Proceedings of the 1992 Winter Simulation Conference
ed. J. J. Swain, D. Goldsman, R. C. Crain, and J. R. Wilson

SIMULATION LANGUAGES AND SIMULATORS

Jerry Banks
School of Industrial and Systems Engineering
Georgia Tech
Atlanta, GA 30332-0205

ABSTRACT

This article describes current environments for popular simulation languages and simulators. Examples are given using one simulation language and one simulator. The selection of simulation software is discussed.

1 INTRODUCTION

Changes in simulation environments over the previous five years have occurred at a fast pace. Several factors contributed to these rapid changes. First, added computing power enables more extensive program features to be accomplished in a reasonable elapsed time. A second hardware feature is improved intelligence of the graphics display processor. Third, new operating systems allow more random access memory leading to larger simulation models. Fourth, relatively inexpensive hard disk subsystems have enabled larger operating systems and software packages of all type to be stored. Fifth, the advent of windows, from a general perspective, has created an awareness as to what might be accomplished to aid the simulation analyst. Sixth, there is a general trend in all software, simulation and otherwise, to be easier to use.

This paper has three sections beyond this introduction. In the following section, we describe existing software for several simulation languages and simulators. The next section contains an example problem solved using a simulation language and a simulator. The last section provides a number of features to consider when selecting simulation software.

2 SIMULATION SOFTWARE

Two categories of software will be discussed. The first category includes simulation languages. These are very flexible packages that can be used to model virtually all discrete systems. The second category consists of simulators. These software packages are basically simulation models that require parameterization by the simulation analyst. Some simulators allow programming to increase their flexibility. On the other hand, some of the simulation languages have features that make them more like simulators.

2.1 Simulation Languages

The simulation languages that will be discussed in this section include GPSS, SIMSCRIPT II.5®, SIMAN IV®, and SLAMSYSTEM®. Two implementations of GPSS will be described including GPSS/H™ and GPSS World™. The references for the languages are taken from the following sources: Brunner and Crain (1991) for GPSS/H, Cox (1991) for GPSS World, Russell (1991) for SIMSCRIPT II.5, Kasales and Sturrock (1991) for SIMAN IV, and Lilegdon (1991) for SLAMSYSTEM.

GPSS/H is a product of Wolverine Software Corporation, Annandale, VA. It provides improvements over GPSS V that had been released by IBM many years earlier. Some of these improvements include built-in file and screen I/O, use of an arithmetic expression as a block operand, interactive debugger, faster execution, expanded control statement availability, and ampervariables that allow the arithmetic combinations of values used in the simulation. The latest release of GPSS/H added a floating point clock, built-in math functions, and built-in random variate generators.

Versions of Wolverine GPSS/H that are currently available for the PC are GPSS/H Professional which is a large memory 32-bit version running under MS-DOS on 80386, 80386SX, and 80486 computers;

Student GPSS/H, a full-featured, limited-model-sized version of the language; and Personal GPSS/H limited by the 640K DOS memory ceiling.

GPSS/H is animated by Proof Animation (Brunner, Earle, and Henriksen, 1991). Proof Animation is a post-processor that can accept properly formatted data from any simulation output, so long as it is in ASCII.

GPSS World is a product of Minuteman Software, Stow, MA. GPSS World is a complete redesign of GPSS/PC™. There are five areas where the redesign is evident: windowing, virtual memory, the modeling language, animation, and hierarchical modeling. In the limited space available in this article, only a fraction of the changes can be described.

1. The new windowing features allow an arbitrary number of windows, of various types, to be opened on one or more simulations. Animations and windows may exist side-by-side.
2. Virtual memory essentially removes model and entity size restrictions.
3. The modeling language provides a set of built-in probability distributions and a range of system functions such as animation control. User defined subroutines can be incorporated into a program. Numerical integration is provided opening the way to continuous simulation.
4. The animation capabilities are enhanced by a separate program Simulation Studio™. It is basically a draw program allowing flexibility in zooming and panning, and in the transformation of objects.
5. The hierarchical modeling feature of GPSS World provides an ability to define submodels, which can be treated as if they were user defined GPSS Blocks. These abstractions can also be expanded in the other direction. This will promote either top-down or bottom-up modeling.

