Don’t Predict Applications When You Should Model the Business

Dr. Tim R. Norton
Colorado Technical University
CMG98 Session 6201, December 11, 1998

Effectively managing a business requires a complete understanding of everything that influences delivery of the final product or service. Using a business simulation tool allows a model to be constructed of the entire business. Then, instead of planning the capacity of individual systems or the responsiveness of transactions, an application can be analyzed within the context of the entire business. Simalytic™ Modeling[1] is used to connect system and application models to the overall business model. A business modeling tool is used as the framework to address the non-computer aspects of the application. The computer parts of an application are represented with the results of modeling tools already in place. Examples of hypothetical applications, along with the steps to construct a business model, show how to integrate application responsiveness into a business model. This combination of modeling techniques provides an overall picture of the business while still maintaining an accurate understanding of the impact of changes in transaction response times.

1. Introduction

We model today’s computer environments with the objective of understanding the ability of the systems to meet the end user’s requirements. By addressing the business needs, we try to avoid viewing the modern computing environment as an end in itself where ‘state-of-the-art’ is equated to ‘advantageous’. Unfortunately, there are many pressures that try to shift the focus to the advances in technology without regard to what the computing environment provides to the business it was intended to support. Because the Data Processing industry is changing at a very rapid pace, it is often difficult to relate the value of many of the changes to the overall business in an objective manner. (Business in this context means more than a for-profit company. It can include any type of company, institution, agency, or organization with an overall objective, be it revenue, service or regulatory.) New applications that long ago would have been implemented as batch systems on a single computer are now multi-platform on-line transaction processing client/server systems combining departmental servers and mainframe repositories. Such complex application designs utilize the features and services of different types of computers (i.e., mainframe, mid-range, desktop) and often impact several aspects of the overall business. Techniques to model individual systems have been well understood for some time. Techniques to model applications have gained sophistication and popularity over the last several years. The challenge now is to develop a technique that connects these system and application models to the overall business models.

There are many business modeling tools (see HPS; Powersim; Vensim) to help planners analyze the different aspects of the business and determine the correct organizational structure and course of action. There are also many application and system modeling tools to help the performance analyst and capacity planner size and configure the computing environment. As applications move into the client/server world, how do we understand the impact of technology decisions on the overall business? What are the criteria to select the right systems at each level and, once selected, how do we insure those systems are providing the expected advantage to the business? If the technology does not address the business needs, the application, and the even the business, can fail. If the technology provides features, functions or performance that the business does not require, the cost of running the application may exceed the revenue it generates. Neither is a very attractive situation.

Capacity planning has traditionally focused on determining if a given system has “enough capacity” to service a workload (which, in theory, translates to an application). 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 also the relationships between them and the business objectives (such as staffing levels and “widgets” sold). This relationship allows projected business volumes to be modeled to predict the capacity required to meet those goals. Instead of planning the capacity of individual systems, the responsiveness of the application needs to be predicted across the entire enterprise. But even when the overall application responsiveness is acceptable (i.e., meeting the current service level agreement), the rela-

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

Permission is granted to publish this article in the 1998 Computer Measurement Group Conference Proceedings

tionship to the overall business process must be determined. This insures the computing environment is addressing the requirements of the business objectives and goals.

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). In addition, there are tools to model the business process (HPS; Powersim; Vensim) which provide the same type of simulation analysis for overall business processes. The Simalytic Modeling Technique (Norton 1996) was developed as a bridge across the existing computer modeling tools enabling the construction of an enterprise level application model. Such a model takes advantage of models and tools already in place for planning the capacity of each system. The same technique can be applied to the process of modeling the overall business by incorporating the results of the system and application models into the business simulation. Thus tools such as Best/1 (BGS) can be used to model the systems, tools such as Workbench and Strategizer (SES) can be used to model the application and tools such as Powersim (Powersim) can be used to model the business. The Simalytic Modeling Technique can be used to connect the different levels of models (system to application and application to business). The advantages of using Simalytic Modeling include a much better understanding of how changes in the computing environment impact the business as a whole. Also, each level is implemented with minimal effort because it uses the most appropriate tool.

2. Simalytic Modeling Review

This section provides a very brief overview of Simalytic Modeling for application modeling. Additional detailed information about Simalytic Modeling is available in other papers published and presented by the author. A detailed description of the technique and a discussion of the changes that provided the impetus for Simalytic Modeling are available in (Norton 1996). Details of the mathematical foundation and validation of the technique are available in (Norton 1997b) and (Norton 1997c). Preliminary results of this implementation technique, using a simple non-computer specific general purpose simulation tool, are presented in (Norton 1997a).

What is Simalytic Modeling? Simalytic Modeling (from **Sim**ulation and An**alytic**) is a modeling technique that uses a general purpose simulation modeling tool as an underlying framework and the results of an analytic modeling tool to represent individual nodes or systems. The results of the system models are used to characterize the application performance at each node, or system, within an enterprise level model of the application. The goal is to predict the capacity requirements of a client/server application executing on heterogeneous computer systems by creating an enterprise level application model.

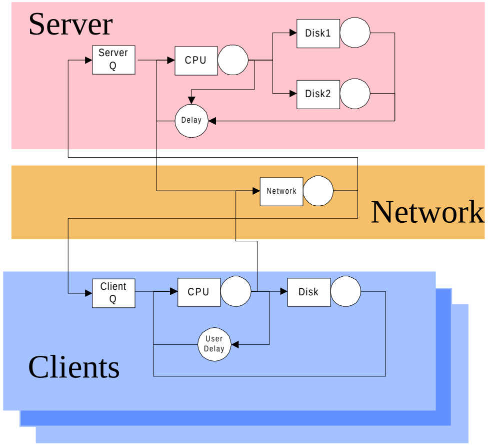
2.1 Background Topics

Before discussing Simalytic Modeling, it is important to review the background topics that support both Simalytic Modeling and the business modeling technique known as System Dynamics (Cover 1996; Richmond 1994). Additional information is available in the works mentioned above and in the references cited in those works. The remaining sections of this paper assume the reader is somewhat familiar with each of these topics:

