have requirements addressed by both types of tools and would be more accurately and efficiently solved by a combination of the tools using a technique developed to combine the best features of both.

2.4 Applicability of Hybrid Modeling

Simalytic Modeling is a hybrid modeling technique combining the different model-ing techniques (simulation and analytic queuing theory) and the different tools (platform-centric and general purpose). The reasons for using a hybrid technique for capacity plan-ning modeling are fundamentally the same as the reasons other areas have turned to hybrid techniques: accuracy improvement and cost/effort reduction. By merging the best fea-tures of the two techniques, along with the use of existing modeling tools, the capacity planner can produce a more accurate model with less effort and lower processing requirements. It is not the intent here to establish the definitive proof as to the overriding superiority of either type of modeling. Rather, the goal is to show that a combination of the techniques will produce results equal to the results produced by either technique alone and that a hybrid technique is better suited to utilize the results of platform-centric models an organization may already be using.

This view of the applicability of hybrid modeling has influenced the approach to validation and verification of Simalytic Modeling. Because the goal of Simalytic Modeling is to produce results equal to either simulation modeling or queuing theory modeling, the testing will focus on the comparison of models of applications of varying complexity using all three techniques. This approach is similar to that taken in (Ahn and Kim 1994) and (Thomasian and Gargeya 1984).

2.4.1 Analysis of Existing Hybrid Techniques

Analyzing the impact of current research on this dissertation topic has been very difficult because no work could be found directly addressing the topic. The uses of hybrid models are discussed in papers dealing with capacity planning, such as (Thomasian and Gargeya 1984), and in papers dealing with distributed systems, such as (Barroso and Dubois 1995). However, generally both of the modeling techniques are custom developed for the particular problem, although some research has been done using one commercial tool implementing one part of the hybrid design. Commercial tools address the areas of capacity planning and client/server, but these tools, such as Best/1 (Buzen 1995), are limited to a single modeling approach. The tool CMWLan builds an analytic model for capacity planning LAN environments (Menascé and Dregits 1995; Menascé et al. 1995), but it deals with averages (transaction arrivals and service times) and relies on general models of the nodes rather than using the results of validated platform-centric models. *Figure 6 Dissertation Topic Relationships* shows a Venn diagram of the relationships between these different topics as they apply to this dissertation. There is some overlap between the different topics, but the very small section in the center of the diagram is the focus of this study. Therefore, the emphasis in looking at current works is not on the direct application of hybrid techniques to capacity planning for client/server systems, but on understanding if those techniques support the application of hybrid modeling as discussed in Simalytic Modeling.

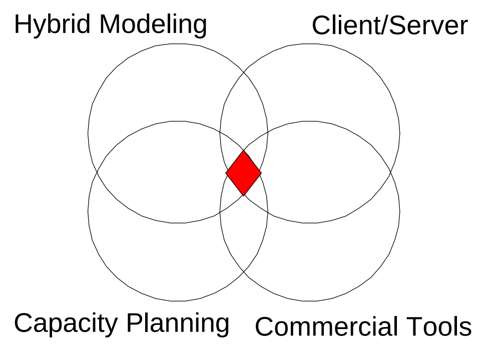

Figure 6 Dissertation Topic Relationships

The trade-offs of accuracy and cost between simulation and analytic modeling are directly related. With either technique, the higher the level of desired accuracy the greater the cost to produce the model. (Cost refers to complexity, effort and resource consumption, all of which are reflected in the financial cost to build and run the model.) The efficiency of hybrid techniques over simulation techniques has been well established and the results from hybrid techniques have been shown to be acceptably close to the results of pure simulation models but at a lower cost (Ahn and Kim 1994; Barroso and Dubois 1995; Baylor, Benveniste, and Boelhouwer 1994; Schwetman 1978; Thomasian and Gargeya 1984). Unfortunately, one weakness with these, and other published papers, is that they are generally investigating a single problem and only describe the use of the models in general terms. None of the papers provide an in-depth discussion of the techniques used, only the results and the validation of those results. Therefore, a generalized hybrid technique for client/server capacity planning, along with a fully documented methodology, seems appropriate. The rest of this section discusses some specific implementations of hybrid techniques to address some problems somewhat related to capacity planning.