SIMSCRIPT II.5 is developed and marketed by CACI Products Company, La Jolla, CA. SIMSCRIPT II.5 is a powerful, highly portable, English-like, general purpose simulation programming language with built-in graphics and flexible statistical analysis. Through SIMSCRIPT II.5's "world view," founded on the basic concepts of entities and processes, programs are easily read and maintained. SIMGRAPHICS®, which comes standard with SIMSCRIPT II.5, allows the analyst to define menus, animated icons, and presentation output graphics with an interactive graphical editor.

SIMAN IV is a product of Systems Modeling corporation, Sewickley, PA. SIMAN IV is a general purpose program for modeling discrete and/or continuous systems. The framework distinguishes between the system model and the experimental frame. The system model defines components of the environment such as machines, queues, transporters and their interrelationships. The experimental frame describes the conditions under which the simulation is conducted including machine capacities and speeds, and types of statistics to be collected. "What-if" questions can usually be asked through changing the experimental frame rather than by changing the model definition. Some important aspects of SIMAN IV are as follows:

1. Special features that are useful in modeling manufacturing systems including the ability to decompose environments into workcenters, the ability to define a sequence for moving entities through the system, and the ability to schedule resource availability.
2. Constructs that enable the modeling of material handling systems including accumulating and non-accumulating conveyors, and transporters.
3. An interactive debugger that permits breakpoints, watches, and other execution control procedures.
4. The SIMAN IV environment that includes menu-driven "fill-in-the-form" editors for constructing system models and experimental frames, animation of the model using Cinema®, the Input Processor that assists in fitting distributions to data, and the Output Processor that can be used to obtain confidence intervals, histograms, correlograms, and so on.
5. Portability of the model across all types of computers.

SLAMSYSTEM is a product of Pritsker Corporation, Indianapolis, IN. SLAMSYSTEM is an integrated simulation system for loaded PCs based on the Microsoft® Windows™interface (under MS-DOS®) or the OS/2® Presentation Manager™. All features are accessible through pull-down menus and dialogue boxes, and are selected from the SLAMSYSTEM Executive Window. A SLAMSYSTEM project consists of one or more scenarios, each of which represents an alternative system configuration. A project maintainer examines the components of the current scenario to determine if any of them have been modified, indicates whether tasks such as model translation should be performed, and allows the user to accomplish these tasks before the next function is requested. Since SLAMSYSTEM is a Windows application, multiple tasks may be performed in parallel while the simulation is operating in the background.

Some of the features of SLAMSYSTEM are as follows:

1. Models may be built using a graphical network builder and a forms-oriented control builder, or text editor. Using the first method, a network symbol is selected with the mouse, then a form is completed specifying the parameters for that symbol. The Windows clipboard allows many other operations such as grouping one or more symbols and placing them elsewhere on the network.

2. Output analysis includes a "report browser" that allows alternative text outputs to be compared side-by-side. Output may be viewed in the form of bar charts, histograms, pie charts, and plots. Output from multiple scenarios can be displayed at the same time in bar chart form. Using the Windows environment, multiple output windows can be opened at the same time.

3. Animations are created under Windows using the Facility Builder to design the static background and the Script Builder to specify which animation actions should occur when a particular simulation event occurs. Animations can be performed either concurrently or in a post-processing mode. Two screens can be updated simultaneously and up to 225 screens can be swapped into memory during an animation.

4. SLAMSYSTEM was designed to be used in an integrated manner. For example, historic data may be read to drive the simulation. CAD drawings may be loaded. Output charts and plots created by SLAMSYSTEM may be exported via the Windows or Presentation Manager clipboard to other applications.

2.2 Simulators

The simulators that will be discussed in this section are limited to those associated with manufacturing and further to only three within that category. Those discussed are SIMFACTORY II.5®, ProModelPC®, and AutoMod2e. AutoMod2e could have easily been placed in the simulation language category. However, its emphasis on material handling leads to its placement in this category. The references for these simulators are taken from the following sources: Goble (1991) for SIMFACTORY II.5, Harrell and Tumay (1991) for ProModel, and *AutoMod User's Manual* (1992) for AutoMod2e.