- **Capacity Planning:** determining the capacity requirements of a computer system to support business growth.
- **Transaction Based Applications:** when the end-user enters a relatively small independent unit of work into the system and receive some information as a response in near real-time; often referred to as OLTP (On-Line Transaction Processing).
- **Client/Server Environments:** independent computer systems inter-related by the application implementation.
- **Modeling Capacity Projections:** the use of models to assess future capacity requirements.
- **Response Time Modeling:** modeling the response time of application transactions to predict the performance of current computer systems at an increased business load.
- **Platform-Centric Modeling Tools:** tools that contain detailed information about the platform, but do not allow more than one platform to be modeled at a time; *generally* implemented using analytic, or queuing theory, modeling techniques.
- **General Purpose Modeling Tools:** tools that contain features to allow the user to model almost anything, but with little or no "built-in" understanding of any given computer platform; *generally* implemented using simulation modeling techniques.
- **System Dynamics:** a methodology for modeling business processes that includes functions to address all aspects of most types of businesses.

Modeling in a client/server environment is a challenge because each of the systems requires a different knowledge base and expertise (Gunther 1995; Hatheson 1995). The systems cannot be modeled independently because the transaction arrival rate for one system may be dependent on the response times of the others. *Figure 1 An Enterprise Model* shows a very simplistic model for each of the major areas of a client/server application and, although it only shows a single server, the interdependence is evident.

System Dynamics is a very broad subject and a full discussion of it cannot be attempted in this paper. Cover provides a concise definition as: "System dynamics is a computer-based simulation modeling methodology developed at MIT in the 1950s as a tool for managers to analyze complex issues." The meth-



Reproduced with permission from Dr. Connie Smith, Performance Engineering Services.

Figure 1 An Enterprise Model

odology has begun to be used in many disciplines, but the primary audience remains business managers. (Cover 1996, 5) The focus here is not to provide an in-depth discussion of System Dynamics, but to show how business simulation tools can be enhanced by using the same technique developed for modeling of computer applications. A more detailed discussion and example are presented in section *4.1.2 Business Model Implementation*.

2.2 Simalytic Modeling Methodology

Simalytic Modeling brings together existing performance models (usually platform-centric analytic models) and application information (best expressed as simulations). The technique allows some parts of the larger model to be replaced with submodels developed from application performance results. The submodels can be implemented with different types of tools, or even different modeling techniques, as long as they provide equivalent functionality.

Using the same approach, the results of an application model (analytic, simulation or Simalytic Model) can be used to represent the performance of the computer environment aspect of the application in a larger model of the business process. A valid model (proven to produce accurate predictions) must exist for each system or application to be included in the business enterprise model. The application details must be understood, and consistently defined at the enterprise level. A business enterprise level model is constructed with a very high level simulation model of the application process, where computer response time of the application is adjusted, based on the business arrival rates, by using a transform function similar to the Simalytic Function™ discussed in (Norton 1997a; Norton 1997b; Norton 1997c). As the simulation model is run, the process time spent in the computer environment is dynamically adjusted depending on the overall activity for the application. Applying the Simalytic Modeling Methodology to a business process models is called a Simalytic Business Model.

The main advantage of the Simalytic Modeling methodology is the ability to combine the results from different modeling techniques and different modeling tools. Using different techniques allows for a better fit between technique and purpose. Using different tools allows results from the tools an organization already uses for modeling individual systems to be used in the overall model. This reduces the time and effort to build an enterprise level model of an application by using the results from commercially available platform-centric tools or existing detailed application models. By combining the results of the application level model with a business simulation tool, the same advantages can be extended to the overall business level.

3. Building a Business Simalytic Model

As with any modeling effort, creating a Simalytic Model requires more than just putting the pieces together in some modeling tool. A substantial amount of information is required about both the application and systems used by the application. This section provides an abbreviated discussion of the steps to build a Simalytic Model of a computer application; a full discussion of is available in the author‘s earlier papers (Norton 1996; Norton 1997a; Norton 1997b; Norton 1997c). The most critical step, Workload Analysis, is a very complex and involved process, and only some of the issues involved, those that relate directly to the construction of a Simalytic Model, are discussed here. Other areas, such as calibration techniques for queuing theory tools and features of simulations tools are assumed to be covered in the training and documentation for the specific tools

The major phases of Simalytic Modeling for a computing environment application are:

1. Workload Analysis
2. Node Models
3. Simulation Model
4. Simalytic Model
5. Model Analysis

Just as system model results are connected using a simulation tool to create a Simalytic Model, the results of a Simalytic Model can be connected using a business modeling tool. Once the computer transactions can be accurately modeled, a model of the business is then constructed. The approach to building a business model is quite different than the approach to building computer system models familiar to capacity planners. There are two major differences. The first is the focus on understanding what happens to the business process over time as external forces exert influence. The second is the concentration on business elements (i.e., inventory and employees) rather than computer elements (i.e., response time and throughput). Once the computer systems and applications have been modeled, those results can be integrated into a business model.

The major phases of Simalytic Business Modeling are added to the Simalytic Modeling phases above:

6. Business Process Analysis
7. Business Model Construction
8. Simalytic Function Integration
9. Simalytic Business Model Analysis

Each of these phases, although similar to the phases of a Simalytic Model, is discussed in the sections related to Simalytic Business Modeling to both emphasize the similarities and explain the differences. This list is not meant to be comprehensive for all phases. Once the Simalytic Business Model has been created, it must be used productively. Generating the speculations, planning the scenarios and analyzing the results in terms of application impact are all activities the reader should be very comfortable with within the context of the modeling tools already being used. The inclusion of business process modeling expands and enhances the capacity planning function by providing the information necessary to understand the causal relationship between the business elements and the computing elements. Simalytic Business Modeling requires the same level of analysis and presentation once the model has been completed and calibrated. A major advantage to the capacity planner is this ability to directly relate computer systems and application changes to the impact on the entire business process. The Simalytic Business Model methodology provides more realistic model results than the traditional business modeling approach of using the best and worst cases and assuming the truth is "somewhere in-between".

