Proceedings of the 2003 Winter Simulation Conference
S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice, eds.

TEACHING DISCRETE EVENT SIMULATION TO BUSINESS STUDENTS: THE ALPHA AND OMEGA

Richard G. Born

Department of Operations Management and Information Systems
College of Business
Northern Illinois University
DeKalb, IL 60115, U.S.A.

ABSTRACT

Managers of businesses worldwide are only beginning to realize the economic and improved decision-making value of discrete-event simulation. In order to accelerate the rate at which business managers employ simulation, such a course needs to be taught to more business students than is currently being done. This, in turn, implies the need for an improvement in the teaching of simulation to beginners, so that these fledglings will encourage fellow students to take a course in simulation because it provides business value, practicality, and promotes the idea that simulation is fun. The manner in which simulation is introduced during the first week of class as well as how the course is summed up during the last week of class are, perhaps, the most critical points in student learning. This paper, therefore, focuses on activities that the author uses during the first and last weeks of his simulation courses for business students.

1 SETTING THE STAGE

My experience with simulation modeling dates back to 1985 when I was in the process of preparing my dissertation, which involved simulation of message transfer in a variety of very-large-scale interconnection networks including the n-dimensional hypercube. The simulation models of these systems were written in GPSS V and typically took about an hour of CPU time on an IBM 370 computer. Hence, I have been using one of the "classics" in simulation modeling for nearly a score of years.

Upon receiving my doctorate in Computer Science in 1988, I became a member of the faculty of the Department of Operations Management and Information Systems in the College of Business at NIU (Northern Illinois University), where I have been ever since. Keeping in the back of my mind the excitement I felt in working with GPSS, I offered a computer simulation course for our M.S. in MIS (Management Information Systems) majors for the first time in 1991. This course has made primary use of micro-GPPS through the fall semester of 2001, and has been using WebGPSS, the successor of micro-GPSS, since the spring semester of 2002. An undergraduate version of the course was also added to the curriculum in the mid-1990s, and it is estimated that approximately 1000 business students have learned simulation modeling at NIU as a result.

My selection and use of micro-GPSS was a decision that was made after attending WSC 1991, where I was introduced to micro-GPSS in a small vendor booth with no bells and whistles, just a man, Ingolf Ståhl, and his computer. I was impressed about his concern for making simulation modeling something that could be learned by business students as well as his emphasis on quality teaching and the learning of simulation through building models of a wide variety of business systems.

In addition to the use of WebGPSS, business students at NIU are also introduced to ProModel, a popular simulation modeling and animation software product designed primarily for manufacturing and logistics. Although the NIU business student works only with ProModel, the student quickly realizes that many additional business applications can be simulated using ProModel Corporation's MedModel and ServiceModel.

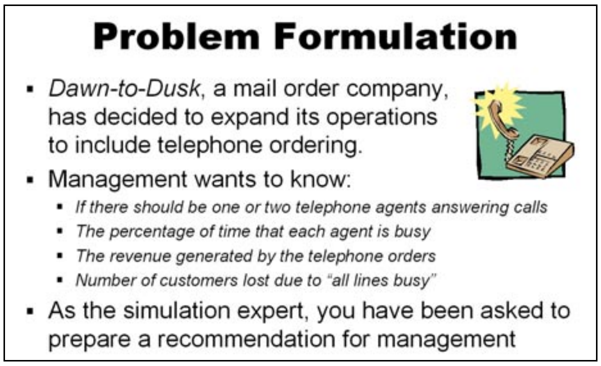
How can the use of WebGPSS and ProModel be tied together in an introductory course in discrete-event simulation for business students? The author has done this by spending the first week of the semester – the alpha – discussing neither of these simulation products. Instead, the first week is spent discussing the entire simulation modeling process by solving a typical business problem using the simplest of simulation technologies – a pair of ordinary six-sided dice. Then, the last week of the semester – the omega, after the student has learned the fundamentals of both WebGPSS and ProModel, the class returns to this business problem to see how it can be solved using both WebGPSS and ProModel. The alpha and the omega are united, and the simulation course becomes a whole.

2 ALPHA: THE FIRST WEEK

While we could begin our discussion of discrete-event simulation with some carefully worded definitions of simulation, it seems much more appropriate in a class of business students to take a more hands-on approach. Many of these students are ultimately going to work in areas such as finance, marketing, accounting, and operations in a wide variety of industries upon graduation. They are continually being told by recruiters that firms want individuals who are good business problem solvers. So we begin our study of simulation by doing our best to solve a typical business problem. Any reasonable business problem that lends itself to a simulation solution will do. I will describe a specific problem that I have used to introduce simulation modeling by presenting a series of PowerPoint slides that I use during the class discussion of the problem.

2.1 Problem Formulation

I begin by suggesting to the class that a reasonable simulation solution could be seen as involving five steps: problem formulation, data collection and analysis, model development, model experimentation, and output analysis. One of the first slides is used to discuss and formulate the problem, as shown in Figure 1.