SIMFACTORY II.5 is a product of CACI Products Company, La Jolla, CA. SIMFACTORY II.5 is a factory simulator written in SIMSCRIPT II.5. It was written for engineers that are not full time simulation analysts. A system amenable to SIMFACTORY II.5 can be modeled rapidly. A model is best constructed in stages by first defining the layout consisting of processing stations, buffers, receiving areas and transportation paths, defining the products, then the resources, next the transporters, and finally the interruptions. The animation automatically follows from the definition of the model.

The layout is created by positioning icons, selected from a library, on the screen. As each icon is positioned, characteristics describing it are entered. The products are defined by process plans that define the operations performed on each part and the duration of that operation. Resources are added to the model in two steps. First, the resource is defined and its quantity is set. Second, the stations requiring specified resources are identified.

Transporters may be batch movers such as forklifts or they may be conveyors. Characteristics of the transporter are specified (pickup speed, delivery speed, load time, unload time, and capacity for a fork lift, as an example). The transporter path is identified on the screen.

Interruptions are any activity that interferes with the operation of a station or a transporter. Equipment failures and preventive maintenance are two examples.

Reports are available concerning equipment utilization, throughput, product makespan, and buffer utilization. A summary report of all replications provides means, standard deviations, and confidence intervals on the model output.

ProModelPC is a product of PROMODEL Corporation, Orem, UT. ProModelPC is a manufacturing simulator that has programming features within the environment, and the capability to add C or Pascal type subroutines to a program. Some of the features of ProModelPC are as follows:

1. Most inputs are provided graphically, and intuitively.

2. Powerful manufacturing constructs are available including AGVs, conveyors, cranes, and robots.

3. ProModelPC operates in the Windows and OS/2 environments taking advantage of memory management techniques, synchronized windowing and data exchange.

4. Virtually unlimited model size is offered.

5. The simulator offers a 2-D graphics editor with scaling, rotating, and so on. CAD drawings can be imported.

6. The animation is developed while defining the model.

7. Business output graphics may be produced and

sent to a laser printer.

A model is constructed by defining a route for a part or parts, defining the capacities of each of the locations along the route, defining additional resources such as operators or fixtures, defining the transporters, scheduling the part arrivals, and specifying the simulation parameters. The software then prompts the user to define the layout and the dynamic elements in the simulation.

AutoMod²ᵉ is a product of AutoSimulations, Inc., Bountiful, UT. Its general programming features include the specification of processes, resources, loads, queues and variables. Processes are specified in terms of traffic limits, input and output connections, and itineraries. Resources are specified in terms of their capacity, processing time, MTBF, and MTTR. Loads are defined by their shape and size, their attributes, generation rates, generation limits, and start times, as well as their priority.

The simulator is very powerful in its description of material handling systems. AGVs, conveyors, bridge cranes, AS/RSs, and power and free devices can be defined. The range of definition is extensive. For example, an AGV can be defined in terms of the following:

- Multiple vehicle types
- Multiple capacity vehicles
- Path options (unidirectional or bidirectional)
- Variable speed paths
- Control points
- Flexible control and scheduling rules
- Arbitrary blocking geometries
- Automatic shortest-distance routing
- Vehicle procedures

Numerous control statements are available. For example, process control statements include if-then-else, while-do, do-until, wait-until, and wait for. Load control, resource control and other statements are also available. C functions may be defined by the user. Attributes and variables may be specified. The animation capabilities include true 3-D graphics, rotation and tilting, to mention a few. A CAD-like drawing utility is used to construct the model. Business graphics can be generated.

3 SOFTWARE EXAMPLES

In this section, an introductory problem is posed and solved using a simulation language and a simulator. The simulation language in which the problem is solved is GPSS/H and the simulator is ProModelPC. The problem does not tax the capability of either of these software packages in the slightest. The purpose is only to demonstrate the differences between the two types of simulation software. Using these two software packages in this tutorial is not an endorsement of either, as there are other software packages that can accomplish the same result. References for the two software packages are *ProModelPC User's Manual* (1991) and *Banks, Carson and Sy* (1989).

