Proceedings of the 1996 Winter Simulation Conference
ed. J. M. Charnes, D. J. Morrice, D. T. Brunner, and J. J. Swain

USE OF SIMULATION TO TEST CLIENT-SERVER MODELS

Yogesh L Deshpande
Roger Jenkins

Faculty of Business and Technology
University of Western Sydney, Macarthur
P O Box 555, Campbelltown
NSW 2560, AUSTRALIA

Simon Taylor

Department of Computer Science and
Information Systems
Brunel University
Uxbridge, UB8 3PH ENGLAND

ABSTRACT

Simulation is used for many purposes: for example, to analyse a complex system, to visualise the functioning of a system, and to optimise or tune a system. While there is no limitation on the use of simulation, the general consensus is that an analytical solution, if one is possible, is always to be preferred to simulation as a methodology. In the field of information systems, client-server models exhibit a degree of complexity and richness not amenable to easy analytical solutions, except for some specific algorithms useful in limited contexts. Simulation could, therefore, be a good strategy to analyse the client-server systems and help in better implementation of feasible solutions. This paper examines the current state of client-server models and use of simulation in dealing with the problems encountered. The paper then compares the seven-layer OSI model for communications architecture and recommends that a similarly layered approach is likely to prove useful in simulating client-server systems. In the process, the paper also points out that the simulation models bring into a sharp focus the importance of software metrics, an area of vital importance in software development.

1. INTRODUCTION

Simulation is a powerful tool in analysing complex systems. It is used for a wide variety of purposes such as optimising or fine-tuning a system, visualising the functioning of a system, or simply for better understanding through the analysis of a system. While there are few limits on the use of simulation, the general consensus is that, for a given problem, an analytical solution, if one is possible, is to be preferred to simulation as a methodology. In practice, however, there are many situations and systems where interactions among the components of the system are not amenable to easy analysis and consequently analytical solutions are based on simplified models. Client-server (c-s) systems are a good example of these limitations and hence a field which could benefit from simulation studies.

The use of simulation to test the design of a client-server system is, however, not so straightforward. Simulation as a strategy for problem-solving requires a good understanding of simulation modelling and the computing and statistical aspects of simulation. Client-server systems themselves can be quite complex with possibly several options for the systems designer to choose from. Since client-server modelling is a relatively recent development, there is a shortage of experts in this area, widely acknowledged by the software industry. Additionally, the developers of such systems are generally under severe time constraints, unable to experiment sufficiently with the alternative designs and not necessarily familiar with simulation strategies or tools. There is thus a possibility that for a number of reasons, the systems designers may neglect simulation altogether or use it in a manner which may limit the usefulness of the results obtained, or even invalidate them.

The difficulties enumerated above notwithstanding, this paper joins a growing body of work which contends that there is much to be gained by the use of simulation in the client-server environment. Even the simplified models presented in this paper bring into a sharp focus the complex nature of layers of sub-systems, such as network topologies and communication protocols, on which a client-

server system is based and about which a designer is forced to make assumptions. At the same time, a review of the literature in this area shows that simulation modelling is used either in a somewhat limited capacity, concentrating on sub-systems and algorithms or at a level of detail which makes it complex and expensive (Robinson 1994). There is thus an untapped potential for the use of simulation of the systems. The possible application of simulation strategy to client-server problem also points to the vital need of proper measurements of the client-server system parameters, an issue directly linked to software metrics and performance monitoring. The paper concludes that even a simple simulation model of a client-server system is beneficial to the system designers and developers.

The paper is organised as follows. Section 2 briefly describes the client-server models. Section 3 relates simulation work to client-server and finds that the modelling exercise in this area includes details of communication networks, hardware and processor characteristics, algorithmic analysis, query optimisation and file or database distribution. Section 4 then relates the client-server systems to the seven layers of the Open System International (OSI) model to bring out the hidden complexity of the models and to recommend simulation strategies in a similarly layered modelling approach. Section 5 describes a case study under way, using a commercially available simulation package. Section 6 deals with the question of statistical aspects of simulation and compares them with software metrics. Section 7 concludes the paper.