Barroso and Dubois use a hybrid methodology of analytical models and a trace-driven simulation model to investigate the performance of the unidirectional slotted ring interconnections for small to medium scale shared memory systems (Barroso and Dubois 1995). They used the simulation model, driven by actual trace data, to generate points of performance information for each set of variables for a given design. They then used analytic formulae to generate additional performance points by extrapolating the input parameters. This approach is a technique to increase the number of resultant data points beyond what could be generated using only a simulation model. Each of the simulation runs required 6 to 8 CPU hours on a Sun SPARCStation2, whereas the analytical models generated a complete set of curves in less than one second of CPU time.

Benveniste and Boelhouwer use a similar hybrid methodology: a simulation to collect data for a small number of performance points and the derivation analytic formulae to fit a curve to those points to understand the performance of parallel I/O subsystems (Baylor, Benveniste, and Boelhouwer 1994). They use an iterative technique for validation and also show acceptable correlation between the simulation results and analytic results. Neither of these techniques take advantage of a platform-centric analytic model; the analytic models are composed of custom formulae designed specifically for the problem. However, they do show the advantage of using an analytic model to improve the performance of the simulation model.

Schwetman provides some foundational work in hybrid models of computer system performance (Schwetman 1978). His work is referenced in many of the other works and provides definitive results to support the efficiencies of hybrid techniques. Although his work centers around analysis of continuous versus discrete events (in terms of long-term and short-term resources), it still shows the usefulness of replacing a complex sub-model with a functional analytic equivalent. The author has discussed the ideas of hybrid modeling with Dr. Schwetman in recent personal correspondence (Schwetman 1996). Unfortunately, he has not done active research in this area since his early work and is now concentrating only on simulation models at Mesquite Software, Inc.

The most generalized hybrid modeling methodology was developed by Ahn and Kim as a framework to transform selected simulation submodels into analytic ones (Ahn and Kim 1994). They also show results that support the efficiency of hybrid models. Because they are able to transform existing simulation submodels into analytic submodels, they are also able to compare the results and the costs of both. They show that, if a sub-model meets the criteria for transformation, an analytic submodel will produce results acceptably close to the original simulation submodel and will require significantly less resources in doing so. They do not attempt to prove that either submodel is an accurate representation of any real situations. It is assumed that the original simulation submodel was validated prior to the transformation.

Miller and Fishwick discuss techniques for different modeling formalisms to interact with each other in terms of data exchanged and inter-model coordination (Miller and Fishwick 1992). They develop a general modeling approach called Hybrid Heterogeneous Hierarchical Modelling (HHH Modelling) which supports multiple representations and hierarchical development. Even though their focus is on the area of Knowledge-Based Autonomous System Simulations, their work demonstrates that a methodology can be developed to promote intra-model coordination. It supports a wide range of modeling techniques and establishes a new modeling formalism to be used when constructing a model. Their emphasis is on providing a way to analyze a single model using different techniques (symbolic, numerical and interpretative) to insure that all aspects of the situation are addressed. However, their technique does not allow for the integration of platform-centric model results as it assumes that all of the model components are under the control of the model builder.

2.4.2 Hybrid Modeling Applicability Summary

The works discussed in the previous section are representative of current research in hybrid modeling and show that, although it has many different meanings, it provides valuable advantages over single methodology models. It is a useful technique with wide application across diverse areas that can be applied to the combining of modeling tools as well as to the combining of modeling techniques. In addition to improving model accuracy and reducing model cost, hybrid techniques avoid the necessity of making compromises when applying a tool to a problem. They allow multiple tools to be used where each is applied selectively to the part of the problem most appropriately addressed by that tool.

2.5 Problem Area Discussion

The problem addressed by Simalytic Modeling is at the intersection of several areas: capacity planning, hybrid modeling (both simulation and queuing theory), client/server systems (transaction processing) and commercial tools (both general purpose and platform-centric) as shown in the Venn diagram in *Figure 6 Dissertation Topic Rela-*

tionships on page 44. Previous sections have discussed each of the areas and, to some extent, the interrelationships between them. No longer is the corporate data processing center a single large mainframe computer with ‘dumb’ terminals. The rapid increase in the client/server type of system has created a demand for capacity planning but the current tools and techniques do not yet address the total enterprise. Investigations such as (Anonymous 1992; Artis 1994b; Bleker 1994; Domanski 1983; Domanski 1995; Pooley 1992; Salsburg 1994; Wallace 1995; Zibitsker 1994) are beginning to describe the problem in greater detail, but they do not provide a total solution.