The problem statement is as follows: A production process consists of four operations: milling, planing, drilling, and inspection. There are four mills, two planers and three drills. The one inspection station is automated. Assume that there are movement times associated with each station that follow a triangular distribution with parameters 0.5, 1.0, 1.5 in minutes. Movement includes the following:

- Arrival to Milling or Arrival to Planing
- Milling to Drilling or Milling to Eject
- Planing to Drilling or Planing to Eject
- Drilling to Inspection or Drilling to Eject
- Inspection to Exit or Inspection to Drilling or Inspection to Eject

The data for arrivals, routing and processing is given in Table 1.

Table 1. Data for Arrival, Routing, and Processing

Activity	Distribution	Parameters
Arrivals	Exponential	Mean = 2
Milling	Triangular	8, 12, 14
Planing	Uniform	6, 10
Drilling	Normal	Mean = 5, Std. dev. = 1
Inspection	Uniform	1, 2
Route time	Triangular	.5, 1.0, 1.5

60% of the jobs are Type I and the remaining jobs are Type II. Type I jobs are served at milling, then move to drilling, and finally to inspection. Type II jobs go to planing, then drilling, and then to inspection. Jobs fail inspection with probability 0.10. 50% of the failed jobs are returned to drilling, and the remainder are scrapped.

There are finite buffers in front of the machines; the buffer sizes are 8 in front of mills, 4 in front of planers, 6 in front of drills, and 3 in front of inspection. When jobs can not join the buffer because it is full, they will be ejected from the system. Simulate the system for 40 hours with an 8-hour warmup. Collect statistics on average numbers in buffers, number ejected, utilization of machines, and throughput.

3.1 GPSS/H

The solution in GPSS/H is shown in Figure 1. There are two types of statements shown; GPSS/H blocks and GPSS/H control statements. The GPSS/H control statements used are SIMULATE, STORAGE, START, RESET, and END. These provide instruction to the software about how the simulation is conducted. For example, the RESET command erases the statistics for the first eight hours (480 minutes) of simulated time. As another example, the STORAGE control statement establishes capacities for buffers and machines.

The GPSS/H blocks begin with GENERATE and continue through the last TERMINATE. They describe how an entity, or transaction, moves through the system. A combination of blocks provides the "logic" of the simulation model. The initial blocks in the program are described as follows:

```
GENERATE    RVEXPO(1,2.0)
```

Creates transactions according to an exponential distribution with a mean interarrival time of 2 units (minutes, in this model) using random number stream 1.

```
ADVANCE     RVTRI(1,.5,1.0,1.5)
```

Routes the transactions from entry to the next location according to a triangular distribution with parameters .5, 1.0, and 1.5 using random number stream 1.

```
TRANSFER    .6,TYPE2,TYPE1
```

Sends 60% of the transactions to location TYPE1 and 40% of the transactions to location TYPE2, at random.

```
TYPE1  TEST LE   S(BUFFMILL),7,EJECT
```

At location TYPE1 a test is conducted to determine if the number in the buffer before the mill, BUFFMILL, is less than or equal to 7. If the result is true, the transaction goes to the next sequential block. If the result is not true, the transaction goes to the location called EJCT.

```
        SIMULATE
        STORAGE     S(BUFFMILL),8/_
                    S(MILL),4/_
                    S(BUFFPLAN),4/_
                    S(PLANE),2/_
                    S(BUFFDRIL),6/_
                    S(DRILL),3/_
                    S(BUFFINSP),3
        GENERATE    RVEXPO(1,2.0)
        ADVANCE     RVTRI(1,.5,1.0,1.5)
        TRANSFER    .6,TYPE2,TYPE1
TYPE1   TEST LE     S(BUFFMILL),7,EJCT
        ENTER       BUFFMILL
        ENTER       MILL
        LEAVE       BUFFMILL
        ADVANCE     RVTRI(1,8.0,12.0,14.0)
        LEAVE       MILL
        TRANSFER    ,DRLA
EJCT    ADVANCE     RVTRI(1,.5,1.0,1.5)
        TERMINATE
TYPE2   TEST LE     S(BUFFPLAN),3,EJCT
        ENTER       BUFFPLAN
        ENTER       PLANE
        LEAVE       BUFFPLAN
        ADVANCE     8,2
        LEAVE       PLANE
        TRANSFER    ,DRLA
DRLA    ADVANCE     RVTRI(1,.5,1.0,1.5)
        TEST LE     S(BUFFDRIL),5,EJCT
        ENTER       BUFFDRIL
        ENTER       DRILL
        LEAVE       BUFFDRIL
        ADVANCE     RVNORM(1,5.0,1.0)
        LEAVE       DRILL
        ADVANCE     RVTRI(1,.5,1.0,1.5)
        TEST LE     S(BUFFINSP),2,EJCT
        ENTER       BUFFINSP
        SEIZE       INSP
        LEAVE       BUFFINSP
        ADVANCE     2.0,1.0
        RELEASE     INSP
        TRANSFER    .9,NOGD,GOOD
NOGD    TRANSFER    .5,DRLA,SCRAP
GOOD    ADVANCE     RVTRI(1,.5,1.0,1.5)
        TERMINATE
SCRAP   ADVANCE     RVTRI(1,.5,1.0,1.5)
        TERMINATE
        GENERATE    480
        TERMINATE   1
        START       1
        RESET
        START       5
        END
```