2. CLIENT-SERVER MODELS

Almost all writings on this subject start with a statement that there is no single definition of what is meant by client-server model of computing. Nevertheless there is a fairly good agreement about what does characterise them (Stallings and Slyke 1994, IESC Report 1995). The following working definition by Boar (1993), later adopted by the IES Committee (IESC) set up by the Australian Federal Department of Finance (1995) is relatively straightforward: "*A computing model whereby the application processing is partitioned across multiple processing platforms and where processors cooperate to complete the processing as a single integrated task*".

This definition emphasises the logical nature of the client-server model and not the physical one which would include the hardware and network components.

Client-server models belong to the class of distributed computing models. In theory, even one client and one server on the same hardware platform could qualify as a client-server system. In practice, though, there are likely to be many clients, based on multiple platforms, and being served by at least one server, connected through networks. Figure 1 illustrates a general client-server model with multiple clients connected to multiple servers via a local or wide area network.

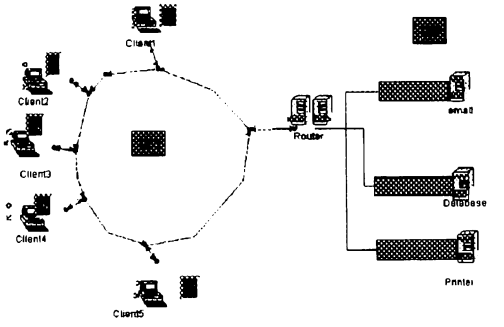

Figure 1: A simple client-server model

A more elaborate model put out by the Gartner Group (1994) has gained wide acceptance among computing practitioners. They define client-server by postulating five configurations through combinations of the three components into which any application may be subdivided:

- Presentation
- Processing
- Data management

These components may be distributed across the client(s) and the server(s) in five ways, as shown in Figure 2.

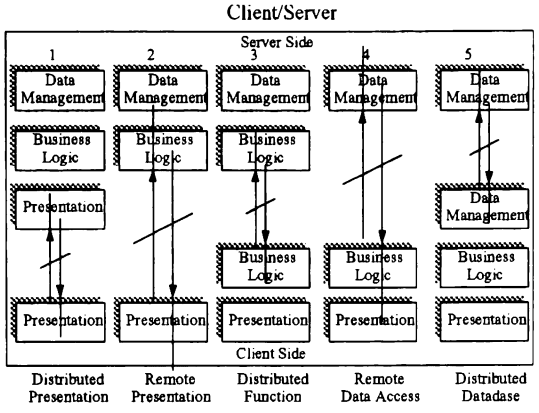

Figure 2: The Gartner Group models of client-server systems

The first two models, distributed and remote presentations, represent a situation when the server is the dominant partner and the client is either a dumb terminal or is there mainly to provide a user-friendly graphical user interface (GUI). The remaining three allow the client to carry out processing either partially or wholly on the client(s) and are of major interest here. In these situations, a designer is faced with many choices about how to split processing and data management functions between the clients and servers.

Consider a hypothetical application for development. Typically, it will contain tens, if not hundreds, of modules dealing with user interface, inputting data, validation of input data, queries, report generation, processing and output messages. The client-server architectural options will allow a split of these functional modules on one or more clients on the one hand and on one or more servers on the other. Furthermore, the server(s) will also be responsible for data management which may lead to distribution of databases and tables over many servers. The performance of the system will clearly depend, among other things, upon how these modules are distributed across all the clients and servers in the system.

A number of authors have written in detail on strategies and guidelines for such functional and data distribution (Booth 1981, Krantz 1995, among others). However, since the design of a client-server system is specific to a given application, such guidelines have been necessarily general in nature. Consequently, details of actual implementation of a system design and its effects on the system performance are also of general nature. Anecdotal evidence points to the designers making choices about splits in client-server functionality on the basis of experience, intuition and trial-and-error methods.