Problem Formulation

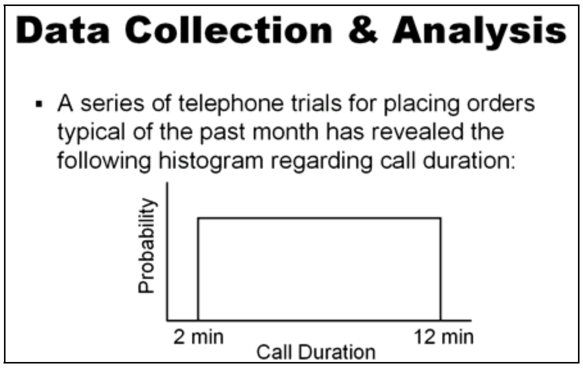
- *Dawn-to-Dusk*, a mail order company, has decided to expand its operations to include telephone ordering.
- Management wants to know:
  - *If there should be one or two telephone agents answering calls*
  - *The percentage of time that each agent is busy*
  - *The revenue generated by the telephone orders*
  - *Number of customers lost due to "all lines busy"*
- As the simulation expert, you have been asked to prepare a recommendation for management

Figure 1: Problem Formulation

A study of Figure 1 indicates that we have introduced a problem that will involve an experiment in which the number of telephone agents is the experimental variable. The concept of utilization is introduced when discussing the fact that management wants to know the percentage of time that each agent is busy. The students are also asked to suggest reasons why management would like to know the utilization of its telephone agents. Clearly, revenue generated is of interest to the Dawn-to-Dusk CEO, and knowing the number of customers that are lost because all agents are busy impacts both revenue and the quality of customer service. Referring to the students as the simulation experts during the first week of class gives them a very positive feeling about their upcoming journey in simulation modeling.

2.2 Data Collection and Analysis

Having formulated the problem, we next look at issues associated with data collection and analysis. There are three kinds of data that need to be collected to drive the simulation: call duration data, call inter-arrival time data, and invoice total for each call placed. Figure 2 shows one approach to obtaining data on call duration. The students get a feel for one technique that could be used to obtain this data empirically and are introduced to the uniform or rectangular distribution.

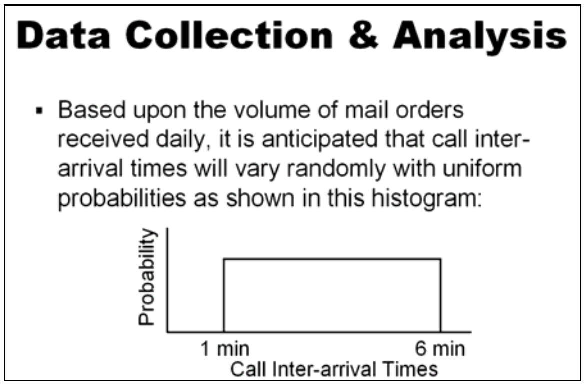


Data Collection & Analysis

- A series of telephone trials for placing orders typical of the past month has revealed the following histogram regarding call duration:

Figure 2: Call Duration Data Collection and Analysis

Figure 3 shows how data was collected in order to determine the inter-arrival time for calls. Here, corporate records were reviewed for the daily volume of existing mail orders, and some assumptions are made regarding the percentage of mail orders that would convert to orders placed by telephone.

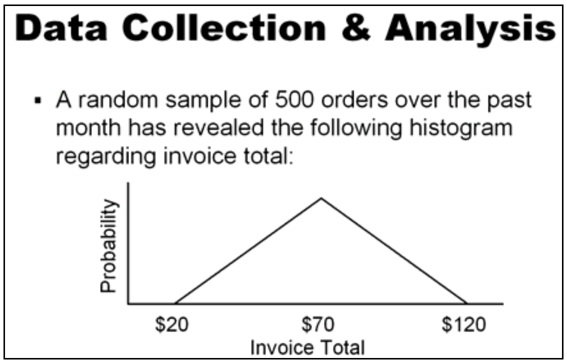


Data Collection & Analysis

- Based upon the volume of mail orders received daily, it is anticipated that call inter-arrival times will vary randomly with uniform probabilities as shown in this histogram:

Figure 3: Call Inter-Arrival Data Collection and Analysis

Figure 4 indicates that a random sample of orders placed over the past month was used to obtain information regarding invoice total. The students become acquainted with the fact that there are random distributions that are not uniformly distributed, and are here introduced to the triangular distribution. They notice that the invoice total varies from a low of \$20 to a high of \$120, peaking at \$70. We



Data Collection & Analysis

- A random sample of 500 orders over the past month has revealed the following histogram regarding invoice total:

Figure 4: Revenue Data Collection and Analysis

can also introduce the student at this time to the terms *symmetric* and *asymmetric*.

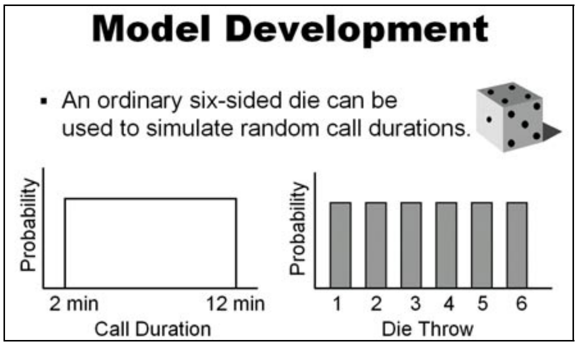
2.3 Model Development

Having completed the data collection and analysis phase, we then move on to model development. We plan to use a very simplistic and old technology, namely dice, to produce our random numbers. Figure 5 shows how we can use an ordinary six-sided die to simulate the random call durations. Students are generally quick to point out that they can simply double the number on the die to get a call duration that varies between 2 and 12 minutes. If the students are prompted by asking a question such as "Do you see any problem with this?", someone will usually point out that we will never get call durations that are an odd number of minutes. But this then motivates a discussion as to whether or not our model results will be significantly affected because we don't produce call durations with an odd number of minutes. We can then get into a discussion that the production of even call durations only is more of a limitation of the technology or how we are using the technology, and that this limitation need not exist later in the course, when we use the computer to produce our random call durations. With some additional thought on the use of a die, we could possibly come up with a scheme for representing odd call duration times, but would this extra effort on our part significantly change any recommendation that might be made to management? The student quickly learns of tradeoffs between producing an exact representation of the real system and our somewhat less detailed model.

