Proceedings of the 1987 Winter Simulation Conference
A. Thesen, H. Grant, W. David Kelton (eds.)

INTRODUCTION TO SIMULATION

William E. Biles
Industrial Engineering Department
Louisiana State University
Baton Rouge, LA 70803

Simulation is a valuable tool in the study of complex systems. This paper discusses the fundamental principles of simulation. Its intent is to provide sufficient background in simulation that the potential user would be encouraged to follow through with additional study and soon come abreast of the techniques of simulation.

I. INTRODUCTION

Simulation is defined as the development of a mathematical-logical model of a system and the experimental manipulation of the model on a digital computer. Two basic steps in simulation are cited in this definition: (1) model development, and (2) experimentation. Model development involves the construction of a mathematical-logical representation of the system, and the preparation of a computer program that allows the model to mimic the behavior of the system. Once we have a valid model of the system, the second phase of a simulation study takes place - experimentation with the model to determine how the system responds to changes in the levels of the several input variables.

The terms "model" and "system" are also very important in the definition of simulation given above. A system is a collection of items from a circumscribed sector of reality that is the focus of study. A system is a relative thing, and we define the boundaries of the system so as to include those items that are deemed most important to our objectives and to exclude items of lesser importance. For instance, if our focus is the operation of an outpatient clinic in a hospital we will define the boundaries of the system so as to include the physicians, nurses, staff, facilities and services of the outpatient clinic, while excluding all other areas of the system.

A model is the means we choose to capture the important features of the system under study. The model must possess some representation of the entities or objects in the system, and reflect the activities in which these entities engage.

The steps in a simulation study are as follows:

1. Problem formulation - a statement of the problem that is to be solved. This includes a general description of the system to be studied and a preliminary definition of the boundaries of that system.

2. Setting objectives - a delineation of the questions that are to be answered by the simulation study. This step allows further definition of the system and its boundaries.

3. Model building - the process of capturing the essential features of a system in terms of its entities, the attributes or characteristics of each entity, the activities in which these entities engage, and the set of possible states in which the system can be found.

4. Data collection - gathering data and information which will allow the modeler to develop the essential description of each of the system entities, and developing probability distributions for the important system parameters.

5. Coding - the process of translating the system model into a computer program which can be executed on an available processor.

6. Verification - the process of ascertaining that the computer program performs properly.

7. Validation - the process of ascertaining that the model mimics the real system, by comparing the behavior of the model to that of the real system where the system can be observed, and altering the model to improve its ability to represent the real system. The combined steps of verification and validation are crucial to establishing the credibility of the model, so that decisions reached about the system on the basis of the simulation study can be supported with confidence.

8. Experiment design - determining the alternatives that can be evaluated through simulation, choosing the important input variables and their appropriate levels, selecting the length of the simulation run and the number of replications.

9. Production runs and analysis - assessing the effects of the chosen input variables on the selected measures of system performance, and determining whether more runs are needed.

10. Simulation report - documenting the simulation program, reporting the results of the simulation study, and making commendations about the real system on the basis of the simulation study. The implementation of the these recommendations is usually the result of a decision by the appropriate manager in the organization.

II. PRINCIPLES OF DISCRETE-EVENT SIMULATION

The concepts of a system and a model of a system have already been discussed. This section expands on those concepts and establishes a general framework for discrete-event simulation. The major concepts are briefly defined and then illustrated with examples:

System - a collection of entities that interact together over time to accomplish a set of goals or objectives.

Model - a mathematical-logical representation of the system in terms of its entities and their attributes, sets, events, activities and delays.

System State - a collection of variables the values of which define the state of the system at a given point in time.

Entity - an object, item, or component of the system which requires explicit representation in the model.

Attributes - the properties or characteristics of a given entity.

Set - a collection of associated entities.

Event - an instantaneous occurrence in time that alters the state of the system.

Activity - a duration of time in which one or more entities are engaged in the performance of some function that is germaine to the system under study, the length of which is known at the outset.

Delay - a duration of time of unspecified length, the length of which is not known until it ends.

To illustrate these concepts, consider the example of an outpatient clinic in a hospital. The entities in this system include patients, physicians, nurses, and examining rooms. The attributes of a patient include, for example, the nature of the disorder, the time the patient arrives at the clinic, and the type of insurance coverage available. In fact, the patient's entire medical history could form attributes of the patient. The attributes of a physician might include type of specialty, number of patients in-process, and number of nurses assigned. A set might include the patients waiting for service, ordered by severity of disorder and first-come-first-serve within disorder priority. An activity might be typified by the examination of a patient by a physician, or the time required to perform a X-ray procedure. A delay might be the time a patient spends waiting to see a physician. An event might be the arrival of another patient into the clinic, the completion of the examination on a patient by the physician, or the completion of an X-ray by a laboratory technician.