3.1 Simalytic Modeling Phases

Each of the phases of Simalytic Modeling is discussed in the following sections, with each phase broken down into the steps to implement it. Under each phase the steps are listed with an identifying phrase followed by the details of the step.

3.1.1 Workload Analysis Phase

In the Workload Analysis Phase the modeler collects information about the application to be modeled. This includes identifying, defining, documenting and measuring the application. This workload analysis is done for all of the systems supporting the application and includes collecting information about the application from the enterprise point-of-view.

Identify: Identify the workload. Identifying the workload to be modeled is often the most difficult step of any modeling activity. Because Simalytic Modeling takes an enterprise view of the application, the identification process seems more difficult because it requires consideration of all systems. This information should be collected for any modeling activity in a client/server environment, but often it is not because the effort focuses on a single system for expediency. Not only does the application need to be identified for each system, but it must also be identified at a global level.

The point of this step is to define business activities, such as orders entered, in terms of measurable work elements, such as transactions A, B and C. The workload projections and response time measurements are at the business level and the models are built at the IT (Information Technology) transaction level. This step must be done in conjunction with both the application developers and end-users.

Document: Document the application topology. What transactions are routed where under what conditions? The documentation technique used should be whatever best supports the application and has the support of the users and developers, who must maintain the documentation.

The objective of this step is to produce a topology description of the application that can be easily and accurately translated into a simulation model. When this step is completed, the modeler should be able to track a business transaction from the originating workstation through the entire environment (including all splits, protocol translations, routing decisions, etc.) back to the same workstation.

Measure: Measure the workload. In this step, the modeler must determine the ability to measure the application and workloads at each system and from the end-user point-of-view (sometimes referred to as end-to-end response time). If adequate measurement data cannot be collected, then the value of continuing must be assessed. Will sufficient interest will be generated by the effort to increase the quality of the measurement data? How well this will be accepted will vary by organization and company.

Correlate: Correlate the workload across systems. The final step of Workload Analysis is to determine the correlation between the workloads at each system. Does the definition for a workload at one system really mean the same thing as that workload at another system? There cannot be any additional or missing transactions. This appears to be a straightforward requirement, but it becomes very complex as the client/server environment grows and applications attempt to reuse functions.

The objective of this step is to insure the consistency of the workloads across the entire enterprise model.

3.1.2 Node Models Phase

In the Node Models Phase, the modeler models all of the systems supporting the application. This phase includes the same type modeling done for system level modeling efforts, but coordinates the node level models to integrate with the additional information about the application from the enterprise point-of-view.

Build: Build a model of each node. Building a model of each node used by the application is not significantly different from any existing system level modeling efforts.

The objective of this step is to build a model of each system, but to also take advantage of any existing modeling efforts. Although the workload definition may change, the processes already in place to collect measurement data and calibrate the models for any of the nodes can be effectively reused.

Calibrate: Calibrate the model of each node. The node level models must be valid (calibrated and verified) before continuing. There must be a high degree of confidence in the predictive nature of each of the node models. Because the Simalytic Model will connect the nodes together using the workload definitions, an error or poor results from any one node model can impact the accuracy of the entire application level Simalytic Model.

The objective of this step is to have a solid predictive model for each node that presents a consistent view of the application across all nodes.

Run: Run the models. Develop a profile of the application by running each of the node models for a series of arrival rates from very low (i.e., .01 or .001) to very high (either the model saturates or the 'knee' of the response time curve is well established). The actual arrival rates used will depend on the application and should make sense to the actual users of the application.

The objective of this step is to establish a response time curve that can be used to extrapolate the response time when presented an arrival rate not modeled.

Create: Create a model results table. Create a table of response times and arrival rates for each system for the workload of interest.

The objective of this step is to characterize the application performance and responsiveness. The information in this table will be used to create the Simalytic Function when the Simalytic Model is constructed.

3.1.3 Simulation Model Phase

In the Simulation Model Phase, the modeler builds an overall model of the application with each of the systems supporting it represented as a node or server. This phase uses the information from the Workload Analysis Phase to connect each of the systems together to provide the enterprise view of the application.

Build: Build an overall model. Using the simulation tool of choice, build an overall model of the application with a single server for each node in the system. This model is defined by the application topology documented in the Workload Analysis Phase. It identifies what transactions are routed to which nodes under what circumstances.

The objective of this step is to build a model of the application that represents the overall application behavior across the enterprise.

Set: Set the overall model parameters. Set the service time of each node to the lowest response time in the table created above. Set each node to have enough servers so that there is ***no*** queuing at any node.

The objective of this step is to set the simulation model such that the response time (service time plus queue time) at any node can be controlled by the Simalytic Function when it replaces the static service time in a later phase. In addition, the simulation model at this stage can be used to verify the application topology and conduct a sensitivity analysis of user expectations.

Calibrate: Calibrate the overall model. Calibrate the simulation model against the end-user response time for the very low arrival rate and verify that there is no queue time at any of the nodes. Because the response time from the queuing theory tool includes the queue time, any queue time in the simulation model will, in effect, double count the queue time. The simulation tool is being used to control the flow and routing of transactions, not calculate the queue time. This step should insure that the topology and routing information is correct.

The objective of this step is to verify that the simulation model accurately reflects both the topology of the application and the response time seen by the users at very low arrival rates.

3.1.4 Simalytic Model Phase

In the Simalytic Model Phase, the modeler incorporates the results of the system models into the overall model of the application. This phase uses the information from the Workload Analysis Phase and the Node Models Phase to provide the predictive capabilities to the enterprise view of the application.

Create: Create the Simalytic Function. Using the table of response times and arrival rates created from the node models, create a Simalytic Function for each node. This can be any of a number of techniques, such as, a look-up table or a formula derived from the curve established by fitting a line to the response time data. The details of the function and how it is implemented will depend on the simulation modeling tool used for the overall model framework.

The objective of this step is to create a function for each node that accurately reflects the application's behavior. Each Simalytic Function must return a value for the node service time for each visit of a transaction consistent with application performance at the current system load.