Figure 6 shows how we can simulate the call interarrival time (IAT), also by the use of an ordinary six-sided die. Here, the conversion from a die throw to an IAT is trivial, as we just use the value of the die throw as the IAT. I find at this point that a student in the class will typically bring up a concern that the throw of the die will always be an integer between 1 and 6, but that in the real telephone ordering system, IATs such as 2.27 and 5.44 minutes are certainly quite feasible. This leads to a discussion of the difference between discrete and continuous distributions.

Figure 7 shows how we can the simulate invoice total for each order. A student is generally quick to suggest that the invoice total can be found by multiplying the sum of the two dice by 10. A discussion also evolves as to why the probability of the various sums of the dice from 2 to 12 vary in a triangular fashion. At this point, I admit to the class that the data has been contrived to simplify mental calculations and the recording of data later during model experimentation, but that this in no way invalidates what the class is learning about the steps involved in a simulation project.

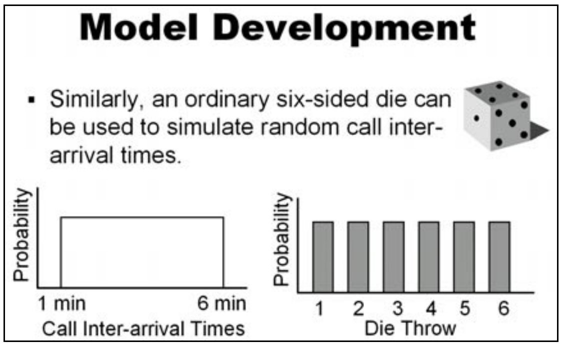
At this point in the model development, we begin considering some of the physical flow considerations for our simulation, as depicted in Figure 8. We assume that if all



Model Development

- An ordinary six-sided die can be used to simulate random call durations.

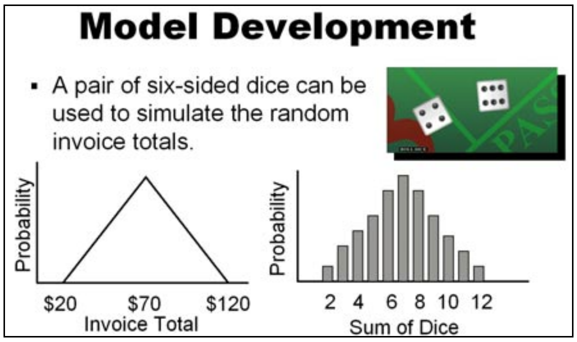
Figure 5: Model Development – Call Duration



Model Development

- Similarly, an ordinary six-sided die can be used to simulate random call inter-arrival times.

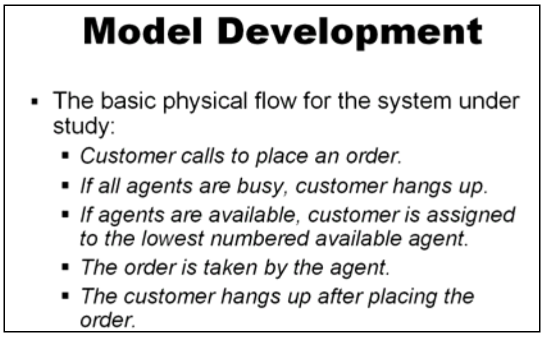
Figure 6: Model Development – Inter-Arrival Times



Model Development

- A pair of six-sided dice can be used to simulate the random invoice totals.

Figure 7: Model Development – Invoice Totals



Model Development

- The basic physical flow for the system under study:
  - *Customer calls to place an order.*
  - *If all agents are busy, customer hangs up.*
  - *If agents are available, customer is assigned to the lowest numbered available agent.*
  - *The order is taken by the agent.*
  - *The customer hangs up after placing the order.*

Figure 8: Model Development – Physical Flow

agents are busy, then the customer will hang up and not call back later. We could allow callback after a predetermined period of time, and we could even allow for call queuing, but agree for the sake of simplicity not to model either of these things at this time. I point out, however, that callback as well as call queuing would be quite straight forward with our computer simulation software. We note that, as a tie breaker, a customer calling in will be assigned to the lowest numbered available agent.

Finally, with regard to model development, we agree to simulate the telephone ordering system for one hour of system operation. We further realize that we need to be sure to collect data that will allow tracking of the following items:

- Number of orders placed
- The revenue generated by each telephone order
- The number of calls "lost" because all agents are busy
- The utilization of each agent.