The scheduling of events is accomplished by a next-event approach. Time is advanced from the time of a current event, t , to the time of the next scheduled event, t . The calendar of events consists of a file containing the time and type of each of n events, arranged in chronological order in the memory of the computer. The simulation program must have a mechanism for fetching the next event that is scheduled to occur, automatically advancing simulation time to the scheduled time of occurrence of that event, and transferring control to the appropriate event program. Figure 1 illustrates the time-advance procedure in next-event simulation.

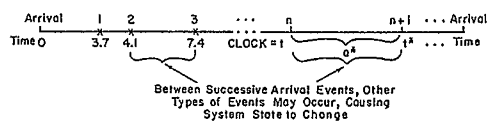

FIGURE 1. TIME-ADVANCE PROCEDURE IN NEXT-EVENT SIMULATION

The method by which events are placed in the event calendar is extremely important. One or more events must be initialized at the outset of the simulation. Two techniques are then applied to generate "future" events in the course of the simulation. One such technique is bootstrapping, which refers to the process by which the occurrence of an event is used to generate the next occurrence of the same type of event. This technique is perhaps best illustrated by the arrival process of patients in the outpatient clinic. The first arrival is initialized to occur just as the clinic opens, or shortly thereafter. When that arrival event occurs, the time of the next arrival is immediately scheduled and placed in the event calendar. Thus, arrivals generate new arrivals as the simulation proceeds. The second method for generating events is the next logical event approach. For example, when a physician has examined a patient he can (a) discharge the patient, (b) schedule one or more laboratory tests, (c) refer the patient to another specialist on the current visit, or (d) schedule another visit. Having done one or more of these, the physician can then take another patient from the queue, which is another example of the bootstrapping approach to event scheduling; that is, service completion events generate subsequent service completion events.

In the event-scheduling approach to simulation, we concentrate on events and their effect on system state. In our outpatient clinic example, system state is reflected by such variables as the number of patients in the system (clinic), the number of busy physicians, the number of busy nurses, the number of busy examining rooms, etc.

For instance, when a patient arrival event occurs, it increases the number of patients in the system by one, possibly increases the number of busy nurses by one, and possibly increases the number of busy physicians by one. The word "possibly" is important, since the newly arrived patient could only activate a nurse or doctor if the nurse or doctor is idle. A "patient discharge" event reduces the number of patients in the system by one and possibly deactivates a nurse or doctor.

Thus, the occurrence of any of the events which make up the model will bring about a resultant change in system state. The model must provide for recording these state changes. This is usually managed by taking one or more types of statistics at the occurrence of each event. For instance, when a patient enters or leaves the system, we would update a time-dependent statistic to reflect the change in the number of patients, whereas we would collect sample statistics on the duration of time the patient spends in the system. We might also update a counter as the patient left an examining room to keep track of the number of times that facility is utilized.

A different outlook on discrete-event simulation is provided by the process-interaction approach. A process is a time-ordered collection of events, activities, and delays which are somehow related to an entity. For example, the sequence of events, activities, and delays encountered by a patient in the outpatient clinic constitute a process. Process-oriented simulations usually have many processes ongoing simultaneously and involve extremely complex interactions among these many processes.

III. DISCRETE-EVENT SIMULATION LANGUAGES

Discrete-event simulation models can be developed using one of three general classes of simulation languages: (1) high-level languages such as FORTRAN, Pascal, Ada, or C; (2) general-purpose simulation languages such as GASP-IV, Simscript, SLAM-II, or SIMAN; or (3) special-purpose simulation languages such as GPSS.

FORTRAN, Pascal, Ada and C are high-level programming languages which were developed for a wide range of computing applications. FORTRAN is the oldest of these languages and has been widely applied as the base language in simulation modeling, due mainly to its being so well known and so widely available on almost any computer of sufficient size and speed to be able to accommodate computer simulation. Almost all simulation models developed directly from FORTRAN have utilized the event-scheduling approach. The recent emergence of smaller microcomputers has led to greater use of Pascal as a base language for simulation, while the growing popularity of the Unix operating system has led to more work with the C language. These languages have not yet been applied in commercially available simulation languages, but soon will be.