Replace: Replace the static service times. Replace the service time for each node with the function created in the prior step. Again, how this is done will differ with each simulation tool.

The objective of this step is to implement the Simalytic Function in each node of the overall simulation model such that the service time used for each transaction visit is the value returned by the Simalytic Function.

Calibrate: Calibrate the Simalytic Model. First calibrate it against the prior simulation model for the very low arrival rate to insure the overall model structure is still correct. Next, calibrate it against known end-user response times for known arrival rates.

The objective of this step is to insure that the Simalytic Model provides the required level of application prediction within the required level of accuracy.

3.1.5 Model Analysis

The next phase uses the Simalytic Model to analyze the application. At this point, the Simalytic Model can be used just as any other type of model which has been calibrated. How the model is used to answer "what-if" questions is very dependent on the questions themselves. Therefore, the details of this phase will not be discussed here. All of the phases of constructing a Simalytic Model should be considered as a spiral development process. The completion of each phase may identify additional information or requirements for the prior phase. This provides the added benefit of allowing the modeler to implement a quick, simple Simalytic Model and then continue to refine it based on the business requirements and objectives. The same philosophy is maintained as Simalytic Modeling is extended to Simalytic Business Modeling.

3.2 Simalytic Business Model Phases

Each of the phases of Simalytic Business Modeling is discussed in the following sections in a similar manner to section *3.1 Simalytic Modeling Phases*, with each phase broken down into the steps to implement it. Under each phase the steps are listed with an identifying phrase followed by the details of the step. There is a great deal of similarity between sections *3.1* and *3.2* because the same concept is used. The major difference between these sections is that *3.1* discusses connecting system models to an application model while *3.2* discusses connecting application models to a business model.

The Simalytic Business Model Phases are the steps to implement a full business model using the Simalytic Modeling Technique. They are shown here as following the Simalytic Model Phases only because the author developed Simalytic Modeling and then realized it could be applied to business modeling. It is very reasonable to develop the business model first and use it to explore the usefulness of building models of the computer systems and applications. These models would then be built only if the business model shows that the business process is sensitive to realistic changes in the computing environment, such as response time reductions or through-put increases. For example, such computing environment changes would not be necessary if a business model showed that reducing the response time of an application to 10% of its current value does not change the staffing requirements or increase the number of orders.

The Simalytic Business Model Phases are implemented with modeling tools developed for Systems Dynamics, a complex discipline focused on the in-depth understanding of modeling business processes. Because Systems Dynamics requires many years of study for a basic understanding (Richmond 1994) this paper does not attempt to provide a complete and comprehensive tutorial of the subject. The rest of this section tries to strike a balance between complete understanding and only enough information for the following example. The interested reader is encouraged to look at the references and download one of the demo business process modeling tools available. The Internet URL's are included in the references for three such tools (HPS; Powersim; Vensim).

3.2.1 Business Process Analysis Phase

In the Business Process Analysis Phase the modeler collects information about the business process to be modeled. This includes identifying, defining, documenting and measuring the business function. This analysis is done for all aspects of the business process and includes collecting information about the business process from all points-of-view.

Identify: Identify the business process. Identifying the business process to be modeled is often the most difficult step of the modeling activity. Not only does the business process need to be identified, but also the criteria to determine the success or failure of that process must be identified at a global level.

The point of this step is to understand the business activities, such as orders entered, in terms of measurable business elements, such as inventory and staffing level. The projections and measurements are at the business level and the models are built to describe how the processes function. This step must be done in conjunction with both the business process owners and the end-users.

Document: Document the business process topology. What are the business elements and how do they flow under what conditions? The documentation technique used should be whatever best supports the business process and has the support of the users and the process owners, who must maintain the documentation.

The objective of this step is to produce a topology description of the business process that can be easily and accurately translated into a simulation model. An advantage to using one of the business modeling tools mentioned in this section is that they are often used for such documentation tasks and therefore provide a technique the business process owner already finds acceptable. When this step is completed, the modeler should be able to account for all aspects of the business process. If one part of the process is not well understood it can be implemented in a simple form and expanded later if the model is sensitive to that area.

Measure: Measure the business process. In this step, the modeler must determine the ability to measure the business process (both elements and flows) at each step in the process from the point-of-view of both the process owner and the end-user .

3.2.2 Business Model Construction Phase

In the Business Model Construction Phase the modeler builds an overall model of the business process with constants to represent each of the transaction response times from computing environment parts of the process. This phase uses the information from the Business Process Analysis Phase to connect the computing environment to the business process.

Build: Build an overall business process model. Using the business process simulation tool of choice, build an overall model of the business process. This model is defined by the business process topology documented in the Business Process Analysis Phase. It is not fundamentally different than a stand-alone business process model, but it is constructed such that the variables required for the computing environment aspects of the model will be available.

The objective of this step is to build a model of the business process that represents the overall behavior and allows the computing environment areas to be expanded.

Calibrate: Calibrate the overall business process model. Calibrate the simulation model against the business process element metrics identified in the Business Process Analysis Phase. The simulation tool is being used to control the flow and routing of business elements, not calculate response times or queue times. This step should insure that the topology and routing information are correct.

The objective of this step is to verify that the business process simulation model accurately reflects both the topology of the business process and how the business process reacts to external forces.

3.2.3 Simalytic Function Integration Phase

Integrating the Simalytic Function into the business process model is much more straight-forward than with a computer environment simulation model. The model constant representing the transaction response time is replaced with an arrival rate calculation based the number of computer transactions per business element. That arrival rate is then used with the Simalytic Function to determine the response time at that arrival rate, which then reduces the flow of business elements somewhere in the model. Because the business model is constructed to analyze the business elements over time, it is relatively easy to calculate the average arrival rate over each model step.

