Simulation versus Analytic Modeling in Large Computing Environments

Predicting the Performance Impact of Tuning Changes

A White Paper from the Responsive Systems Company

Prepared and Written by Dr. Bernie Domanski
Independent Industry Analyst

Abstract - With complex, heterogeneous computing architectures drawing much attention of late, capacity managers are faced with interesting new challenges. What methodology(ies) should be used to for capacity planning and tuning of large database applications? In this paper, our focus is on the strengths and weaknesses of analytic modeling versus simulation particularly for these environments.

1. Model Types and the Modeling Process

To help illustrate some of the basic concepts behind modeling, we'll consider the problem of evaluating workloads over time. If we look at historical performance data, and perform some statistical forecasting (i.e. linear regression), then we can define a *capacity exhaust volume:* the point at which our existing system no longer provides acceptable service. Here, throughput has peaked at a specific level and/or response time is too high and is no longer acceptable. When workload volumes approach capacity exhaust, we need to evaluate *alternative physical configurations,* e.g.

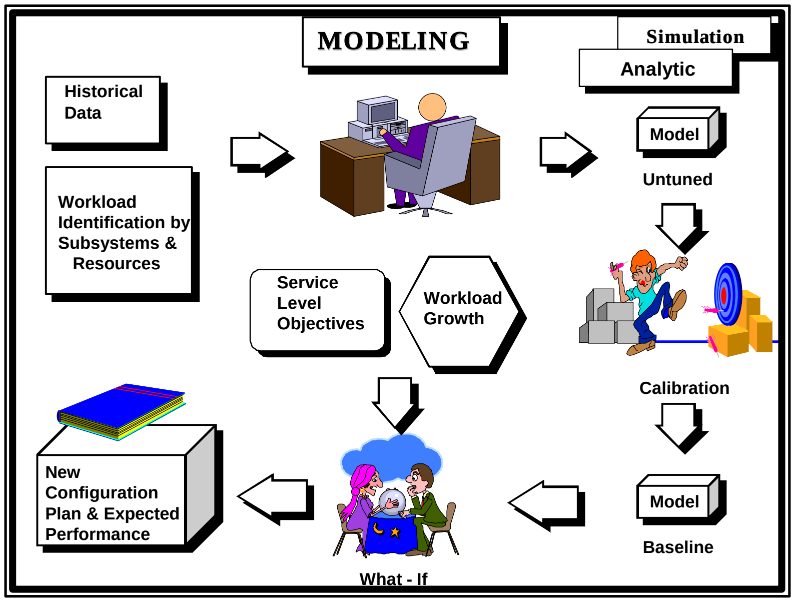
- add memory to a server, or
- add buffers where they are needed while reducing the buffer count in other areas, or
- alter the number or type of workloads presented to a system, or
- buy a whole new machine.

These *what-if* alternatives are best examined using either *analytic modeling* or *simulation modeling*.

Analytic models (also referred to as *queuing models*) represent the activity in a queuing system at a particular moment in time; almost like a snapshot. Mainframe systems have been typified for nearly 25 years by devices with inherent *queues,* like a CPU or a disk. Modelers concern themselves here primarily with how long a request for service lives in the queue, and how fast the server delivers service to a request. Queuing relationships are represented by a set of simple algebraic equations for computer systems. And because they are simple equations, they are solved rapidly on a computer. The set of relationships represented by these equations - the methodology - is called *operational analysis* and was developed by Jeff Buzen (the *B* of BGS Systems) and Peter Denning (who created the concepts of paging and swapping in operating systems). Operational analysis was extended to encompass devices that don't have service times, e.g. memory, but the equations directly represent how workloads operate in a system of queues. But the equations represent that *snapshot* of activity - no detail as to what happens moment-to-moment is considered. Thus, when you execute a queuing model, questions that drive the model usually involve workload volumes - e.g. what *will response time be if I run 10,000 transactions an hour?* These volumes are usually for workload averages. In addition, and this is key, is that a "*transaction*" is the average transaction. The resources used by what the "average" transaction is defined to be are what drive the queuing model. Thus, any "batch"-type transactions (long running or resource intensive transactions) are not really considered if they are not "average". For the question we've posed, 10,000 is the average number of "average" transactions arriving during a peak hour. This implies that the actual distribution of workload volume is ignored - at times within the hour it is higher than the average, and sometimes it's lower. Thus, queuing models ignore the *structure* of workloads, and therefore cannot answer all questions regarding the possible *behavior* of a system. Analytic models sacrifice accuracy, but gain in execution speed.

