Proceedings of the 1991 Winter Simulation Conference
Barry L. Nelson, W. David Kelton, Gordon M. Clark (eds.)

SELECTING SIMULATION SOFTWARE

Jerry Banks

School of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332-0205

ABSTRACT

The selection of appropriate simulation software from the vast number of packages available is a difficult task. This tutorial describes the types of software packages that are available, discusses some factors that should be considered in selecting software, and provides suggestions for the selection process.

1 INTRODUCTION

Let us say that you or your firm decides to use discrete-event simulation for the solution of some manufacturing problems. You obtain a copy of the Directory of Simulation Software (1991) published by the Society for Computer Simulation, and you find 37 entries that show manufacturing as one of their application areas, and you remember seeing advertisements for others. Suppose that your interest is in animation. There are 31 entries that show animation in their area of application. Some 15 of the 31 animation entries are also in the manufacturing applications list, and you know of some other animation software used in manufacturing. The bottom line is that there is a bewildering number of software packages available. A person new to simulation could spend months looking at vendor literature and demonstration diskettes, and examining evaluation copies of software.

The purpose of this tutorial is to help with the decision-making process. A set of factors that should be considered when making a software decision is presented and some guidance is provided.

Although the appropriate software is very helpful in completing a simulation project, it will not do all of the work by it self. Simulation authors show model coding as one of perhaps twelve steps in performing a simulation study (Banks and Carson, 1984). Model coding can require anywhere from 30% to 40% of the total required work in a typical simulation study (Law and Haider, 1989). In addition to software, the model builder needs to know something about simulation methodology. This expertise is usually obtained through university instruction, a public short course or a specialized private short course.

2 CLASSES OF SIMULATION MODELING TOOLS

Table 1 shows four classes of simulation modeling tools. The first class consists of spreadsheets such as LOTUS 1-2-3 and QUATTRO. Although these are not generally known for their simulation capabilities, it is possible to perform simulation using the @RAND function, or some similar command. @RAND generates a random number, uniformly distributed between zero and one. Applications using spreadsheets for simulation modeling are given by Seila and Banks (1990a, 1990b).

Developing simulation models using spreadsheets takes a minimal amount of time, mainly because the systems are small or the applications are limited. An example application would be the determination of the distribution of the returns from a portfolio of possible real estate investments, where each investment has probabilistic components making up its return. Note that the model is of the Monte Carlo form, i.e., the system is not dynamic (there is no contention for resources, no waiting lines, etc.). A year is simulated in one pass, with the necessary random numbers for the individual components of the return, the total return for the year is determined, then the next year is simulated in one pass, until the horizon is reached. Many runs of the scenario are made, and a distribution of net returns is plotted.

When using a spreadsheet, the output is defined by the user. The output is usually in graphic form, with some very attractive results. Such attractive outputs are made possible by business graphics offered by many spreadsheet packages.

The next category in Table 1 includes rapid modeling tools such as ManuPlan/SimStarter, marketed by Network Dynamics, Inc. (Suri, Derleth, and Tomsicek, 1990). The purpose of these tools is to gain an idea about such measures of performance as throughput and bottlenecks. The system is modeled in very general terms, omitting many of the details in order to get an idea about the performance measures. In many instances this level of output is sufficient as it answers the questions that are being asked in a timely manner.

The next classification in Table 1 is the simulator.

Table 1
Classes of Simulation Modeling Tools

Consideration	Spreadsheets	Rapid Modeling Tools	Simulators	Simulation Languages
Development Time	Minimal	Minimal	Moderate	Moderate to Substantial
Model Control and System Complexity	Poor for Dynamic Systems	Vague/Moderate	Must Comply with Software Constraints	Excellent/Virtually Any Complex System
Output	User Developed and Defined	Statisically Adequate, Report Oriented	Varies by Package, Typically Report-Oriented	User Defined Reports, Customizable
Accuracy/ Fidelity	Generally Inaccurate for Dynamic Systems	Good for Rough Cut Planning	Varies with Level of Assumptions	Excellent
Training	Minimal	Moderate, Often Uses Spreadsheet-type Interface	Moderate	Moderate to Substantial
Environment Best Suited to	Static Systems Deterministic Operations	Low Complexity Probabalistic Operations/ Multiple Alternatives	Medium Complexity/ Probabalistic Operations/ Data Available/ Specific Applications	High Complexity/ Probabalistic Operations/Data Available/General Applications

Reproduced with permission of Bob Kittel and Doug Smith, Advanced Industrial Concepts, Sacramento, CA.