The foregoing description and discussion indicate the possibility, and desirability, of applying simulation strategies to test and improve client-server designs. The following sections elaborate on this theme.

3. SIMULATION AND CLIENT -SERVER

This paper is mainly concerned about using simulation modelling in a specific environment, viz: client-server systems. Familiarity with simulation methodology will, therefore, be assumed and the reader is referred to well-known titles such as Tocher (1963), Naylor et al (1966), Banks and Carson(1984), Pidd (1992), Szymankiewicz et al (1988) for further elaboration of simulation strategies. Any comments related to the simulation methodology in this paper should be read only in the context of client-server systems modelling and design and not as any critique contributing to the field of simulation.

Section 2 described the five generally accepted models of client-server computing. It also pointed out that the design of a client-server system is specific to a given application. For the discussion of use of simulation, for the rest of this paper, therefore, it is necessary to make certain assumptions which are as follows:

- There is only one server with more than one client;
- The client will have presentation and partial functionality while the server will carry out the remaining processing and data management, i.e. the third model in the Gartner Group's classification will be implemented;
- The application will be a database management;
- The systems designers want to reduce the response time to the minimum.

These assumptions are really meant to facilitate identification of characteristics of the client-server environment germane to simulation. As will be seen, any or all of these assumptions may be relaxed without adversely affecting the conclusions.

The assumptions made above are partial details of only a logical model. They do not at all consider the possible variations in hardware

and software platforms, nor are the network and communication parameters identified. The view is of a systems designer who takes all the other parameters as given and is only concerned with decisions about how to split the functionality within the application across multiple clients and the server.

A typical working scenario of the system would involve the following:

- a client would present a user interface to validate input and generate requests (transactions) for the server to process;
- the transactions could take the form of simple queries, or updates to the database, or requests for detailed reports;
- the server would receive these transactions/ requests over the network and respond appropriately;
- the server would carry out database management tasks, as and when needed.

In order to simulate such a system, a simulation expert would want to know answers to questions such as:

- what are the possible types of transactions and what are their (statistical) distributions?
- do all clients follow the same distributions?
- how much processing time is needed on the client and how much on the server?
- what is the communication overhead and how is it measured?

Depending on the level of detail in a simulation model, there could be many more similar questions to which answers may be required.

Given the required details, a simulation model may be built and statistically valid experiments run to report on various combinations of functional division between the clients and the server.

The idea of using simulation in client-server environment has been explored by several researchers in much greater detail, as reported in literature (Smith 1990, Jain 1991, Gold 1993, McBeath and Keezer 1993, Shen and Butler 1994, Robinson 1994, Komatsu, Wakayama and Nose 1994). The level of detail, however, is a potential challenge to systems designers in general, as discussed below.

The main characteristic of the first three questions about a simplified client-server system raised above is that they are related to the application under consideration and not to the hardware/software platform on which the application will be built. They are also the questions of interest to any systems designer. The performance of a system based on this logical design could be analysed without much more data.

In the final analysis, however, performance of a given system will depend upon many factors, such as network topology, transmission protocols, client and server processor speeds, number of clients supported, choice of software packages and underlying operating systems. A systems designer is aware of these factors but has far fewer measurements available on which to judge the relative contribution of these factors and hence their likely effect on the system performance. Many of these factors belong to areas of specialisation about which a systems designer has only a vague model in his/her mind.

In the references cited above, for example, Shen and Butler (1994) include the network characteristics in their model-building, Smith (1990) and Robinson (1994) allow hardware characteristics, Komatsu, Wakayama and Nose (1994) incorporate internal processes of a computer system and models using the so-called Remote Procedure Calls (RPC) have also been reported.

The detailed simulation modelling would make the subsequent experiments and output analyses more valid than those based on a less detailed one. On the other hand, Robinson notes that the increased complexity is a drawback and, in many cases, simpler models may be adequate to address basic issues.