2.4 Model Experimentation

We are finally ready to do the simulation experiment. The class does the experiment once together as a group along with the instructor to make sure that everyone understands how to go about collecting the data during the experiment. Data sheets are provided for each student to record results while doing the experiment, as shown in Figures 9 and 10.

The class has no trouble understanding why they need to do two experiments, i.e., one experiment in which there is one agent answering calls and one in which there are two agents. But then I point out that we need to do both experiments concurrently as we roll the dice; we cannot just roll the dice while doing experiment one, and then continue to roll the dice while we perform the second experiment. When asked why, someone in the class usually responds by saying that we want to be sure that we have the same sequence of call arrivals, call durations, and invoice totals for both experiments. A good discussion of the need to control variables in an experiment then follows.

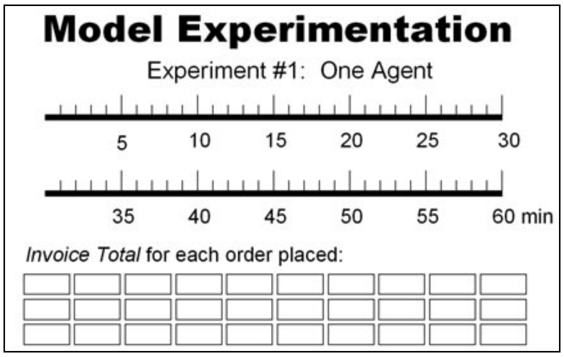


Figure 9: Model Experimentation – One Agent

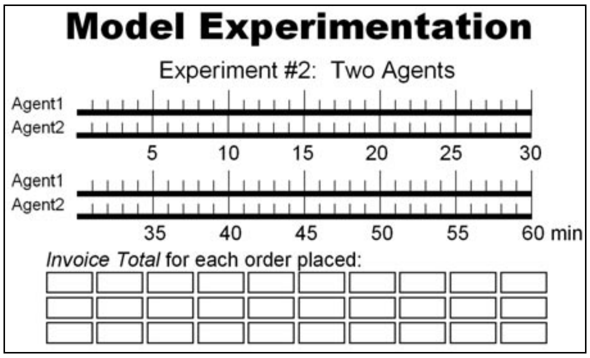
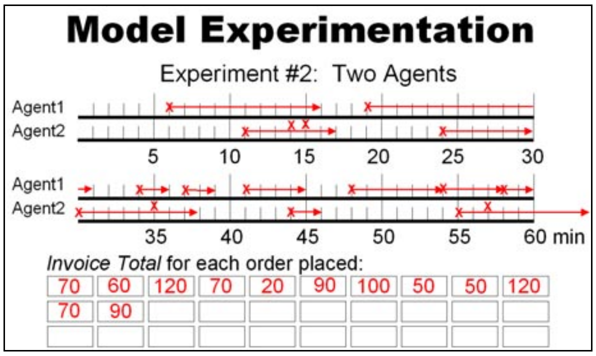


Figure 10: Model Experimentation – Two Agents

As the die/dice are rolled, the class agrees to indicate call arrivals by marking an X at the appropriate place on the timeline. The filled in data sheet for the experiment with two agents is shown in Figure 11. An arrow whose length indicates the random call duration is then drawn to the right of the X. If the call is a lost call, the X will appear without the arrow to its right. The invoice total for each of the orders is recorded in a little while box at the bottom of the data sheet. When getting near the end of the experiment, the students often find that the arrow goes beyond the 60 minutes, meaning that the call is not completed during the hour simulated. They then need to agree on how to handle the statistics for this particular call. Finally, they realize that the experiment has ended when the

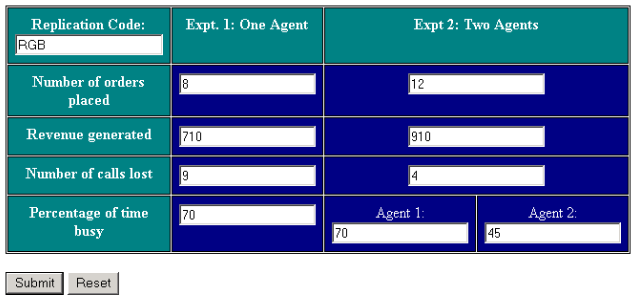


70	60	120	70	20	90	100	50	50	120
70	90								

Figure 11: Example Data Sheet for Two Agent Experiment

random throw of the IAT die would place a call after an hour of time has elapsed.

Having completed the experiment as a class, each student is then given a pair of dice to take home. They are asked to perform the experiment again on their own and submit the results in a summary form provided on the web before the next class meeting. An example filled out summary form, based in part upon the data in Figure 11, is shown in Figure 12. The replication code is the student's initials, serving to identify that they have submitted the data as well as provide the source in the event that there is a problem with the data they have submitted.



Replication Code: RGB	Expt. 1: One Agent	Expt 2: Two Agents	
Number of orders placed	8	12	
Revenue generated	710	910	
Number of calls lost	9	4	
Percentage of time busy	70	Agent 1: 70	Agent 2: 45

Figure 12: Summary Form Ready to Submit Via the Web

The students are told that the data they submit will be imported into a spreadsheet that computes means and confidence intervals for the various items of interest to management. This spreadsheet will be used in the next class meeting as the basis of the output analysis phase of the simulation modeling process.