These are data-driven simulations that require no programming for certain types of models. However, the more robust simulators allow for programming either within the simulator, by dropping into another language, or by adding compiled code. The model must be within the framework of the software, as it is very difficult to model beyond the scope provided. Some simulators are much more robust than others in this respect. The time required for modeling using a simulator is moderate, regardless of what the vendors say. Real problems are almost always more complex than the made-up examples that are used for teaching purposes. The outputs from the simulators are usually rather impressive. They produce attractive business graphics, and create files that can be read into a spreadsheet or graphics package. A two- or three-day training course is usually provided by the vendors, and it is generally important to receive this training rather than to rely only on the documentation.

The last column in Table 1 pertains to simulation languages. Virtually any realistic problem can be modelled using a simulation language, although it may require a large investment of time to insure that all of the details have been captured. However, this is an advantage of simulation languages: they allow very detailed modeling. The output from a simulation

language varies from standardized to tailor-made. Even the popular products with standardized output provide the capability to obtain a customized output instead of, or in addition to, the standardized output. Three- to five-day basic training courses are usually offered either by the vendors or by third parties. To obtain more expertise, an additional one- or two-day course can be taken, after having worked with the software long enough to attack some real problems.

Carson (1990) classifies the last two columns in Table 1 as follows:

Pure simulator
Simulator with programming-like capability
Simulation language with simulator-like extensions
Simulation language

This classification is preferable to simulator software vendors that have responded to the suggestions of users to provide a means for conditional routing. Likewise some simulation language vendors have provided material handling features that are similar to those provided in several simulators.

3 FEATURES OF SIMULATION SOFTWARE

Haider and Banks (1986), Law and Haider (1989), and Banks, et al. (1991) discuss features that are important for simulation software to be used in the analysis of manufacturing systems. The features mentioned in the three references, and some others, are classified in this tutorial as Input, Processing, Output, Support, and Cost.

Words of warning are in order before you use the criteria shown in this section. First, you must know which criteria are appropriate and which are inappropriate. Hence, it is not necessary to ask for 3-dimensional animation unless you know how you will use it. It is not necessary to ask for 10 random number streams or 10 distributions, etc., unless you know their purpose and that you will need them. The bottom line is that you must understand the criteria and what is important for your situation. Second, the response to some of the criteria should not be judged on a "yes" (present) or "no" (absent) basis, but on the ability of the language or simulator to perform a service needed. A good example is the first criterion in the list, interface to other software (FORTRAN, C, Pascal, ...). A simulation language or simulator should be judged on its ability to avoid the need for FORTRAN, etc. (Carson, 1990), not simply whether it can be interfaced with a programming language.

The features discussed in this section are meant for application to simulators and simulation languages. Spreadsheets and rapid modeling tools are not meant for detailed simulation analysis so they should not be judged by these features.

3.1 Input Features

Interface to other software: Ability to drop into another language if the simulation language or simulator requires the same to model certain details.
Input data analysis capability: Ability to determine an empirical or mathematically described distribution and its parameters given raw input data.
Portability: The program can be written on one class of computer and run on another class of computer(s).
Syntax: Modeling terminology should be easily understood by the user and should be consistent and unambiguous.
Input flexibility: Should be receptive to inputs that are provided interactively and/or in batch mode.
Interactive debugger: Allows a modeler to control simulation execution and to access the important data that are being collected.
Modeling flexibility (applies to simulation languages only): Simulation language perspectives include process interaction, event oriented, and continuous. Ideally, a language should allow for the selection of any of the three perspectives or combinations of perspectives to be included in the model.
Modeling conciseness (applies to simulation languages only): Powerful blocks or nodes in the process-interaction approach of a simulation language or powerful commands in the event-oriented perspective enable the development of compact models.

3.2 Processing Features

Execution speed: Is important in constructing models since numerous runs are made for validation purposes. Is very important in making production runs consisting of many scenarios, each of which is replicated numerous times.
Model size: For PC's running under plain DOS, this factor is important. For computers running under Extended DOS, OS/2, or UNIX this factor is much less important.
Material handling: Possible MH capabilities include transporters, AGVS, conveyors (transport, accumulation, etc.), robots and cranes. The presence of these MH capabilities does not assure that each is an accurate representation of the system under investigation. If the simulation problem does not require MH, then this feature can be ignored.
Random variate generators: Includes the basic distributions such as exponential, uniform, triangular, and normal, plus empirical distributions.
Reset: Allows the discarding of observations recorded

prior to reaching steady state.
Independent replications: Ability to run the simulation repeatedly using a different set of random numbers.
Attributes: Local values assigned to the transactions moving through the system.
Global variables: Values available to all transactions moving through the system.
Programming (applies to simulators only): The user can enter code to incorporate special characteristics in the model. In many instances, this feature makes the difference between a truly helpful model and an oversimplified model that provides no useful results.
Conditional routing (applies to simulators only): Allows sending transactions to different locations depending on a prescribed condition, such as the current number waiting for a resource, or whether a resource is in operation or not in operation.