Figure 1. Solution to example problem using GPSS/H

```
ENTER       BUFFMILL
```

The transaction enters the storage designated as BUFFMILL.

3.2 ProModelPC

The solution in ProModelPC is shown in Figure 2. It should be noted that Figure 2 is generated entirely on the basis of prompts from the screens in the software. The ASCII code is available to the user and that is what is shown in Figure 2. In this elementary model, the software first prompts for the routing of the parts through the system. The parts are indicated as T1 for Type 1, T2 for Type II, and ANY indicating that both T1 and T2 follow the logic indicated. Under "Condition" the probabilities sum to 1000 and ALT means alternately go to the location indicated.

The next prompt from the screen is for part scheduling at location ENTRY. In the example, the model will be stopped after the appropriate time has elapsed, so the number of arrivals is set artifically high.

Next, the software asks about capacities. ENTRY is set to 100 as it is not specified in the problem statement and 100 will never be reached.

The simulation parameters provide for the 8-hour warmup followed by a 40-hour run of the model. The report code requests all of the standard outputs.

The graphic options are indicated for the type of monitor that is being used. The default screen color is D for dark and the default figure color is W for white. There are icons available within the software and these are stored in the filename PM.LIB.

PM.LIB contains many standard figures that are referenced in the next section. For example, the static symbol 3 used at ENTRY is a square with rounded corners. In this model, it will be green, and it will be placed at row 25 and column 10. Again, the user enters all of this information through prompts, and places the static symbol on the screen at the desired location.

The dynamic symbols are those that move around the static background. The symbols are referenced by ";" and other characters, although these are not what appear on the screen. For example, a static symbol may be a diamond, whose color is indicated, i.e., W for white and Y for yellow.

4 SELECTION OF SIMULATION SOFTWARE

The features to be considered in selecting simulation software can be classified in the categories of input processing, output, support, and cost. These features were discussed in a recent tutorial (Banks, 1991). In this section, a condensation of the first three features is shown in Tables 2 through 4, and a brief discussion of the last two features is presented.

Table 2: Input Features

Feature	Descriptor
Interface with other languages	Accept C, Pascal, etc.
Input data analysis capability	Empirical or mathematical description of raw data
Portability	Run same model on workstations, PCs, etc.
Syntax	Easily understood, consistent, unambigous
Input flexibility	Accepted both interactively and in batch form
Interactive debugger	Break, trap, run until, step
Modeling flexibility (simulation languages)	Process interaction, event, continuous perspectives
Modeling conciseness (simulation languages)	Powerful blocks or nodes

Table 3: Processing Features

Feature	Descriptor
Execution speed	Many scenarios, many replications to be simulated
Model size	A concern under plain DOS
Material handling	Transporters, conveyors, AGVs, etc.
Random variate generators	Exponential, normal, etc.
Reset capability	For steady state analysis
Independent replications	Using a different set of random numbers
Attributes	Local values
Global values	Data available to all entities
Programming (simulators)	Incorporate special characteristics
Conditional routing (simulators)	Send entities based on prescribed conditions