Domanski describes the changes to the environment across the broad spectrum of capacity planning, forecasting, modeling, reporting and management (Domanski 1995). His questions bring into focus the limitations of the current tools and techniques to deal with the growing complexity of the increasingly distributed enterprise. He identifies the key challenges in client/server as instrumentation, data reduction and summarization, workload characterization, and reporting. He concludes that “Modeling tools must become more graphical to allow topologies to be easily defined, and must then address modeling heterogeneous combinations of hardware and system software platforms.” (Domanski 1995, 12). His earlier work establishes a foundation in these areas in terms of what was, at the time, traditional data processing (Domanski 1983). The evolution of data processing from then to its current level is reflected in the problems he has identified in planning for heterogeneous environments.

Artis has identified the lack of a way to even describe applications in this new environment as enough of a problem to develop a graphical application representation (Artis 1994b). His technique addresses the business process with facilities for split/join, fan out and complex elements. It then provides a mapping to associate each stage in the process to the corresponding hardware layer and provide performance and utilization reports. Although he briefly discusses the use of models to predict the applications performance (and the use of Software Performance Engineering (Smith 1990) during the design phase), he does not propose the integration of his technique with an existing modeling tool. However, his graphical representation exhibits a striking resemblance to many of the current simulation modeling tools.

Wallace discusses the importance of capacity planning in a distributed transaction environment and a generalized approach to modeling such an environment using system utilization and linear regression (Wallace 1995). This approach provides only general estimations and only addresses response times by comparing the utilization results to performance benchmarks such as the Transaction Processing Performance Council (TPC) benchmark. This provides only an approximation of expected response times and is only valid if the workloads under study are accurately represented by the benchmarks. He makes a case for understanding application performance across the enterprise, and even develops a methodology to record end-user response times. However, his technique does not provide for analysis beyond node level utilizations.

Zibitsker addresses the use of analytic modeling when planning the capacity of MPP (Massively Parallel Processing) systems (Zibitsker 1994). He discusses how performance modeling can be used both for the evaluation of different hardware and to assist in developing new MPP applications. Because the MPP “shared nothing” environment is a group of processors connected by a very fast network, there is a high degree of similarity between Zibitsker’s work and the challenges of client/server environments. Although Simalytic Modeling appears to be very applicable to capacity planning in MPP environments, this topic will not be addressed here and is left as an area of further research.

Bleker examines the role of transactions in capacity planning for a distributed environment (Bleker 1994). He discusses the different types of mappings between end-user actions (business transactions) and measurable units of work (system or database transactions) and the relationship each has to the Unix process. He focuses on modeling disk accesses because his presumption is that the client system does GUI (Graphical User Interface) processing and the server provides database functions using an access mechanism such as NFS (Network File System). He identifies many of the problems with the current level of capacity planning for client/server systems, and although he emphases the importance of capacity planning in general, he does not present any significant solutions.

Pooley describes an implementation of performance evaluation techniques (both system and application) and tools to support those techniques including modeling, measurement and use of the results (Pooley 1992). He discusses the four levels of development of performance modeling tools: simple (single analytic or simulation tool), multiple solver (single modeling language driving multiple solvers that are a combination of numerical, analytic or simulation), integrated methodology (closer to system description than modeling) and modeling environments (both solution techniques and formalisms for expressing models). He also discusses the value of the integration of these tools into a single tool set which includes a graphical interface, a workload analysis tool, a structural description tool, an experimental plan tool, an object store (in this context objects include models, plans, objectives, reports, results, etc.) and an interface for using either existing modeling tools or custom tools. The modeling tools currently implemented are stochastic Petri nets

(PNT), queuing networks (QNET, which uses the QNAP2 product) and process interaction view (PIT, which uses DEMOS models).

Salsburg discusses the distributed client/server OLTP environment from the network viewpoint (Salsburg 1994). He addresses the complexities associated with understanding the flow of messages between many nodes and the impact on transaction response time as the network’s traffic increases. He identifies model complexity as one of the major factors in the effective use of the results by the business.

At the ICCM's 5th Capacity Management Forum, Tom Bell, president of Rivendel Consultants, talked about the emerging distributed computing environment (Anonymous 1992). He discussed how capacity plans must be based on both economics and technical aspects because the performance impacts are more pronounced in a distributed environment than in a mainframe environment.

The intersection of all these areas has been synthesized into the single problem statement presented in section *1.8 Problem Area* on page 18 to identify both the direction and importance of the research in this dissertation.