3.3 Output Features

Standardized reports: Includes performance measures such as average number in queue, average time in queue, average utilization of resources, and throughput.
Customized reports: Computation and display of specialized performance measures and the tailoring of output reports for presentation to managers.
Confidence intervals: For desired measures of performance in order to develop a range which contains the true value of the measure with stated accuracy.
Business graphics: Bar charts, pie charts, histograms and other plots that can be sent to a laser printer.
File creation: Output that becomes input to other software such as a spreadsheet or database.
Tracing capability: Shows each event and the status of the active transaction at the time of the event.
Data base maintenance: Simulation requires numerous replications of one or more scenarios. This feature provides for the collection of the output from these scenarios in an organized fashion.

3.4 Environment Features

These self-explanatory features include the following:

- Ease of use
- Ease of learning
- Quality of documentation
- Animation capability
  - Ease of development
  - Quality of picture
  - Smoothness of movement
  - Portability for remote viewing
  - CAD interface
- On-line help
- On-line tutorial
- Customer support
  - Training
  - Technical support
  - Updates and enhancements

3.5 Cost Feature

A very difficult feature since the range is so large: about $1,500 to about $80,000 as an initial outlay for simulation software. Another cost consideration is in hardware requirements. Verify these requirements with software vendors and realize that special graphics cards, math co-processor chips, large memory, and so on, may require additional outlay of funds. The cost should also include the time spent learning to use the software and the time required for building models. Software costs change as new features are added and as vendors position themselves in the market.

4 WHICH SIMULATION MODELING TOOL IS THE RIGHT ONE?

The answer to this question can be difficult, but some guidance can be given:
You may need more than one tool: Many simulation analysts use several tools. They may use both a simulator and a simulation language, or they may use more than one simulation language, etc. Using one tool is the vice-grip philosophy, i.e., just reset the opening of the pliers' nose for a different size of nut. Using multiple tools is the socket-wrench-set philosophy. The socket wrench set costs more, but performs better for the specific size nut encountered.
Get the greatest power that you can afford: Once a package has been selected, buy the fastest version of that software. Having simulation analysts wait for output is usually more costly than paying for more number crunching capability.
Accuracy and detail are available only with simulators and simulation languages: Spreadsheets and rapid modeling tools only have limited uses in answering questions where detail is important.
Beware of fancy ads and demos: This caution is especially true of some simulators. A simulator can have wonderful interfacing capability and produce attractive outputs, but may have a very weak and less than robust engine. This is not an indictment of all simulators!
Beware of checklists: Beware of checklists that list generic features of simulation software. The presence of some of these features is not nearly as important as the implementation and extent of capability offered by the software overall (Carson, 1990).

Obtain a trial copy of the software: Many vendors offer a student version, a limited version, or a trial copy of the software. It is advisable to obtain a copy of the software in this form, as well as the software documentation, and use it for solving a small version of the problem. It is still difficult to know the capability of the full version of the software until a real problem is tackled (Carson, 1990).

Ask the vendor to solve a sample problem: The problem would likely be a reduced version of the real problem which you are trying to solve. This request may involve a consulting fee, but it can have great payoff. If you are willing to put some money on the line, the vendor will know that you are serious. Otherwise, you should not expect a vendor to assign an analyst to a potential non-revenue activity. If the vendor can not solve the sample problem to your satisfaction, then you should look at other software possibilities. The payoff is in avoiding the purchase of software that will not solve the type of problem that you encounter.

Eliminate most products: An article in USA Today (Maney, 1991) describes the dilemma of consumers faced with a flood of products. (There were 13,244 consumer products introduced in 1990 including 123 new breakfast cereals and 130 new pet foods.) Research in many areas indicates that consumers initially eliminate a vast number of products, they consider six alternatives, they seriously look at three, and they purchase one. With 656 cars from which to choose, and 150 cable channels (in many locations) this or some similar procedure is necessary if we are going to make it through the morass. This elimination process can be applied to selecting simulation software. (For example, eliminate simulation languages, eliminate non-manufacturing simulators, ...).

You are going to buy a decent simulation language: The most popular simulation languages are all decent tools. Their features and performance may be quite different, but their robustness is impressive. You won't get stuck with a lemon.

Decide if you need animation: If you need animation, the total cost of a simulation language is going to be more, in some cases much more. Also, note that the popular simulators include animation, although it can be turned off in most instances.

The distinction is blurring: The distinction between simulators and simulation languages is blurring. Some of the simulation languages have moved toward the simulators by offering extensive material handling features, by automating the modeling via drop down menus, and by making it very easy to draw or import the static layout for animation. Some of the simulators have moved toward the simulation languages by allowing limited programming capability, attributes, and global variables. This makes for a more difficult choice. When there was a clear distinction between simulators and simulation languages, choosing one broad category or the other eliminated an entire set of options.