ROUTING

Part	Location	Operation (min)	Output part	Next location	Condi-tion	Qty	Move time (min)
ANY	ENTRY	0	T1	MILLQ	600	1	T .5,1,1.5
			T1	SCRAP	ALT	0	T .5,1,1.5
			T2	PLANEQ	400	0	T .5,1,1.5
			T2	SCRAP	ALT	0	T .5,1,1.5
T1	MILLQ	0	T1	MILL	0	1	0
T1	MILL	T 8,12,14	T1	DRILLQ	0	1	T .5,1,1.5
			T1	SCRAP	ALT	0	T .5,1,1.5
T2	PLANEQ	0	T2	PLANE	0	1	0
T2	PLANE	U 8,2	T2	DRILLQ	0	1	T .5,1,1.5
			T2	SCRAP	ALT	0	T .5,1,1.5
ALL	DRILLQ	0	ALL	DRILL	0	1	0
ALL	DRILL	N 5,1	ALL	INSPQ	0	1	T .5,1,1.5
			ALL	SCRAP	ALT	0	T .5,1,1.5
ALL	INSPQ	0	ALL	INSP	0	1	0
ALL	INSP	U 1.5,.5	ALL	EXIT	900	1	0
			ALL	DRILLQ	50	0	T .5,1,1.5
			ALL	SCRAP	ALT	0	T .5,1,1.5
			ALL	SCRAP	50	0	T .5,1,1.5
ALL	SCRAP	0	ALL	EXIT	0	1	0

PART SCHEDULING

Part	Location	Qty per arrival	No. of arrivals	Start (min)	Arrival frequency (min)
ANY	ENTRY	1	9999	0	E 2.0

CAPACITIES

Resource	Qty	Resource	Qty	Resource	Qty	Resource	Qty	Resource	Qty
DRILL	3	DRILLQ	6	ENTRY	100	INSP	1	INSPQ	3
MILL	4	MILLQ	8	PLANE	2	PLANEQ	4	SCRAP	1

SIMULATION PARAMETERS

Run (hrs)	Startup (hrs)	Rept code	Resource to track
40	8	5	0

GRAPHIC OPTIONS

Seed	Graph mode	Max row	Max col	Scr clr	Fig clr	Icon file
0	3	100	200	D	W	PM.LIB

STATIC SYMBOLS

ID	Sym	Clr	Row	Col	ID	Sym	Clr	Row	Col
ENTRY	3	G	25	10	MILLQ	ZR	C	19	20
MILL	a	R	19	31	PLANEQ	ZR	M	28	20
PLANE	g	O	28	26	DRILLQ	ZR	W	25	40
DRILL	i	2	25	48	INSPQ	ZR	W	25	54
INSP	5	M	25	61	SCRAP	4	R	30	70

DYNAMIC SYMBOLS

ID	Sym	Clr	ID	Sym	Clr	ID	Sym	Clr
ANY	;	W	T1	,	Y	T2	/	W

Figure 2. Solution to Example Problem Using ProModelPC

Table 4: Output Features

Feature	Descriptor
Standarized reports	Average number in queue, etc.
Customized reports	Tailored presentation for managers
Statistical analysis	Confidence intervals, etc.
Business graphs	Bar charts, pie charts, etc.
File creation	Input to spreadsheet
Data base maintenance	Collect output in an organized manner

Prior to ascertaining the features of simulation software, it is appropriate to determine what is required for the application being considered. For example, it is not necessary to ask for 3-D animation unless it is needed. Another consideration is that judgements should not be made on the basis of "yes" and "no" answers. For example, the question may be asked as to whether the software can accept general programming code. The more important consideration is whether the software can avoid having to use such code.

The environment features include ease of use and ease of learning the software. The quality of documentation is also a consideration. Animation capability is a feature that includes ease of development, quality of the picture on the screen, smoothness of movement, portability for remote viewing, and the availability of a CAD interface. Law and McComas (1992) expand on animation capability in a recent article. Also of importance are on-line help and an on-line tutorial. Customer support is essential including training, technical support, updates and enhancements.

The cost feature is a difficult one as the software can vary from as little as $1,500 to as much as $80,000, and a $15,000 package may not have three times the utility of a $5,000 package. Included in the cost should be the hardware requirements if a computer is dedicated to simulation alone, the amount of learning needed to use the software, and the amount of time needed for model building.