2.5 Output Analysis

Summary statistics for a class of 32 students who submitted their replication results are shown in Figure 13. One of the first questions I ask is "What general observations can you make by observing the minimum and maximum values for each of the output variables?" Students point out that there is quite a large range between the minimum and maximum for each output variable, and there is significant overlap in these ranges for the experiments for

	Expt 1: Orders Placed	Expt 2: Orders Placed	Expt 1: Revenue Generated	Expt 2: Revenue Generated	Expt 1: Calls Lost	Expt: 2 Calls Lost	Expt 1: % Time Busy	Expt 2: Agent 1 % Busy	Expt 2: Agent 2 % Busy
MINIMUM	3	5	230	300	4	0	63.33	58.33	28.33
MAXIMUM	9	13	580	920	13	8	90	92	88.3
n	32	32	32	32	32	32	32	32	32
Average	5.88	10.34	406.25	690.00	8.31	3.63	76.91	76.88	58.35
Variance	1.40	2.68	7759.68	20929.03	5.00	3.08	44.73	48.51	145.23
% Conf. Int.	90	90	90	90	90	90	90	90	90
alpha	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10
d.f.	31	31	31	31	31	31	31	31	31
critical t-value	1.6955	1.6955	1.6955	1.6955	1.6955	1.6955	1.6955	1.6955	1.6955
Confidence									
Interval									
Lower Limit	5.52	9.85	379.85	646.64	7.64	3.10	74.91	74.79	54.73
Upper Limit	6.23	10.83	432.65	733.36	8.98	4.15	78.92	78.97	61.96

Figure 13: Statistics for Output Analysis

one and two agents answering calls. These observations by the students motivate a discussion that centers around the fact that one should not, in general, in simulation studies, base any recommendations to management on a single replication. This helps to emphasize the need for computation of means and confidence intervals for a reasonable number of replications.

Then we address the question "So what are you going to tell management as a result of your simulation study?" Typical student responses include the following:

- There is about an 80% increase in the number of orders placed when the number of agents goes from one to two.
- There is about a 70% increase in the total revenue generated when the number of agents increases from one to two.
- There is about a 60% decrease in the number of calls lost when the number of agents changes from one to two. The use of two agents could significantly increase the quality of service and the image of the company by reduction of lost calls.
- Agent 1 will be busy about 75% of the time, whereas agent 2 will only be busy about 55% of the time. Management may wish to consider assigning additional tasks to agent 2 to keep that agent busier throughout the day.

Finally, I remind the class that the simulation was run for one hour of simulated time and then ask "Are there any quantitative dollar figures that you could provide management regarding the benefits of two agents over one when considering a 160-hour work month?" After a little calculation, someone would tell that class that two agents result in about $43,000 more revenue per month than would be obtained with one agent.

2.6 Summary of The Alpha

In about 100 total minutes of classroom time, the stage for the rest of the semester has been set. The students have gone through an entire simulation project from problem formulation through output analysis. They have learned many things about simulation modeling that will aid their understanding of the computer-based software they will be studying during the bulk of the semester. Having noted the arrival times for calls, they have gained a feeling for the concept of *discrete-event* simulation, and how the simulation clock "jumps" from one time to the next. Having done the detailed calculations required to compute things like agent utilization and revenue collected, they have gained an appreciation for the fact that the computer-based software they will be studying can do all of these calculations automatically for them. Having spent upwards of 15 minutes at home doing the experiments using dice, they will truly appreciate the speed of the computer in generating pseudo-random numbers to accomplish the same result.

Having gone through the details of recording events on a timeline for a relatively simplistic physical flow, they will appreciate the fact that the computer-based simulation software they will soon learn can easily account for more complex scenarios including call queuing or callback. And everything they have learned during these short 100 minutes is completely independent of the simulation software they will be studying.

3 OMEGA: THE LAST WEEK

Having spent the bulk of the semester learning both WebGPSS (Ståhl 2002, and Born and Ståhl 2002) and ProModel (ProModel 2002), the students are now ready to return to the Dawn-to-Dusk telephone ordering system. This time they are asked to design simulation solutions using each of these software packages. Their solutions will draw heavily on many of the specific concepts they have learned about each of these packages during the course of the semester. While there are numerous ways for developing a solution to the Dawn-to-Dusk telephone ordering system for each of these packages, we shall here present a typical solution for each.

3.1 A WebGPSS Solution

The WebGPSS solution shown in Figure 14 was designed so that it would be easy to perform experiments by using the *Experiment* dialogue box provided with WebGPSS. This dialogue box allows one to specify the experimental variable, the result variable, the number of values the experimental variable is to take on, its lowest and highest value, and the number of simulation runs. If the number of values is specified as 1, then WebGPSS will compute a 95% confidence interval for the result variable. If the number of values is specified as 2, then WebGPSS will perform a pair-wise comparison. If the number of values exceeds 2, then WebGPSS will allow optimization of the result variable. To aid in explaining the solution shown in the block diagram of Figure 14, each block has been numbered for ready reference.

- ***Blocks 1, 2, and 3:*** *FN$IATDIE*, *FN$DURDIE*, *FN$INDIE1*, and *FN$INDIE2* are all random discrete functions that produce integers between 1 and 6, inclusive, and are used for creating random inter-arrival times, call durations, and invoice totals. The holding time and invoice total for each call are stored in parameters *p$hold* and *p$invtot*, respectively.
- ***Blocks 4, 14, 15 and 25:*** In block 4, *agent1* is checked for being busy. In block 14, the number of agents is checked. If there is only one agent, then the call is lost and leaves the system in block25. The call reaches block 15 only in the event that *agent1* is busy and there are two agents taking calls. Here, *agent2* is checked for being busy.