5 ON USING A SCORING MODEL

One of the suggestions in the previous section was to eliminate most products, paring the list to three. It would be quite possible to use a scoring model for this purpose. This could be accomplished by first assigning a value between zero and ten to each factor (interface to other software, input data analysis capability, etc.) in Section 3. These would then be summed and normalized so that the total adds to a convenient number, say 100. The resulting normalized number is called the factor weight. Then, each software package would be scored on a zero to one scale for each factor. This raw score (between zero and one) would then be multiplied by the factor weight to obtain the weighted score. The weighted scores are added to determine the software score. A 100 is the maximum software score. The three top software packages are then subjected to further scrutiny until a decision is made.

Although this procedure sounds rather straightforward, it would be difficult to apply for someone new to simulation. Experience using several simulation languages and simulators is required to know the shades of difference and how they can affect a specific user. In general, simulation vendors are not appropriate people to give advice. This is like asking the barber if you need a haircut.

One possibility is to ask an independent third party to help with the decision concerning simulation software. Many simulation consulting firms use either one or two products, having narrowed the available choices based on their experience, the kinds of problems that they solve, or their relationships with vendors. These consultants are not unbiased when it comes to recommending simulation software. However, they could be asked to determine if a particular selection could solve the type of problems to be encountered.

In the case of an independent consulting firm, a request for a software recommendation usually goes unanswered. By answering the question some amount of independence is lost. If a firm wishes to engage the consultant to study the firm's application and make a software recommendation, the consultant would probably be willing to accept such an assignment.

6 CONCLUSION

This tutorial began with a classification of simulation modeling tools. Then, a collection of features of simulation software were discussed. Next, guidance was provided for selecting a simulation modeling tool. Lastly, a technique was described to reduce the vast number of simulation modeling tools to a manageable few. The selection of simulation software depends on the problems to be solved as much as the characteristics of the various modeling tools.

ACKNOWLEDGEMENTS

The author would like to thank Ken Tumay of Production Modeling Corporation of Utah, Dan Brunner and Bob Crain of Wolverine Software Corporation, and S. Manivannan of Georgia Tech for their suggestions.

REFERENCES

Banks, J., E. Aviles, J.R. McLaughlin, and R.C. Yuan. 1991. The Simulator: New Member of the Simulation Family. *Interfaces.* **21**, 76-86.

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

Carson, J. 1990. Simulation Concepts in Manufacturing and Material Handling. *Autofact '90 Conference Proceedings,* 4-1 through 4-19, Society for Manufacturing Engineering, Detroit, MI.

Directory of Simulation Software. 1991, ed., Elliot Estrine, Society for Computer Simulation, **2.**

Haider, S.W., and J. Banks. 1986. Simulation Software Products for Analyzing Manufacturing Systems. *Industrial Engineering.* **18**, 98-103.

Law, A.M., and S.W. Haider. 1989. Selecting Simulation Software for Manufacturing Applications. *Proceedings of the 1989 Winter Simulation Conference,* eds., E.A. MacNair, K.J. Musselman, P. Heidelberger, 29-31, Institute of Electrical and Electronics Engineers, Washington, DC.

Maney, K. 1991. Consumers Face Flood of Products. *USA Today.* July 12, B-1 and B-2.

Seila, A.F., and J. Banks. 1990a. Use an Electronic Spreadsheet as a Decision Support System. *Industrial Engineering.* **22**, 40-46.

Seila, A.F., and J. Banks. 1990b. Spreadsheet Risk Analysis Using Simulation. *Simulation.* **55:** 163-170.

Suri, R., D. Derleth, and M. Tomsicek. 1990. Modeling Manufacturing Systems Using ManuPlan and SimStarter-A Tutorial. *Proceedings of the 1990 Winter Simulation Conference,* eds., O. Balci, R.P. Sadowski, R.E. Nance, 168-176, Institute of Electrical and Electronics Engineers, New Orleans, LA.

AUTHOR BIOGRAPHY

JERRY BANKS is an Associate Professor of Industrial and Systems Engineering at the Georgia Institute of Technology. He received his Ph.D. from the Oklahoma State University in Stillwater. His research interests are in the areas of real-time knowledge-based simulation modeling and the verification and validation of large scale simulation models. He teaches courses in simulation and quality control. He is the author or co-author of nine books and numerous technical articles. Texts related to simulation co-authored by Professor Banks include *Discrete-Event System Simulation* published by Prentice Hall in 1984, and *Getting Started with GPSS/H* published by Wolverine Software in 1989. Professor Banks is a founding partner of the simulation consulting firm Carson/Banks & Associates, Atlanta, Georgia. He serves as the Institute of Industrial Engineers' Representative to the Board of the Winter Simulation Conference and is completing the second of a two year term as Board Chair of WSC.