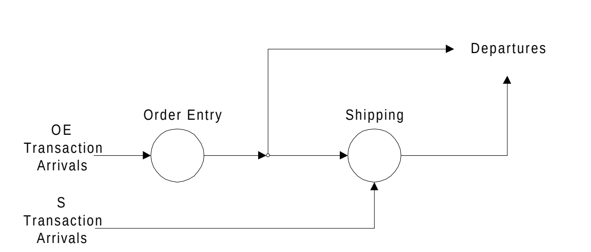


Figure 2 Computer Transaction Flow

3.2.4 Simalytic Business Model Analysis Phase

The last phase uses the Simalytic Business Model to analyze the entire business process. At this point, the Simalytic Business Model can be used just as any other type of business process model which has been calibrated. How a model is used to answer “what-if” questions is very dependent on the questions themselves. Therefore, the details of the phase will not be discussed here other than to note that all of the phases of constructing a Simalytic Business Model should also be considered as a spiral development process. The completion of each phase may identify additional information or requirements for the prior phase.

4. Simalytic Model Implementation

Simalytic Modeling effort involves creating and calibrating multiple models. In order to focus on the model building process, a simplistic example is used to illustrate the steps presented in *Section 3 Building a Business Simalytic Model*. The details of constructing the Simalytic Model for this example, and the results of that model, are presented in full in (Norton 1997a) and will not be repeated here other than the final results shown in *Table 1 Application Simalytic Model Results*. The following section extends that example to include the business model and shows how the Simalytic Modeling Technique can be applied.

4.1 Implementation Example

This implementation of a Simalytic Business Model uses a hypothetical order entry function and a client/server computing environment to illustrate the process. Assume the business process of interest is the Order Entry function and the computer workload of

OE Arrival Rate	OE Response Time	S Response Time
0.01	0.661	2.080
0.10	0.716	2.123
0.20	0.728	2.126
0.50	0.754	2.166
1.00	0.780	2.292
2.00	0.907	2.696
3.33	1.622	5.085
3.70	2.036	6.468
4.00	2.838	9.062
5.00	11.492	38.144

Table 1 Application Simalytic Model Results

interest is the Order Entry application. A Shipping application on another server also used by the Order Entry application. The Order Entry user types in the name of an existing customer and gets not only their address, but information about any orders. This may provide better service, but it also causes transactions to be sent the other system. Defining the topology of the application identifies that some number of the Order Entry transactions are routed to the Shipping server. The measurement data provides the number and distribution of transactions. If the Shipping workload outgrows that system, it can impact the responsiveness of the Order Entry transactions. In addition, growth in the Order Entry workload will now impact the Shipping system, but only if the orders are from existing customers. The systems cannot be modeled independently because the service time for one system is dependent on the response time of the other. When the Order Entry transaction rate increases, more transactions are sent to the Shipping server. The increased response time at Shipping will cause the overall response time for those transactions to increase, which will be seen as longer operator call time. Therefore, there is a direct relationship between the responsiveness of the Order Entry application and the number of Order Entry operators required to handle the incoming calls.

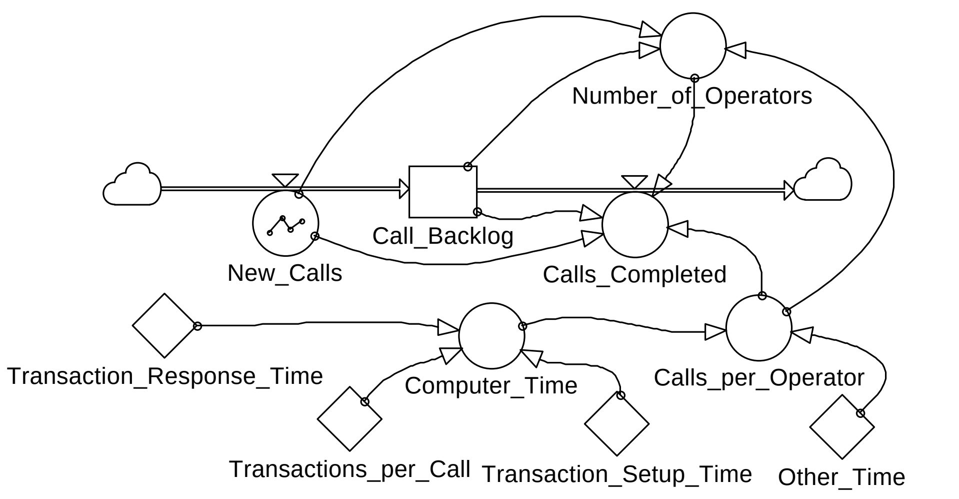


Figure 3 Basic Business Model of Order Entry

4.1.1 Computer Model Implementation

Figure 2 Computer Transaction Flow shows a diagram of the computer system. The response time is measured from Arrivals to Departures, either through the Shipping node or around it. The Simalytic Model connects what is happening in the application on the different servers. If the Order Entry system is modeled by itself, the workload representing the long transactions (those also sent to Shipping) would not reflect the increased response time due to the load at Shipping. Because of the additional application information in the Simalytic Model, it can adjust the response time in the Shipping server based on the current load, which will then be reflected in the Order Entry transactions that visit the Shipping server.

4.1.2 Business Model Implementation

The Simalytic Business Model connects the application to the business processes by making the transaction response time a dynamic, rather than static, component of the model. The business process flow is illustrated in *Figure 3 Basic Business Model of Order Entry* using the Powersim modeling tool (Powersim). This model simulates the order entry process over one day (24 hours) with each simulation step being one hour. The metric that is used to determine the effectiveness of the process is the number of operators required for each hour to answer the incoming calls. The modeling tool provides icons to represent different business functions just as computer modeling tools provide icons for computer elements. The clouds represent the model boundaries and show business elements entering and leaving the model. The double line arrows show how the business elements flow through the model. The rectangle labeled **CALL_BACKLOG** represents the current level, or number, of backlogged calls. The circle with a small triangle on the double line is a flow and represents something that influences the business elements along that path. The **NEW_CALLS** flow icon has a graph symbol to show that the number of calls arriving for each hour is determined by graph function, which, in this case, is set to a low value overnight with both a morning and an afternoon peak. The other circles represent calculated values and the diamonds represent constant values. The single lined arrows point from a value used in a calculation to the icon where it is used (they do *not* show business element flow).