Simulation models do take workload distributions into account; in fact, they consider each *event* (or state transition) a transaction that goes through as it visits different servers in a complex computing environment. But simulation models historically take longer to build (because you have to understand the workflow behind each application), and take longer to execute (because there are so many events to be simulated). Unfortunately, simulation was given a "*bad wrap*" early on. In 1978, Kobayashi[1] stated the then perception of simulation - "*... It is quite often found, however, that a simulation model takes much longer to construct, requires much more computer time to execute, and yet provides much less information than the model writer expected.*" As time moved forward, and more models were built, it was found that simulation would be used where no analytic solution existed. Within a database-driven application, only simulation can deal with multiple transaction types effectively. How can we size a buffer pool for a single "homogenized" average transaction? Simulation models began to be constructed that were extremely accurate - general-purpose simulation tools evolved positively. Simulation now nearly always provides more detailed results than do analytic models when built correctly. Prediction with simulation-based model area has the ability for more complex predictions than just what would happen when increasing a workload. In the case of DB2, only simulation-based product like Responsive Systems' ***Buffer Pool Tool*** can predict what would happen if we move some objects from one buffer pool into another, or into a new pool that does not exist yet.

Figure 1 - *The Modeling Process*



No matter what type of model is used, we still must go through a *calibration* step; that is, we must verify that the model can reproduce reality before it is used to predict the future (see Figure 1 above). Note, calibration is required for analytic models; simulation models, once calibrated, can be used for prediction immediately. Usually, we run the model using the current physical configuration and the current workload. If the model produces results that are *close* (within 15%-25% for response time; 5%-10% for utilizations) to what the actual system characteristics are, we consider the model calibrated. *Black magic* is usually practiced when the results are not (a common practice for analytic models). Calibration techniques for analytic models are well covered in the paper by Carroll[2] and the interested reader is again strongly encouraged to examine that work.

[1] Kobayashi, *Modeling and Analysis*, Addision-Wesley, 1978.
[2] Carroll, J., *Calibrating a Baseline Model*, CMG 93 Proceedings, page 936.

The calibrated model is often called the baseline, and it is used for *what-if* questions of interest:

- what if I add more buffers,
- what if I move some objects from one buffer pool into another,
- what if I create a new buffer pool and move some objects there,
- what if I add more memory to the processor,
- which objects are monopolizing current resources,
- which objects will gain the most from increased resources,
- which objects will not gain from increased resources.

The results generated often are used to formulate a plan for a new configuration. Analytic as well as simulation models have traditionally ignored cost, as no modeling package has ever adequately factored cost against capacity[3]. However, in the case of the ***Buffer Pool Tool***, a companion white paper "*The Value of Measurement, Analysis, and Modeling*" discusses the value gained from using the product in both quality as well as to the long term budget.

2. ***How Has Performance Been Modeled?***

There have been very few commercially available modeling packages that allowed modeling of a complex database-driven computing environment. Thus, a variety of techniques have thus far emerged. At first, the traditional analytic techniques were used primarily to examine database workloads at a high level. The objective was to see how more work would effect the service delivered. But applications that deployed a large DBMS like IBM's DB2 were modeled at the system level, and provided little to no detail of what the underlying DBMS was doing to response time. While this provides some adequate results for those capacity planners working on the acquisition of new mainframe hardware, these results are suspect for use by a performance analyst because all of the internal events happening in the DBMS were virtually *ignored*. Without recognizing **all** the DBMS resources contributing to the service achieved, this very traditional mainframe approach had limited accuracy. In the late 1970's, IBM recognized this modeling shortcoming for IMS, and created a simulation tool - *SNAP/SHOT* - that would address the contribution of the DBMS (IMS in this case) to the service levels the application would achieve. Users, unfortunately, could not use the tool themselves; they had to schedule time at an IBM facility and work hand-in-hand with IBM system engineers to build and run the model.

By using discrete-event simulation, we can define, in general, each critical component of the DBMS, and follow each object as it visits each of these components within that environment. This completely captures the object's use of DBMS resources, but runs the risk of generating a long-running simulation model. Fortunately, commercial simulation-based tools are emerging in this area that execute rapidly on workstations.