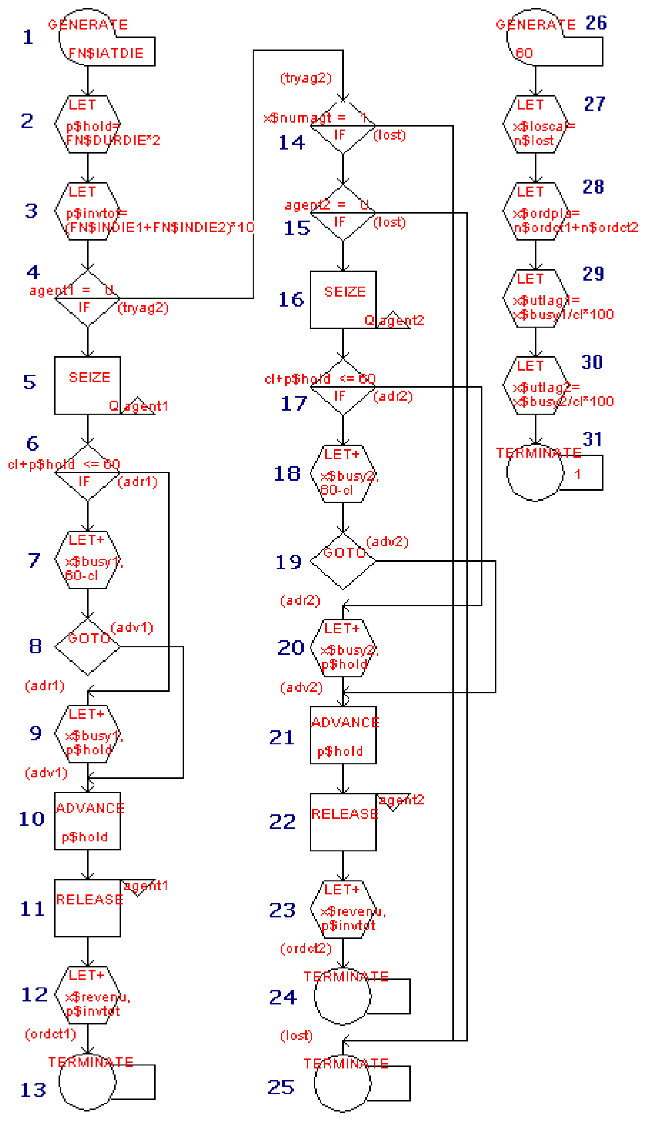
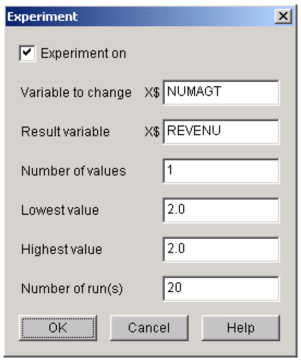


Figure 14: WebGPSS Block Diagram for Dawn-to-Dusk Telephone Ordering System

- ***Blocks 5 through 13:*** Blocks 5 through 13 represent the processing of the call by *agent1*. Blocks 16 through 24 represent the processing of the call by *agent2* and are similar to the call processing by *agent1*. We, therefore, discuss only blocks 5 through 13. In block 5, the call seizes *agent1*. In blocks 6 through 9 the total amount of time that *agent1* has been busy, *x$busy1*, is updated. Here, we must be careful to consider the case where the call would go beyond the 60 minutes run time for the simulation. In this case, the busy time for *agent1* is increased by *60-p$hold* and not *p$hold*. In block 10 *agent1* actually takes the order from the caller, and in block 11 *agent1* is released. In block 12 the total revenue, *x$revenu*, is increased by the invoice total, *p$invtot*, for this call. Block 13 completes the transaction for this caller.

- ***Blocks 26 through 31:*** The simulation will run for one hour of simulated time. At 60 minutes, a stop transaction is created in block 26, it then computes four statistics of interest to management in blocks 27 through 30, and is finally destroyed in block 31, where it causes the termination counter to go from its START value of 1 to 0, shutting down the simulation. Block 27 computes the number of lost calls in the savevalue *x$loscal*, from the total count *n$lost*, of block 25. The number of orders placed, *x$ordpla*, is computed in block 28 by adding the total counts of blocks 13 and 24. The utilization, *x$utlag1*, of *agent1* is calculated in block 29 by dividing the busy time for *agent1* by the simulation clock time at stop. A similar calculation occurs for *agent2* in block 30.

Since the Experiment dialog of WebGPSS requires that the result variable be a savevalue, we are all set to do experiments related to the items of major interest to management: number of orders placed (*x$ordpla*), total revenue from the telephone orders (*x$revenu*), calls lost because all agents are busy (*x$loscal*), and agent utilization (*x$utlag1* and *x$utlag2*). The experiment variable is the number of agents, *x$numagt*. As an example experiment, suppose that we want to determine a 95% confidence interval for the revenue generated when two agents are answering calls and that we plan on running 20 replications. In this case the Experiment dialog would appear as shown in Figure 15.