In this example, the flow **CALLS_COMPLETED** is calculated by the multiplying the number of calls an operator can answer in an hour times the number of operators available (**CALLS_PER_OPERATOR * NUMBER_OF_OPERATORS**). **CALLS_PER_OPERATOR** is the total time an operator spends per call (**COMPUTER_TIME** + **OTHER_TIME**) in minutes divided into the simulation step (one hour). The current number of operators (**NUMBER_OF_OPERATORS**) equals the maximum of either 20 operators or the number to answer the new calls plus the backlog of calls (**MAX (20, (CALL_BACKLOG / CALLS_PER_OPERATOR) + (NEW_CALLS / CALLS_PER_OPERATOR))**. All of the variable calculations are shown in

Figure 6 Business Model Icon Definitions. This model also has an input control not shown that allows the model user to vary the **TRANSACTION_RESPONSE_TIME** value for different simulation runs (this is the value that is replaced with a dynamic calculation in the Simalytic Business Model shown in *Figure 4*).

4.1.3 Implementation Objectives

As with any modeling effort, there must be business objectives to analyze using the model. Assume that the manager of the Order Entry department has requested a model to determine how many operators are required over the day so he can schedule the required number of operators for each hour. He would also like to know if upgrading the server to improve the current Order Entry system response time objective of less than 1.7 seconds per transaction will reduce the required number of operators (because they can each handle more calls).

4.2 Simalytic Business Model Implementation

Implementing the Simalytic Business Model begins with the basic business model of *Figure 3* and the results of the Simalytic Model of the application from (Norton 1997a). *Table 1 Application Simalytic Model Results* shows the results of that application model. The **OE ARRIVAL RATE** column shows the number of Order Entry transactions per second. The **OE RESPONSE TIME** and **S RESPONSE TIME** columns show the corresponding transaction response times for the Order Entry application and the Shipping application, respectively. (The Shipping results are not used here and are included only for consistency with the prior papers.) These are the results that form the Simalytic Function in the **TRANSACTION_RESPONSE_TIME** icon of the Simalytic version of the model, shown in *Figure 4 Simalytic Business Model of Order Entry*. The only difference between the basic model shown in *Figure 3* and the Simalytic version shown in *Figure 4* is that the **TRANSACTION_RESPONSE_TIME** icon is calculated instead of being a constant. It uses the constant **TRANSACTIONS_PER_CALL** and the flow **NEW_CALLS** to calculate the arrival rate, which is then used (via the information from *Table 1*) to determine the response time at that arrival rate.

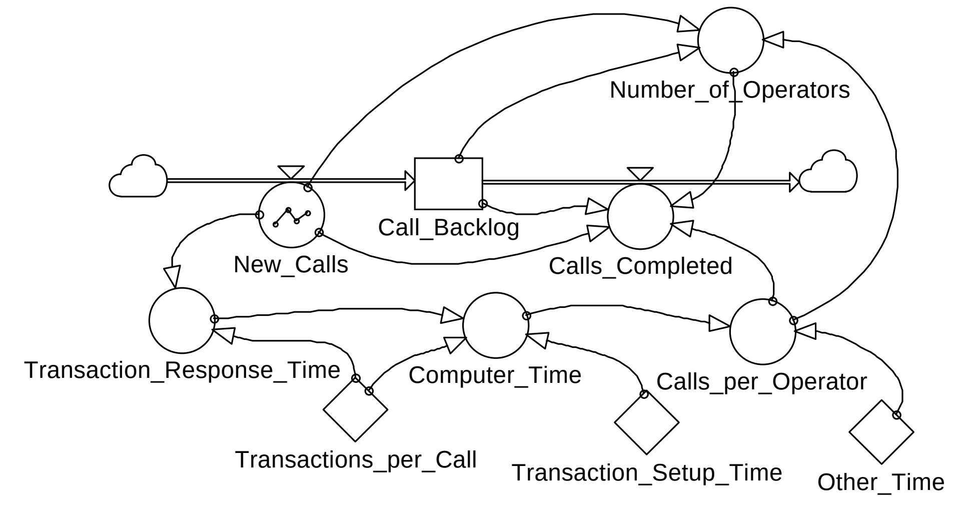


Figure 4 Simalytic Business Model of Order Entry

4.3 And The Answer Is…

How do the these results relate to the questions asked in section *4.1.3*? *Figure 5 Operator Comparison* shows the answer. The basic model uses static response time values with best case and worst case scenarios while the Simalytic Business Model used a dynamic response time value and requires only a single scenario.

The basic model best case scenario uses the minimum response time and the response time goal from the application service level agreement, but under-predict the required operators during the peak hours. The worst case scenario uses the maximum response time to insure enough operators for the peaks, but over-predicts the required number for most of the day. Neither model provides any information about when the application no longer meets the response time objective or the impact of that failure on the business. *Figure 5* shows that the hours when the computing environment does not meet the application's response time objective cause a significant increase in the number of required operators.

Because the Simalytic Business Model dynamically calculates the response time value, it is easy to see in *Figure 5* that the computer system meets the service objective most of the day and only the minimum number of operators is required. The impact of the few hours of poor performance is seen as the number of operators required ramps to the maximum number shown by the worst case scenario. Now it is a business analysis situation to determine if the cost of upgrading the application servers is off-set by the savings realized by reducing the number of operators required for those few hours. In addition, further study with this model shows additional savings in operator staffing for the non-peak hours if the response time is also improved. If a server upgrade is modeled, the impact of any changes in application response time can be seen as reduced staffing requirements; not only for the troublesome peak hours, but also for the rest of the day.

Operator scheduling is an existing and ongoing function of the business manager. He must understand its importance to the business and he must see the direct connection to the profitability of the business. The model allows him to remain focused on that aspect of his function without being distracted by the complexities of the computing environment technology that he has little or no interest in understanding. For example, assume an hourly pay rate for the operators of $20/hour, which includes benefits and overhead. An extra 25 to 50 operators for eight hours over 365 days means increased expenses of $1.5M to $3M a year, which might, or might not, offset the cost of a server upgrade. The result of any server upgrade becomes another variable, along with the available work-force, legal and union regulations, overtime analysis, etc., that he can use to determine the best operator schedule to meet the business objectives. In addition, the justification for a server upgrade is much easier when the impact on the entire business process is presented.