The foregoing analysis suggests that what is likely to be useful to a systems designer is the ability to include the levels of detail selectively and in such a way as to make modelling easier and more tractable. Section 4 suggests ways of doing that by utilising the familiar Open Systems Interconnection (OSI) model.

4. CLIENT-SERVER AND OSI

As very briefly discussed in Section 3, client-server modelling is complicated by the considerations of a large number of variables, ranging from software and hardware platforms through to networks and communications protocols. The OSI model, representing both client and server (Figure 3), is in fact a good guide to sort these factors in layers where they belong.

	Layer
1	Physical
2	Data link
3	Network
4	Transport
5	Session
6	Presentation
7	Application

Figure 3: OSI model of communications architecture

Reverting back to the simplified example of Section 3, it is seen that the first three questions about the system characteristics belong to the Application layer whereas the fourth properly formulated straddles two layers, Network and Communications. From the system designer's point of view, a model involving a (client) application layer connected to (server) application layer is easily understood. Such a model would then necessarily assume that the characteristics of the layers below remained unchanged and therefore allow exclusive consideration of allocation of functionality between a client and the server.

This approach logically leads to another step in the full exercise of simulation. Each layer, especially the Network and Communications Protocols can be independently modelled, in their own right, to a level of detail relevant to a given system. It may be then possible to draw upon work already done in these areas (e.g. Mitrani 1987, Law and McComas 1994, Komatsu, Wakayama and Nose, 1994, Rumekasten 1994, Shen and Butler, 1994). Since there will generally be more than one information system running on these platforms, these Network and Communications models could be directly utilised in other simulation exercises. The layered models also allow analysis of possible or proposed changes to specific characteristics of any given layer.

The layered approach need not follow the OSI model exclusively. IESC report (1995), for example (Figure 4), or TCP/IP protocol does not adhere to seven layers. As mentioned above, what is needed is a good model at the appropriate level and a systems designer may in fact be in the best position to decide upon the layer of interest.

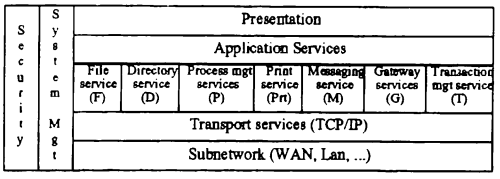

Figure 4: Layered model of client-server systems (Source: IESC Report 1995)

Work is in progress to implement this layered approach for the simple system described above. Even at an early stage, this modelling exercise has highlighted benefits, such as better understanding of issues, the role of Software Metrics and the possibility of using commercially available and inexpensive simulation packages. Section 5 describes the case study and discusses the relevant issues.

5. CASE STUDY

The case study is based on the assumptions listed in Section 3. The simplicity of the model meant that there was no need to write customised simulation package. The choice of the package was ProModel, for its ease of use, ability to animate the model, statistical support and the at least one author's familiarity with it. The first models were shown to a number of people who suggested a few, later obvious, improvements. The results of these attempts have been described in Sections 3 and 4 and led to the adoption of the layered approach to client-server systems modelling. There are now three separate models, one each dealing with the Application layer, the Network layer and the Communications Protocol layer.

To illustrate the nature of these models, consider how the system would deal with a transaction generated by a client. The broad steps involved in processing the transaction would be as follows:

- The client side would present the user interface, accept input and process it as per a given specification, passing it on to the Network layer for further processing.
- The Network would process the transaction and pass it on to the Communications system.
- The Communications layer on the client side would connect with that on the server side

and ensure that messages were passed on correctly.

- These messages would travel upwards to the Application layer of the server in the reverse fashion, processed and the output would travel back to the client in a similar manner.

Although this is a very simplified model, it has already proved to be useful in many ways.

Firstly, the exercise brought home to the people involved in the discussion that there are a number of possible views of the same system, all of them important, and that it is essential to develop a common language across the different expertise. The layers clarified to everyone exactly what was under discussion and needed further clarifications.