GPSS (Schriber, 1974), one of the first process-oriented simulation languages, is a highly structured special-purpose simulation language which was designed for use primarily with queueing systems. The later, general-purpose simulation languages, notably GASP-IV (Pritsker, 1974) and Simscript (Kiviat et al, 1973) were largely event-oriented, but afforded more general constructs for model building. These languages initially found favor among those simulation modelers who had previously relied on FORTRAN as the base language. SLAM-II (Pritsker, 1985) and Simscript II.5 (CACI, Inc., 1976) evolved from GASP-IV and Simscript, respectively, and offer the analyst a choice of either event or process orientations. SIMAN (Pegden, 1986) also provides a choice of orientations, but differs from the other general-purpose simulation languages in that it enables the analyst to develop separate model and experiment frames, thus permitting greater ease in experimenting with the simulation model once it has been developed.

The special-purpose simulation languages were designed to allow easy modeling in a highly specialized area of application. These languages have not found as wide an application as the general-purpose languages because analysts prefer to acquire simulation tools that afford them greater flexibility in a broader range of application environments.

IV. STATISTICAL TOOLS IN SIMULATION

The simulation modeler sees a probabilistic world. The time it takes a machine to fail is a random variable, as is the time it takes a maintenance mechanic to repair it. Simulation modeling requires skill in recognizing the random behavior of the various phenomena that must be incorporated into the model, analyzing data to determine the nature of these random processes, and providing appropriate mechanisms in the model to mimic these random processes. This section discusses the basic concepts in probability and statistics as they relate to discrete-event simulation modeling. Thses basic terms are as follows:

Random variable - a variable X which can assume any of several possible values over a range of such possible values.

Discrete random variable - a random variable X in which the range of possible values is finite or countably infinite. For $x_1, x_2, \ldots,$ $p(x_i) = P(x = x_i)$ $p(x_i) \geq 0$ for all i $\sum_i p(x_i) = 1$

Continuous random variable - a random variable X in which the range of possible values is the set of reals. $-\infty < x < \infty$. If f(x) is the probability density function of x, then $P(a \leq x \leq b) = \int_a^b f(x)dx$ $f(x) \geq 0$ for all x in R_X $\int_{R_X} f(x) = 1$

Cumulative distribution function - denoted F(x), measures the probability that a random variable X has a value less than or equal to the value x, that is

$$F(x) = P(X \leq x)$$

If X is discrete, then

$$F(x) = \sum_{x_i \leq x} p(x_i)$$

If X is continuous, then

$$F(x) = \int_{-\infty}^{x} f(x)\,dx$$

Expectation - the expected value of the random variable X is given by

$$E(X) = \sum_i x_i\, p(x_i) \qquad \text{if X is discrete}$$

and by

$$E(X) = \int_{-\infty}^{\infty} x\, f(x)\,dx \qquad \text{if X is continuous.}$$

The expected value is also called the mean and denoted by μ . It is also the 1st moment of X. We can also define the nth moment of X as

$$E(X^n) = \sum_i x_i^n\, p(x_i) \qquad \text{if X is discrete}$$

and as

$$E(X^n) = \int_{-\infty}^{\infty} x^n f(x)\,dx \qquad \text{if X is continuous.}$$

From these results, we can define the variance of the random variable X as

$$V(X) = E[(X - E(X))^2] = E[(X-\mu)^2]$$

Useful statistical models in discrete-event simulation include queueing systems, inventory systems, and reliability and maintainability systems. Underlying these systems, however, are several very important discrete and continuous probability distributions that we shall examine here.

Discrete Distributions

Discrete random variables are used to describe random phenomena in which only integer values of the random variable X can occur. The probability distributions of four such random variables which are fundamentally important in discrete-event simulation are discussed here.

1. Bernoulli trial and the Bernoulli distribution. Consider a random experiment consisting of n trials in which each trial can produce one of only two outcomes, success and failure. Let $X_j = 1$ if the jth trial results in a success and $X_j = 0$ if the jth trial produces a failure. For example, let the random experiment consist of the inspection of a manufactured assembly, and "success" be defined as finding a defective assembly. Thus, a defective assembly yields the value $X_j = 1$, while an acceptable assembly yields the value $X_j = 0$. (This viewpoint can, of course, be reversed without altering the model.) The n Bernoulli trials are called a Bernuolli process if the successive trials are independent.