5. Conclusion

The traditional view of planning the capacity of a system is evolving because of the desire to predict the performance of the application, but if the impact of the application on the business is not known, that predictive ability may be of little value. Applications designed to exploit a client/server architecture greatly increase the complexity of both the computer system configurations and the applications themselves. Predicting the responsiveness of those more complex applications requires a more complex modeling methodology to address the interrelationships between systems. But adding complexity to a modeling effort also adds time, effort, and cost. There are many techniques and tools that are beginning to address this evolution, but none of them can provide the desired level of detail for every situation and every application.

By following these steps for implementing a Simalytic Model at the business level, the modeler can rapidly produce a high level application model. The level of detail can be refined as necessary to make the required business decisions. Combining different modeling techniques (simulation or analytic queuing theory) and different modeling tools (platform-centric, general

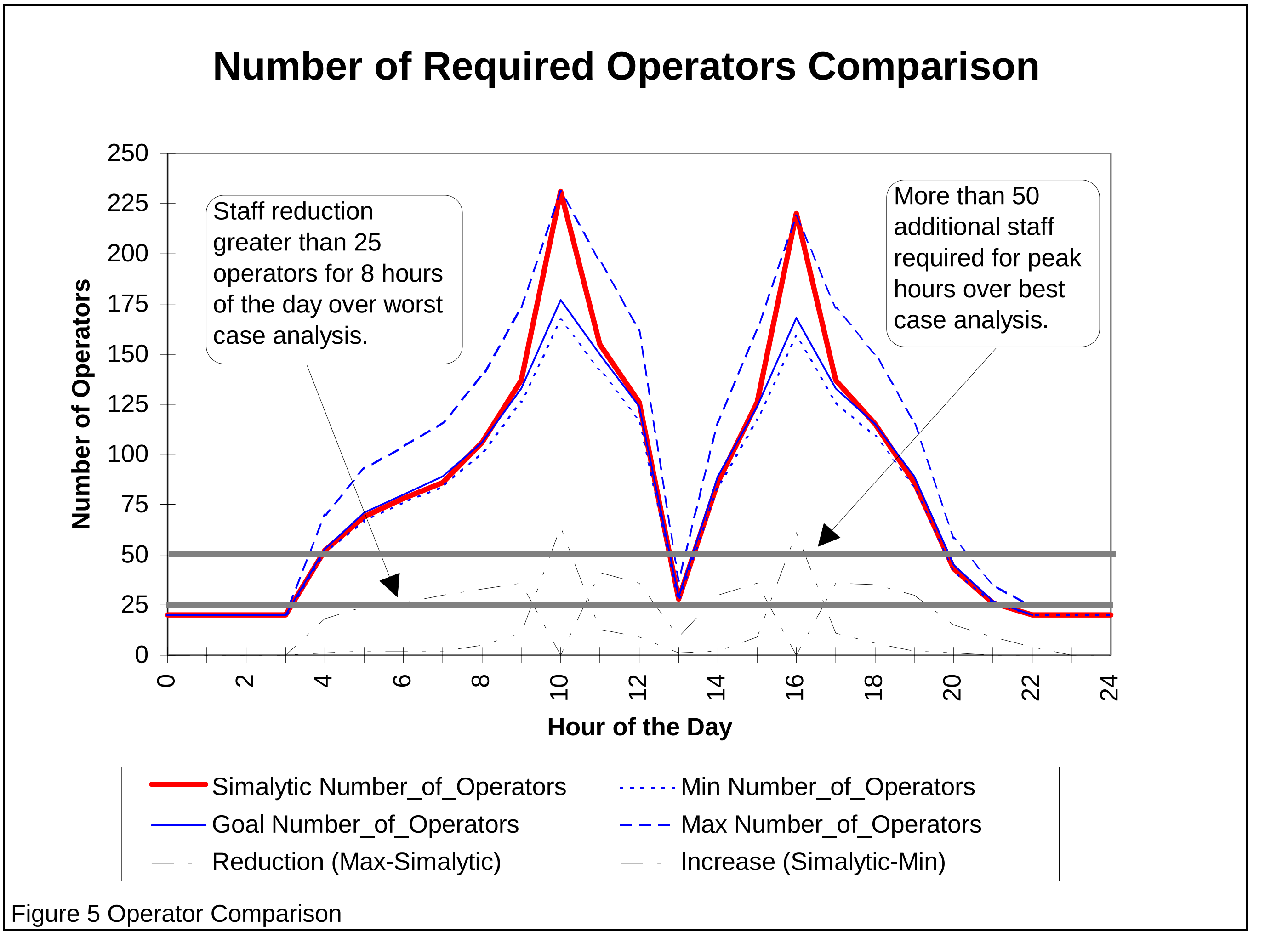


Figure 5 Operator Comparison

purpose or business) reduces the time, effort and cost of developing enterprise models of either the application or the business process. As more detailed results are required, more sophisticated tools can then be used to increase the understanding of critical sections of the model.

This level of analysis provides insight into the application's future performance, and its impact on the overall business, that would not otherwise be available. Using the Simalytic Modeling Technique improves the understanding of the application, identifies which systems require more detailed analysis and quantifies the relationship between the application computing environment and the overall business. It protects the investment an organization has made in acquisition and training of existing tools while allowing the most appropriate tools to be used for each modeling effort.

Capacity planning is even more fundamental to business success today than it has been in the past. Just as application designs have moved away from single system solutions, modeling for capacity planning has moved away from single system analysis to predicting the application across the enterprise. But predicting the application is not enough. The relationship between the application's performance and the business process must be fully explored to understand the true impact of capacity planning decisions. Models used to analyze business processes must fully integrate the behavior of the computing environment into the model. Capacity planning has had significant impact on the overall business by controlling computer environment costs and improving application performance. Simalytic Business Modeling increases that impact by including the business process and allowing the benefits of capacity planning to be realized across all aspects of the business. This understanding provides the necessary information to allow us to truly model the business.