3. Limitations and Efficiency

Let’s take a moment to examine analytic techniques versus simulation techniques in today’s environments with emerging technologies. As we stated earlier, analytic or queuing models are characterized by equations to express steady-state conditions giving results of limited accuracy. In the real world of enterprise computing, systems don’t often experience steady state conditions. Questions are usually directed at peak conditions - sometimes called conditions at the “edge-of-the-envelope” - and are usually beyond the capabilities of the queuing model. The queuing model assumes

[3] One might recall the product ISS3 once marketed by Computer Associates. Through the use of an expert system, ISS3 tried to provide recommendations that weighed cost vs performance. Its primary problem was keeping the cost figures of equipment current.

that expected service times are exponentially distributed, and that arrival times are Poisson distributed. This means that
the expected service times and arrival times for service are independent of all previous events. These types of queuing systems are often called memory less because each event has nothing to do with events that have previously occurred. Consider an I/O in mainframe system: it would be great if the next I/O would be automatically routed to the disk that is the least busy. But that dependency on previous events just does not occur. Salsburg[4] and McNutt[5] have both shown that the arrival process in an I/O subsystem is anything but Poisson. And Serrazi[6] discussed the amount of error inherent in assuming a fixed, non-changing population in a closed queuing system. Nevertheless, queuing models are solved quickly compared to other methods, and historically easy to construct.

Executing a given list of events one at a time, thus recreating the actual system interactions solves simulation models. Thus, they can provide much more accurate results. Historically, they are more time consuming to construct because the set of events must be built based on a thorough understanding of the underlying application. While solving an analytic model is equivalent to solving a set of equations, solving a simulation model requires executing each event which can require more time for execution.

So which method should we choose to model a complex database-driven environment? The real truth tells us that queuing models provide average & standard deviation types of statistics. From our knowledge of I/O systems, we know that the arrival process for I/O is not exactly distributed as a Poisson distribution - hence a built-in inaccuracy. In addition, we know that service times for I/O are not quite distributed exponentially. Again, this leaves room for additional inaccuracy. Analytic modeling packages have developed proprietary heuristics to help compensate for these inaccuracies. Finally, analytic algorithms get more involved and complex as the model grows more elaborate.

Thus, if we're trying to see when analytic models break down, we find it occurs when the capacity planner is trying to evaluate non-average conditions, e.g. conditions "at-the-edge". For example,

- How long will it take to process a batch-type workload while the queues are full?

- How long will it take to get a backup done?

What confounds the analytic situation even more is factoring in any activities that look like a network. Packets move at different speeds, and often using different protocols, making the analytic modeling of pipelining impossible. Piggybacking is the process of putting packets onto other packets - saving some setup overhead - that just happen to be headed for the same destination. Sometimes piggybacking occurs, and sometimes it doesn't. In addition, if you know of a situation where you have non-exponential patterns of service time, you know your results will be suspect. For example, consider *network access* - the time it takes to access the network depends on the load that is present, which is anything but memory-less! Finally, if you want to model a situation where you know you have non-exponential arrivals, e.g. a print server comes on every 5 minutes, this too can lead to less than adequate results with an analytic model.

[4] Salsburg, M. A., *Disk Cache Performance Modeling*, CMG87 Proceedings, p. 423-431.
[5] McNutt, B., *Large Capacity DASD: Is Performance Something to be Afraid Of?*, CMG87 Proceedings, p. 399-404.
[6] Serrazi, *Workload Characterization of Computer Systems and Computer Networks*, North-Holland, p. 159-176, 1986.

Simulation seems to be the obvious analysis choice ... but it's not an easy choice to make. On the positive side, simulation will be able to provide not only average response times, but also maximum values. A complete analysis of service time at varying levels of resource availability (in the case of Buffer Pool Tool, the number of buffers) can be provided, which would include:

- I/O rates per second
- Buffer Pool hit ratios
- Object working set sizes
- Average page residency time

For some environments, graphic-based simulation tools have the additional ability to allow the capacity planner to *watch* the simulation. Thus, instantaneous utilizations as well as averages could be seen for any resource involved in the simulation.

4. ***What is Really Needed?***

- Ideally, we want the speed of execution of queuing models - especially as we're in the "what-if" stage of building the buffer pool configuration. We could use the queuing approach to quickly build models of nearly all-possible configurations, and evaluate these quickly. Poorly performing configurations should stand out immediately, thus leading to creating a "*short list*" of possible "*real*" configurations to analyze further.