Experiment

[✓] Experiment on

Variable to change	X$ NUMAGT
Result variable	X$ REVENU
Number of values	1
Lowest value	2.0
Highest value	2.0
Number of run(s)	20

Figure 15: Example Experiment Dialog Box

The experiment output for run 20 would appear as follows in the WebGPSS output, indicating that with 95% probability, the average revenue will lie between $697.21 and $809.79.

```
Result in run 20    870.00
After 20 runs:
Average: 753.50  Standard deviation: 120.27
With 95 percent probability:
Average lies between 697.21 and 809.79
```

This solution has made use of many WebGPSS concepts that the student has learned during the semester, including random discrete functions, parameters, savevalues, standard numerical attributes (e.g., the simulation clock *CL* and *N$*), server and SNA type IF statements, the unconditional GOTO, and use of the *Experiment* dialog box.

3.2 A ProModel Solution

When the students are asked to design a ProModel solution to the Dawn-to-Dusk telephone ordering system problem, they are encouraged to make use of many of the features of ProModel that make it different from WebGPSS, but are at the same time asked to consider similarities and find analogies between the two software packages.

Their ProModel solutions typically center on the built-in animation capabilities of ProModel and include on-screen counters, entity spots, status lights, and dynamic plots, as shown in the screen capture of such a model in Figure 16. Agent1 (i.e., represented by the desk on the left) is currently idle as shown by the blue status light. Agent2 is busy, as shown by the green status light, and is handling a call, represented by the red disk at the desk's entity spot. The on-screen counters indicate that 131 orders have been placed, 41 calls have been lost, and the total revenue is $9370. The dynamic plot shows utilization of each of the agents as a function of simulation clock time, which has elapsed to about 600 minutes when the screen capture was made. Study of this plot leads to an excellent discussion of issues associated with warm-up and steady-state. Students are also asked to indicate modifications that could be made to their WebGPSS solution to produce such a graph. They quickly realize that they could do so easily using the HELP GRAPH block, and that the only difference is that the WebGPSS graph would not be created dynamically.

Students quickly recognize many analogies between WebGPSS and ProModel. ProModel arrivals correspond to the WebGPSS GENERATE block. ProModel EXIT

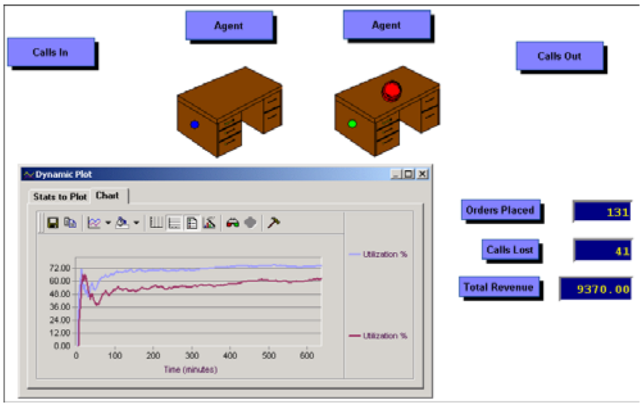


Figure 16: ProModel Screen Capture of the Dawn-to-Dusk Simulation Model

corresponds to the WebGPSS TERMINATE block. ProModel locations correspond in part to WebGPSS facilities, and ProModel multi-capacity locations correspond to WebGPSS storages. ProModel user distributions can be used to create functions similar to WebGPSS random functions. ProModel entities are analogous to WebGPSS transactions. Streams and seeds in ProModel can be used in the same way that the random number streams RN1 through RN8 are used in WebGPSS. ProModel global variables correspond to WebGPSS savevalues. ProModel attributes are analogous to WebGPSS parameters. This list could go on and on.

The major point is that the learning of one of these software packages compliments the learning of the other. I actually teach WebGPSS first and then proceed with ProModel during the last month of the semester. I am quite convinced that the students learn ProModel much more quickly as a result of recognition of analogies between the two. The students are also somewhat surprised that with ProModel, the need for coding has not disappeared. In fact, if anything, coding is a bit more complex because they now must decide where to put the code. Some code may be needed in initialization logic, some in arrival logic, some may be required in operation logic, some in move logic, and some in termination logic.

The ProModel text version of the Dawn-to-Dusk telephone ordering system simulation contains a number of elements that will now be discussed. The time unit is explicitly stated as is the distance unit, required for proper timing of movement of entities in the animation (though our simulation will not make use of movement). The only initialization logic we have is to start the dynamic plot for agent utilizations:

```
Time Units: Minutes
Distance Units: Feet
Initialization Logic:
       DYNPLOT "Agent Utilizations"
```

Global variables are declared for accumulating revenue, orders placed, and lost calls:

```
ID      Type     Initial value
------- -------  -------------
REVENU  Real     0
ORDPLA  Integer  0
LOSCAL  Integer  0
```

User distributions are defined to represent the dice. The code for IATDIE is shown below, and similar user distributions are defined for DURDIE, INDIE1, and INDIE2:

```
ID      Type     Cumulative Percentage Value
------- -------- ---------- ---------- -----
IATDIE  Discrete No         16.66667   1
                            16.66667   2
                            16.66667   3
                            16.66666   4
                            16.66667   5
                            16.66666   6
```

Streams and seeds are declared to be used in the above mentioned user distributions. With the *Reset* field set to *NO*, the stream will continue where it left off for subsequent replications:

```
Stream #     Seed #       Reset
------------ ------------ ------------
1            51           No
2            52           No
3            53           No
4            54           No
```