OE-BASE.SIM - Basic Powersim model of Order Entry (Figure 3)

```
init  Call_Backlog              = 0
flow  Call_Backlog              = -dt*Calls_Completed +dt*New_Calls
aux   Calls_Completed           = MIN(MAX(New_Calls,Call_Backlog),
                                     (Number_of_Operators*Calls_per_Operator))
aux   New_Calls                 = GRAPH(TIME,0,1,{0,20,30,100,600,800,900,1000,1200,
                                     1500,2000,1700,1400,320,
                                     1000,1400,1900,1500,1300,1000,500,300,200,100,50,30,30,30,0,0,0,
                                     0,0,0,0,0 "Min:0;Max:4200"})
aux   Calls_per_Operator        = 60/(Computer_Time+Other_Time)
aux   Computer_Time             = ((Transaction_Response_Time/60)
                                     +Transaction_Setup_Time)*Transactions_per_Call
aux   Number_of_Operators       = MAX(20,(CEIL(Call_Backlog/Calls_per_Operator)
                                     +CEIL(New_Calls/Calls_per_Operator)))
const Other_Time                =  3.0
const Transaction_Response_Time   =  1.7
const Transaction_Setup_Time =  0.2
const Transactions_per_Call   = 10.0
```

OE-SF.SIM - Powersim model using Simalytic Modeling Technique (Figure 4)

```
init  Call_Backlog              = 0
flow  Call_Backlog              = -dt*Calls_Completed +dt*New_Calls
aux   Calls_Completed           = MIN(MAX(New_Calls,Call_Backlog),
                                     (Number_of_Operators*Calls_per_Operator))
aux   New_Calls                 =
                                     GRAPH(TIME,0,1,{0,20,30,100,600,800,900,1000,1200,1500,2000,1700,
                                     1400,320,
                                     1000,1400,1900,1500,1300,1000,500,300,200,100,50,30,30,30,0,0,0,
                                     0,0,0,0,0 "Min:0;Max:4200"})
aux   Calls_per_Operator        = 60/(Computer_Time+Other_Time)
aux   Computer_Time             = ((Transaction_Response_Time/60)
                                     +Transaction_Setup_Time)*Transactions_per_Call
aux   Number_of_Operators       = MAX(20,(CEIL(Call_Backlog/Calls_per_Operator)
                                     +CEIL(New_Calls/Calls_per_Operator)))
aux   Transaction_Response_Time   = IF((New_Calls/3600*Transactions_per_Call)<0.1, 0.6,
                                     IF((New_Calls/3600*Transactions_per_Call)<1.0, 0.75,
                                     IF((New_Calls/3600*Transactions_per_Call)<2.0, 0.8,
                                     IF((New_Calls/3600*Transactions_per_Call)<3.3, 0.9,
                                     IF((New_Calls/3600*Transactions_per_Call)<3.7, 1.6,
                                     IF((New_Calls/3600*Transactions_per_Call)<4.0, 2.2,
                                     IF((New_Calls/3600*Transactions_per_Call)<5.0, 2.8,11.5)))))))
const Other_Time                =  3.0
const Transaction_Setup_Time =  0.2
const Transactions_per_Call   = 10.0
                                        NOTE: Statements that differ between models are underlined.
```

Figure 6 Business Model Icon Definitions

6. Acknowledgments

The author would like to thank the referees and editor for their careful reading and helpful comments. A special thanks is expressed to Mr. Rick Lebsack and Mr. Brian C. Tinsley for their interest and in-depth critiques of the early versions.

7. References

BGS. Best/1 and Crystal . BGS Systems, Inc., 128 Technology Center, Waltham, MA 02254.

Cover, Jennifer M. 1996. Introduction to System Dynamics. Reston, VA: Powersim Corporation.

Gunther, Neil J. 1995. Performance Analysis and Capacity Planning for Datacenter Parallelism. In Proceedings. Computer Measurement Group: CMG, Inc.

Hatheson, Amanda. 1995. Two Unix Client/Server Capacity Planning Case Studies. In British Proceedings. Computer Measurement Group: CMG, Inc.

HPS. STELLA and ithink http://www.hps-inc.com. High Performance Systems, Inc., 45 Lyme Road, Suite 200, Hanover, NH 03755.

Norton, Tim R. 1996. Simalytic Enterprise Modeling: The Best of Both Worlds. In Proceedings. Computer Measurement Group,: 1-12. San Diego, CA: CMG, Inc.

Norton, Tim R. 1997a. Don’t Plan Capacity When You Should Predict Applications. In Proceedings. Computer Measurement Group,: 1-12. Orlando, FL: CMG, Inc.

Norton, Tim R. 1997b. Simalytic Hybrid Modeling: Planning the Capacity of Client/Server Applications. In Proceedings Volume 6. 15th IMACS World Congress,: 583-588. Berlin, Germany: International Association for Mathematics and Computers in Simulation.

Norton, Tim R. 1997c. Simalytic Modeling: A Hybrid Technique for Client/Server Capacity Planning. In Proceedings. Summer Computer Simulation Conference,: 172-177. Arlington, Virginia: Society for Computer Simulation.

Pooley, Rob. 1995. Performance Analysis Tools in Europe. Informationstechnik und Technische Informatik 37 : 10-16.

Powersim. Powersim http://www.powersim.com. Powersim Corporation, 1175 Herndon Parkway, Suite 600, Herndon, VA 20170.

Richmond, Barry. 1994. System Dynamics/Systems Thinking: Let's Just Get On With It. International Systems Dynamics Conference. Sterling, Scotland.

SES. SES/workbench and SES/strategizer . Scientific and Engineering Software, Inc, 4301 West Bank Dr. Building A, Austin, TX 78746.

Smith, Connie U. 1995. The Evolution of Performance Analysis Tools. Informationstechnik und Technische Informatik 37 : 17-20.

Vensim. Vensim http://www.std.com/vensim/. Ventana Systems, Inc., 149 Waverley Street, Belmont MA 02178.