Secondly, it forced the participants to consider different ways of measuring the "traffic". A transaction viewed at the client Application layer goes through a few transformations before reaching the server Application layer, again as a complete transaction. The statistical distributions of the types of transactions and of the time it takes for these transactions to reach the destination are likely to be different at each layer. A larger question is thus raised about the appropriate measurements and collection of data in order to validate the simulation exercise.

Thirdly, the case study has highlighted the appropriateness of constructs available in the simulation packages. ProModel is primarily designed to aid in modelling manufacturing systems and uses icons and terminology of that environment. The success and speed of modelling depends substantially on the ability of experts participating in the study to interpret the terminology flexibly and not be diverted by the language used by the package. This is not always so obvious.

A further advantage which simulation modellers clearly recognise, was the effect of animating the model. As a generalisation, systems designers tend to think in more abstract terms, in algorithmic patterns and, at times, in static models of their systems. Animated simulation models of even the simplified models seems to generate much more interest than even extended discussions.

As mentioned above, this case study has raised a major concern about the measurements of any given system and the different points of view of simulation strategists and systems designers. Section 6 briefly discusses these issues.

6. STATISTICAL ASPECTS OF SIMULATION AND SOFTWARE METRICS

There is a lot of similarity between the approaches adopted in simulation modelling and systems design (Robinson 1994). However, after the logical design of a system model, the system developers proceed to its physical design straightaway, very often under severe time pressure. Validation of the models they may have built is static and through structured walkthroughs or meetings with the responsible clients or through proto-typing.

In contrast, simulation strategists attach a great deal of importance to relevant and accurate measurements, or data, before any experiments are conducted and models validated.

One of the early benefits of the case study, as mentioned above, was the focus on input and output data of the given system. From the point of view of validating the simulation model of a client-server system, data would be needed to cover as many of the variations in the systems parameters as possible. Depending on the level of modelling, such data would include measurements specific to the application, network and communications layers before simulation experiments could begin.

The conceptual modelling exercise and consequent attention to measurements also is extendable to post-implementation systems. Whatever assumptions are made in the pre-design stage and, data collected accordingly, the data capture and further validation tests can be continued throughout the systems life cycle, leading to more pro-active systems maintenance.

It may be pointed out that the model can only perform the task of highlighting the paucity and/or the (poor) quality of data about the system(s) under consideration. Changes in practice and improvements in the quality of data are matters for action only by the appropriate authorities.

7. CONCLUSIONS

Simulation modelling and experiments are used widely and for a variety of purposes. This paper

joins a growing body of simulation work in the client-server environment. The paper analysed the client-server model to relate it to the seven-layers of OSI model and suggested that a similar, layer-based approach be used in the simulation exercise. The benefits of this approach are in terms of clearer understanding of the client-server models, possibility of using commercially available simulation packages, and a sharp focus on the issues of software metrics. The modelling exercise alone is worth the effort to clarify the issues to the software community. The client-server environment is still unfamiliar to many people and causes difficulties in understanding since it engages many rapidly-changing technologies. Simulation modelling makes the client-server models much more tractable and thus should prove to be of great benefit overall.

REFERENCES

Banks, J. and Carson, J.S. 1984. *Discrete-Event System Simulation*. Prentice-Hall:

Boar, B.H. 1993. *Implementing Client/Server Computing: A Strategic Perspective*, New York, NY: McGraw-Hill.

Booth, G.M. 1981. *The Distributed System Environment*. New York: McGraw-Hill.

Gartner Group. 1994. *Client/Server Computing in the 1990s*, Stamford,CT: Client/server Logical View Personal Advisory Service.

Gold, J. 1993. Simulation aims to predict software performance. *Software Magazine*. July: 15-19.

Information Exchange Steering Committee (IESC). 1995. *Client/Server Computing in the Australian Public Service: The Next Wave?* Canberra: Department of Finance, Australian Federal Government.