The probability distribution for the Bernoulli trial is

$$P_j(x_j) = p(x_j) = \begin{cases} p, & x_j=1,\ j=1, 2,\ldots,n \\ q = 1-p, & x_j=0,\ j=1,2,\ldots,n \\ 0, & \text{otherwise} \end{cases}$$

For one trial, this equation is called the Bernoulli distribution. The mean and variance of the Bernoulli distribution are given by

$$E(X_j) = p$$

$$V(X_j) = p(1-p) = pq$$

2. Binomial distribution. If we define the random variable X as the number of successes in n independent Bernoulli trials, then X has the binomial distribution given by

$$p(x) = \binom{n}{x} p^x q^{n-x} \qquad x=0,1,2,\ldots,n$$

with mean and variance

$$E(X) = np$$
$$V(X) = npq$$

3. Geometric distribution. The geometric distribution applies to the random variable X which is defined as the number of Bernoulli trials until the first success, and is given by

$$p(x) = \begin{cases} q^{x-1} p & x = 1,2,\ldots \\ 0 & \text{otherwise} \end{cases}$$

with mean and variance

$$E(X) = 1/p$$

$$V(X) = q/p^2$$

4. Poisson distribution. The Poisson distribution describes so-called "rate processes". For instance, the rate of occurrence of arrivals of patients at the outpatient clinic in our earlier example might well follow a Poisson distribution. The rate of occurrence of bubbles per square meter of plate glass, of pits per linear meter of extruded copper wire, and of failures of a milling machine might also follow Poisson distributions. The Poisson probability distribution is given by

$$p(x) = \frac{e^{-\alpha}\alpha^x}{x!} \qquad x = 0,1,2,\ldots$$

with mean and variance

$$E(X) = \alpha$$

$$V(X) = \alpha$$

Continuous Distributions.

Continuous random variables are used to describe random phenomena in which the random variable X can take on any value in some real interval. Seven continuous distributions which are fundamentally important in discrete-event simulation are the uniform, exponential, gamma, Erlang, normal, and Weibull distributions. We shall consider a few of these here.

1. Uniform distribution. A random variable X is uniformly distributed in the interval [a,b] if its probability density function, or pdf, is given by

$$f(x) = \begin{cases} \dfrac{1}{b-a} & a < x \leq b \\ 0 & \text{otherwise} \end{cases}$$

The <u>cumulative distribution function</u>, or cdf, is given by

$$F(X) = \begin{cases} 0 & x < a \\ \frac{x-a}{b-a} & a \le x \le b \\ 1 & x > b \end{cases}$$

The pdf and cdf for the uniform distribution are shown in Figure 2.

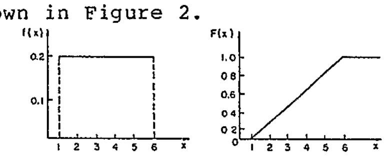

FIGURE 2. UNIFORM pdf AND cdf

The mean and variance of the uniform distribution are given by

$$E(X) = \frac{a+b}{2} \quad \text{and} \quad V(X) = \frac{(b-a)^2}{12}$$

The uniform distribution plays a vital role in discrete-event simulation. <u>Random numbers</u>, uniformly distributed in the interval [0,1], are used to generate the <u>random variates</u> X which give rise to random events. Random number generation is discussed in a later section.

2. <u>Exponential distribution</u>. The random variable X is said to be exponentially distributed with parameter $\lambda > 0$ if its pdf is given by

$$f(x) = \lambda e^{-\lambda x} \qquad x \geq 0$$

The exponential distribution is used to model interarrival times in a random process in which the rate of arrivals is Poisson distributed. In this case, λ is the mean rate of arrivals and $1/\lambda$ is the mean time between arrivals. The mean and variance of the exponential distribution are

$$E(X) = \frac{1}{\lambda} \quad \text{and} \quad V(X) = \frac{1}{\lambda^2}$$

The cumulative distribution function is given by

$$F(X) = 1 - e^{-\lambda x} \qquad x \geq 0$$

Figure 3 shows the pdf and cdf for the exponential distribution.

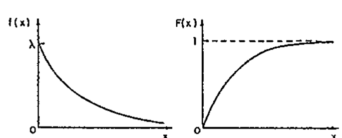

FIGURE 3. EXPONENTIAL pdf AND cdf

3. <u>Normal distribution</u>. A random variable X with mean and variance has a normal distribution if its pdf is given by

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-[\frac{1}{2}(\frac{x-\mu}{\sigma})^2]} \qquad -\infty < x < \infty$$

The normal distribution is illustrated in Figure 4.