Some final pointers in selecting simulation software are the following:

1. You may need more than one package. Both a simulator and a simulation package may be necessary.
2. Get the greatest power that you can afford. Having simulation analysts wait is extremely expensive.
3. Beware of fancy ads and demos. These may not reflect the ability to solve your problem.
4. Beware of checklists. Implementation and capability are what is important.
5. Obtain a trial copy. Try solving a small version of your problem.
6. Ask the vendor to solve a sample problem. Avoid purchasing software that won't solve your problem.

REFERENCES

AutoMod User's Manual. 1992. AutoSimulations, Bountiful, UT.

Banks, J. 1991. Selecting simulation software. *Proceedings of the 1991 Winter Simulation Conference*, eds., B.L. Nelson, W.D. Kelton, G.M. Clark, 15-20, Institute of Electrical and Electronics Engineers, Piscataway, NJ.

Banks, J., J.S. Carson, II, and J.N. Sy. 1989. *Getting started with GPSS/H*, Wolverine Software Corporation, Annandale, VA.

Brunner, D.T., and R.C. Crain. 1991. GPSS in the 1990s. In *Proceedings of the 1991 Winter Simulation Conference*, eds., B.L. Nelson, W.D. Kelton, G.M. Clark, 81-85, Institute of Electrical and Electronics Engineers, Piscataway, NJ.

Brunner, D.T., N.J. Earle, and J.O. Henriksen 1991. Proof animation: The general purpose animator. In *Proceedings of the 1991 Winter Simulation Conference*, eds., B.L. Nelson, W.D. Kelton, G.M. Clark, 90-94, Institute of Electrical and Electronics Engineers, Piscataway, NJ.

Cox, S.W. 1991. GPSS World: A brief preview. In *Proceedings of the 1991 Winter Simulation Conference*, eds., B.L. Nelson, W.D. Kelton, G.M. Clark, 59-61, Institute of Electrical and Electronics Engineers, Piscataway, NJ.

Goble, J. 1991. Introduction to SIMFACTORY II.5. In *Proceedings of the 1991 Winter Simulation Conference*, eds., B.L. Nelson, W.D. Kelton, G.M. Clark, 77-80, Institute of Electrical and Electronics Engineers, Piscataway, NJ.

Harrell, C.R., and K. Tumay. 1991. ProModel tutorial. In *Proceedings of the 1991 Winter Simulation Conference*, eds., B.L. Nelson, W.D. Kelton, G.M. Clark, 101-105, Institute of Electrical and Electronics Engineers, Piscataway, NJ.

Kasales, C.J. and D.T. Sturrock. 1991. Introduction to SIMAN IV. In *Proceedings of the 1991 Winter Simulation Conference*, eds., B.L. Nelson, W.D. Kelton, G.M. Clark, 106-111, Institute of Electrical and Electronics Engineers, Piscataway, NJ.

Law, A.M. and M[illegible]nos. 1992. How to select simulation software for manufacturing applications. *Industrial Engineering* 24: 29-35.

Lilegdon, W.R. 1991. Introduction to SLAMSYSTEM. In *Proceedings of the 1991 Winter Simulation Conference*, eds., B.L. Nelson, W.D. Kelton, G.M. Clark, 118-121, Institute of Electrical and Electronics Engineers, Piscataway, NJ.

ProModelPC User's Manual. 1991. PROMODEL Corporation, Orem, UT.

Russell, E.C. 1991. SIMSCRIPT II.5 and MODSIM II: A brief introduction. In *Proceedings of the 1991 Winter Simulation Conference*, eds., B.L. Nelson, W.D. Kelton, G.M. Clark, 62-66, Institute of Electrical and Electronics Engineers, Piscataway, NJ.

AUTHOR BIOGRAPHY

JERRY BANKS is Associate Professor in the School of Industrial and Systems Engineering at the Georgia Institute of Technology. His research interests include knowledge-based on-line simulation and the verification and validation of large scale simulation models. He is completing his last period of service on the Board of the Winter Simulation Conference and served as Chair of that body for two years. He is Chairman of Carson/Banks & Associates, Inc., a simulation consulting firm.