- We want the *accuracy and detail* of simulation models - especially when we're close to settling on a final configuration. Each of the resulting "*real*" configurations could be evaluated for "*at-the-edge*" conditions with simulation, thus completing the analysis process by illustrating the best configuration for our environment in even the most convoluted situations.

With the proliferation of fast workstations, we fully expect to see emerging simulation tools that will combine the positive features of both simulation and analytic modeling for general modeling applications. We fully expect domain-specific simulation tools to be so fast that the speed usually expected from solving a queuing model would be achievable through simulation (as is the case with ***Buffer Pool Tool***). Again, the key for the capacity planner would be to use a rapidly solvable modeling approach to eliminate poor configuration alternatives quickly. When left with only several remaining configuration alternatives, the capacity planner would turn to simulation to evaluate those environments under different "*at-the-edge*" conditions to get the more detailed and accurate results.

5. ***Conclusion: Who Have We Been Listening To?***

Capacity Management techniques have been developed focusing on system resources used over the last 25 years. While IBM provided the necessary operating system, the necessary *resource management infrastructure* emerged primarily from third party companies. As the old saying goes, history does repeat itself! From the humble beginnings of SNAP/SHOT to evaluate IMS-based applications, we come again to consider the use of simulation for capacity planning. So whom do we listen to as far as capacity planning goes?

Artis[7], nearly fifteen years ago, discussed the maturity levels of organizations as their capacity planning efforts mature. Amazingly, his analysis holds up well today for database-centric environments. At the *Vendor* stage, all capacity planning services are provided as part of a hardware vendors' marketing effort. This was in large part due to the fact that we did not know what to buy or how to set it up, so we used *trust* in our favorite vendor. As the organization grew and matured, we've assigned more people and importance to the capacity management function, until at the *Mature* stage, corporate executives perceive capacity planning results as essential to the decision making process. While many organizations would consider their mainframe capacity planning efforts as mature, these same organizations are at the *vendor* stage as far as purchasing configuring a mission-critical database-centric application! Primarily due to the ever dropping cost of hardware, individual managers have gone out to their favorite vendors, listened to a marketing pitch, and have made the buy decision for database gear. Unfortunately, the traditional capacity planner has not always been included in the process. Thus, the next few years will see the planning function return to central IS management in many shops.

In recent times, we must ask how we evolved to this point? Primarily cost; the hardware has, for the most part, become a commodity, like bread or eggs in the grocery store! A 16 Megabytes upgrade to the mainframe once required a 2-month justification study; today, we can buy 16 Megabytes of memory for a PC at our local computer store for about $40! So rather than burden the planner with commodity shopping, managers took on that responsibility.

This notion of hardware being a commodity has thrust capacity management under the microscope - do we need capacity planning any more? Capacity Planning is now going through a re-engineering process. Simply stated, we need, to be able to first, quickly identify system resources being used and whether those resources are degrading performance. Then, once identified, we need tools that can ***offer solutions*** - that is, the methods employed by today's tools should rapidly identify alternative configurations that will ***immediately improve performance***. What becomes more important are questions like "*Is the configuration providing adequate performance?*" or "*Which database resource will become the critical bottleneck and under what load conditions?*" We need to understand how to grow key database-centric applications; that is, is a particular application scalable? How many more users can be added, and what is ratio of users to memory or processor. These questions force the capacity planner to seek a new perspective. Simulation models comprise a technology that will help answer these questions - if applied at the appropriate time in the planning process.

The ability to predict, accurately, the effect of changes - and to provide guidance on how to accomplish complex tasks is a key requisite for improving performance, improving the ability of the analyst to complete their job, and optimizing the performance/hardware trade-offs.

[7]Artis, Dr. H.P., *The Five Stages of Capacity Planning*, CMG Proceedings, 1985, p.564

For More Information

We provide unique solutions for your difficult DB2 performance problems. Our worldwide client base and proven track record ensures that you can improve your system and application performance, just as our other clients have done. To learn more about Responsive System's software products, performance solutions, or consulting services, visit our web site and contact us either by email or by calling one of the telephone numbers below. .

Responsive Systems Company
281 Highway 79
Morganville, NJ 07751
(732) 972-1261
(800) DB2-EXPErt
(732) 972-9416 FAX

Website: **www.responsivesystems.com**

Email: **db2xpert@superlink.net**