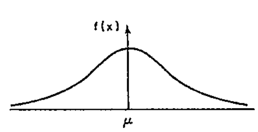

FIGURE 4 NORMAL DISTRIBUTION

V. RANDOM NUMBERS AND RANDOM VARIATES

Random numbers are an essential component of discrete-event simulation models. Most computer languages have function or subroutine that will generate random numbers that can in turn be used to generate random events or other random variables. In this section, we shall examine how to generate (a) random numbers, and (b) random variates from several common probability distributions.

<u>Properties of Random Numbers</u>

A sequence of random numbers R_1, R_2, to be useful in discrete-event simulation: (a) uniformity, and (b) independence. <u>Uniformity</u> requires that each random number R is an independent sample drawn from a continuous uniform distribution in the interval [0,1]. That is, its pdf is given by

$$f(x) = \begin{cases} 1 & 0 \le x \le 1 \\ 0 & \text{otherwise} \end{cases}$$

as shown in Figure 2. The mean and variance of the U[0,1] distribution are 1/2 and 1/12, respectively.

The only way to generate truly random numbers on a digital computer is to store a table of such size to provide all numbers needed during the course of a simulation. This technique would require either very large storage in main memory, or many accesses of off-line memory, neither of which is desirable. The alternative is to use a <u>pseudo-random</u> number generator. Desirable properties of such algorithms are as follows:

1. The routine should be fast.

2. The routine should not require large storage in main memory.

3. The routine should have a sufficiently long <u>cycle</u>, which is the length of the sequence until the series begins to repeat itself in the previous order.

4. The routine should avoid <u>degeneration</u>, which is the condition of continuously repeating the same number.

5. The random numbers should be <u>replicable</u>, given a particular starting seed.

6. The generated random numbers should be uniform and independent.

Techniques for generating random numbers are as follows:

1. <u>Midsquare technique</u>.

This technique starts with an initial number, or <u>seed</u>, consisting of n digits. This number is squared, and the middle n digits taken as the next random number. The random number R is found by simply placing a decimal before the first digit in the n-digit set.

2. <u>Midproduct technique</u>.

Two initial seeds of n digits each are

used to start the sequence. These numbers are multiplied, and the middle n digits taken as the next number in the sequence. This process is repeated as often as needed. This process is illustrated with the following example:

Let $X_0^1 = 7143$ $\quad X_o = 1689$

Then, $U_1 = X_0^1 X_0 = (7143)(1689) = 12{,}064{,}527 \rightarrow X_1 = 0645$ and $R_1 = 0.0645$

$U_2 = X_0 X_1 = (1689)(0645) = 1{,}089{,}405 \rightarrow x_2 = 0894$ and $R_2 = 0.0894$

$U_3 = X_1 X_2 = (0645)(0894) = 576{,}630 \rightarrow X_3 = 5766$ and $R_3 = 0.5766$

3. Constant multiplier technique.

A constant K is used as a multiplier of a seed X . The middle n digits are taken as the next number in the series, and the process is repeated. The example below illustrates this process.

Let $K=4552$ $\quad X_0=6129$

Then $V_1 = KX_0 = (4552)(6129) = 27{,}899{,}208 \rightarrow X_1 = 8992$ and $R_1 = 0.8992$

$V_2 = KX_1 = (4552)(8992) = 40{,}931{,}584 \rightarrow X_2 = 9315$ and $R_2 = 0.9315$

$V_3 = KX_2 = (4552)(9315) = 42{,}401{,}880 \rightarrow X_3 = 4018$ and $R_3 = 0.4018$

4. Additive congruential method.

The method by which values are generated is

$$X_i = (X_{i-1} + X_{i-n}) \bmod m$$

where, by definition, a = b mod m if (a-b) is divisible by m with zero remainder.

5. Linear congruential method.

Using an initial seed X_0, a constant multiplier a, a constant increment c, and a modulus m, random numbers are generated using the following recursive relationship:

$$X = (aX + c) \bmod m \qquad i = 0,1,2,\ldots$$

It is necessary to test random numbers for validity as follows:

1. Frequency test - using either the Kolmogorov-Smirnov or the Chi-Square test, test whether the distribution of the generated set is U[0,1].

2. Runs test - tests runs up or down, or above or below the mean, using a Chi-Square statistic.

3. Autocorrelation test - tests the correlation between numbers in the series.

A gap test and a poker test should also be used. A standard text in statistics should be consulted to see how to apply these tests. Once a reliable random number generator has been ascertained, the analyst can go on to using these numbers to generate random variables in the simulation.