Entities are defined for use in the simulation. *Call* represents a customer who plans to place an order with Dawn-to-Dusk, *Completed_Call* represents a customer who has successfully placed his/her order. *Lost_Call* represents a customer who has hung up the phone because all agents were busy when the call was placed. Time series is selected if you wish to view this plot in the ProModel output module, though we did not make use of this feature in the Dawn-to-Dusk simulation:

```
Name           Speed(fpm)   Stats
-------------- ------------ -----------
Call           150          Time Series
Completed_Call 150          Time Series
Lost_Call      150          Time Series
```

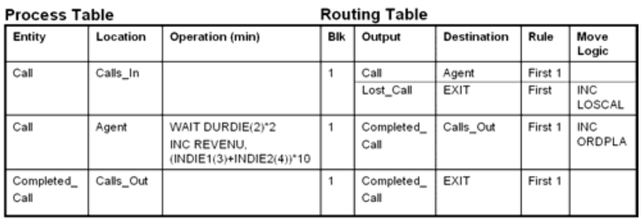
Three major locations were defined including *Agent*, a location with either one or two units (*Agent.1* and *Agent.2*), depending on the number of agents answering calls. *Calls_In* is a location where customer call arrivals occur, and *Calls_Out* is a location where successfully completed calls go after processing by the agent. The rule *Oldest* indicates how the location will select the next incoming entity in the event that several are waiting to enter the location. The rule *First* indicates that the first available unit in a multi-unit station is selected by an incoming entity:

```
Name       Cap  Units  Rules
--------   ---  -----  ---------------
Agent      1      2    Oldest, First
Agent.1    1      1    Oldest,
Agent.2    1      1    Oldest,
Calls_In   1      1    Oldest,
Calls_Out  1      1    Oldest,
```

We have only one type of arrival in our model, and this arrival describes call inter-arrival times. Frequency actually refers to inter-arrival time, not arrivals per unit time. The number in parentheses after *IATDIE* refers to the random number stream to be used by this user-defined distribution:

```
Entity:            Call
Location:          Calls_In
Qty each arrival:  1
Occurrences:       INFINITE
Frequency:         IATDIE(1)
```

The heart of the ProModel solution is contained in the Process and Routing Tables, shown in Figure 17. As an example of how to read these tables, consider the first row immediately following the column headers. It says that a *Call* at the location *Calls_In* will be routed to the location *Agent* and output as a *Call*. However, if all agents are busy, then it will be output as a *Lost_Call* to *EXIT*, incrementing *LOSCAL* by 1 while moving to *EXIT*.



Process Table			Routing Table				
Entity	Location	Operation (min)	Blk	Output	Destination	Rule	Move Logic
Call	Calls_In		1	Call	Agent	First 1	
				Lost_Call	EXIT	First	INC LOSCAL
Call	Agent	WAIT DURDIE(2)*2 INC REVENU, (INDIE1(3)+INDIE2(4))*10	1	Completed_ Call	Calls_Out	First 1	INC ORDPLA
Completed_ Call	Calls_Out		1	Completed_ Call	EXIT	First 1	

Figure 17: ProModel Process and Routing Table for the Dawn-to-Dusk Simulation Model

4 CONCLUDING REMARKS

Having designed the Dawn-to-Dusk telephone ordering system simulation using both WebGPSS and ProModel, the students are finally asked how they could employ changes to make the model more realistic. Suggested changes include:

- The use of exponential call arrivals, rather than call arrivals based upon the discrete uniform distribution provided when modeling a single die.
- The use of the normal distribution to represent call durations.
- The use of a continuous triangular distribution for invoice totals rather than the discrete one obtained when modeling the throwing of a pair of dice.
- Allowing customer callback after a given period of time, for customers who are lost the first time they call because all agents are busy.
- Allowing call queuing.
- Extending the solution to three agents answering calls.

Based upon informal comments and feedback from students having taken this course in computer simulation in business, the alpha/omega approach discussed in this paper has been quite successful. The idea of presenting simulation modeling the first week of class using a dice-based technology gives the students a hands-on approach to simulation that is quite intuitive, yet drives home many simulation concepts at an early point in the course. Returning to the same problem at the end of the course and developing computer-based solutions using the software the students have studied throughout the course, brings the student full-cycle and challenges the student to apply what they have learned.

REFERENCES

Born, R. and I. Ståhl. 2002. *WebGPSS Slide Presentation*. PowerPoint Presentation to accompany Ingolf Ståhl's *Simulation Made Simple with WebGPSS*.

ProModel, 2002. *ProModel Version 5.0 Manufacturing Simulation Software User's Guide*. Oren, Utah: ProModel Corporation.

Ståhl. I. 2002. *Simulation Made Simple with WebGPSS*. Tutorial, Stockholm School of Economics, Stockholm, Sweden.

AUTHOR BIOGRAPHY

RICHARD G. BORN is an Associate Professor of Management Information Systems in the Department of Operations Management and Information Systems in the College of Business at Northern Illinois University. He has taught simulation modeling for the past 12 years to university students at all levels from undergraduate to graduate, including M.S. students in Management Information Systems, M.S. students in Accountancy, and M.B.A. students. His email address is <rborn@niu.edu>.