Jain, R. 1991. *The Art of Computer Systems Performance Analysis*. New York, NY: John Wiley and Sons.

Jiang, Y., Cao, J. and Gupta, G.K. 1992. *Dynamic Distributed Query Processing Through Query Migration*. In *Research and Practical Issues in Databases, Proceedings of the 3rd Australian Database Conference*, ed. B. Srinivasan and J. Zeleznikov, 197-209. Melbourne: World Scientific.

Komatsu, T., Wakayama, H. and Nose, J. 1994. Model Description in the INSYDE Simulator for Evaluating Large-scale Computer System Performance. In *Proceedings of the 1994 Winter Simulation Conference*, ed. J.D.Tew et al, 1272-1279, Boston: The Society for Computer Simulation.

Krantz, S. 1995. *Real World Client/Server*. Gulf Breeze, Fl: Maximum Press.

Law, A.M. and McComas, M.G. 1994. Simulation of Communications Networks. In *Proceedings of the 1994 Winter Simulation Conference*, ed. J.D.Tew et al, 166-170, Boston: The Society for Computer Simulation.

McBeath, D.F. and Keezer, W.S. 1993. Simulation in Support of Software Development. In *Proceedings of the 1993 Winter Simulation Conference*, ed. Evans, G.W., Mollaghesemi, M., Russell, E.C., Biles, W.E., 1143-1151. Baltimore, MD: Association for Computing Machinery.

Mitrani, I. 1987. *Modelling of computer and communication systems*. Cambridge: Cambridge University Press.

Naylor, T.H., Balintfy, J.L, Burdick, D.S. and Chu, K. 1966. *Computer Simulation Techniques*. John Wiley:

North, K. 1996. Performance Testing, ODBC, and Native SQL APIs. In *Dr Dobb's Sourcebook*, Vol 21, Issue 13, (January-February), 17-21.

Pidd, M. 1992. *Computer Simulation in Management Science*. John Wiley and Sons:

Robinson, J. 1994. Capacity and Performance Analysis of Computer Systems. In *Proceedings of the 1994 Winter Simulation Conference*, ed. J.D.Tew et al, 34-41, Boston: The Society for Computer Simulation

Rumekasten, M. 1994. Simulation of Heterogeneous Networks. In *Proceedings of the 1994 Winter Simulation Conference*, ed. J.D.Tew et al, 1264-1271, Boston: The Society for Computer Simulation

Shen, J. and S. Butler. 1994. Performance Modeling Study of a Client/Server System Architecture. In *Proceedings of the 1994 Winter Simulation Conference*, ed. J.D.Tew et al, 1280-1287, Boston: The Society for Computer Simulation.

Smith, C.U. 1990. *Performance Engineering of Software Systems*. Reading, MA: Addison-Wesley.

Stallings, W. and Van Slyke, R. 1994. *Business Data Communications*. 2nd edition. New York, NY: Macamillan.

Szymankiewicz, J., McDonald, J and Turner, K. 1988. *Solving Business Problems by*

Simulation. McGraw-Hill: Maidenhead, England.

Tocher, K.D. 1963. *The Art of Simulation.* English University Press: London.

AUTHOR BIOGRAPHIES

YOGESH L. DESHPANDE is a senior lecturer in the Department of Computing and Information Systems in the Faculty of Business and Technology at the University of Western Sydney, Macarthur. His research interests include discrete event simulation modelling, client-server systems, and information systems. He is a member of the Australian Computer Society, Computer Society of the IEEE, and the Operational Research Society.

ROGER JENKINS is a lecturer in the Department of Manufacturing and Quality Systems in the Faculty of Business and Technology at the University of Western Sydney, Macarthur. His research interests include discrete event and continuous simulation modelling and materials and resource planning.

SIMON TAYLOR is a lecturer in the Department of Computer Science and Information Systems at Brunel University. His research interests include simulation modelling, and distributed and parallel simulation.