Random Variate Generation

Assuming the availability of a source of random numbers from the U[0,1] distribution, random variates from specified probability distributions can be generated from one of the following three methods: (1) the inverse transform technique, (2) the convolution method, and (3) the acceptance-rejection technique.

1. Inverse-transform technique

A four-step procedure is described below, and illustrated with the exponential distribution:

Step 1. Compute the cdf of the desired random variable X. For the exponential distribution

$$F(x) = 1 - e^{-\lambda x} \qquad x \geq 0$$

Step 2. Set F(X) = R.

$$1 - e^{-\lambda x} = R$$

Step 3. Solve for X in terms of R.

$$1 - e^{-\lambda x} = R$$
$$e^{-\lambda x} = 1 - R$$
$$-\lambda X = \ln(1 - R)$$
$$X = -\ln(1 - R)/\lambda$$

Step 4. Generate random numbers R_1, R_2, R_3, ... and compute the random variates X_1, X_2, X_3, ...

2. Convolution method.

The probability distribution of the sum of two or more independent random variables is called a convolution of the distributions of the original variables. For instance, the Erlang random variable X with parameters (K, Θ) can be shown to be the sum of K independent exponential random variables $X_1, X_2, \ldots, X_K$. An Erlang variate can be generated by summing K exponential variates.

Normal variates are generated by exploiting a result from the central limit theorem which asserts that the sum of n independent and identically distributed random variables $X_1, X_2, \ldots, X_K$, each with mean μ_x and variance σ_x^2, is approximately normally distributed with mean zero and variance one . Applying this result to uniform variates U[0,1], which have mean 0.5 and variance 1/12, it follows that

$$Z = \frac{\sum_{i=1}^{n} R_i - 0.5n}{(n/12)^{\frac{1}{2}}}$$

is approximately normally distributed with mean zero and variance one. One could sum various numbers of U[0,1] variates, but n = 12 is convenient and efficient. To generate a variate $N[\mu_y, \sigma_y^2]$, simply apply the transformation

$$Y = \mu_y + \sigma_y Z$$

3. Acceptance-Rejection Method

In this method, the analyst generates a random number or random variate, subjects this quantity to a test to determine if it meets pre-established criteria, accepts the quantity if it passes the test or rejects the quantity if it fails the test. The procedure continues to generate random numbers or variates and test them until a sufficient number of them pass the test.

Banks and Carson (1984) describe procedures for using the acceptance-rejection method to generate Poisson and gamma random variables. Fishman (1973) and Law and Kelton (1982) give extensive treatments of the subject of random variate generation.

VI. ANALYSIS OF SIMULATION DATA

Competent simulation modeling requires careful analysis of input data as well as output results. Input data form the base on which realistic models are derived. The expression "garbage in - garbage out" should be the motto of the simulation analyst in treating input data.

Input Data Analysis

There are four steps in developing a valid simulation model: (1) data collection; (2) identifying the underlying probability distributions; (3) estimating the parameters of those distributions; and (4) using the estimated parameters and the assumed distribution, test goodness-of-fit. This section discusses porcedures for input data analysis.

1. Data collection. Plan the data collection activity. Be alert to unusual circumstances which might warrant discarding certain data elements. Perform data analysis as the collection activity is underway. Employ scatter diagrams to visually examine the data for correlation among variables. Look for so-called autocorrelation in time-series data.

2. Identifying the distribution. Construct frequency histograms to give an indication of the shape of the probability distribution.

3. Estimating parameters. If the observations in a sample of size n are X_1, X_2 variance from the expressions

$$\bar{X} = 1/n \sum_{i=1}^{n} X_i \quad \text{and}$$

$$S^2 = 1/n-1 \, [\sum_{i=1}^{n} x_i^2 - n\,\bar{x}^2]$$

If the data are grouped into a frequency distribution, it is usually more convenient to compute the sample mean and variance from the relation $\bar{X} = 1/n \sum_{j=1}^{k} f_j x_j$ and $S^2 = 1/n-1 \, [\sum_{j=1}^{k} f_j x_j^2 - n\,\bar{x}^2]$

4. Goodness-of-fit tests. The two most important tests of goodness-of-fit of an assumed distibution to a theoretical distribution are the Kolmogorov-Smirnov test and the Chi-Square test. The K-S test is typically applied to sample with fewer than 50 observations, while the -test is applied to those with more than 50 observations. Banks and Carson (1984) give a detailed treatment of these techniques.

Other techniques which are often useful in analyzing input data are linear and multiple regression, correlation analysis, and time-series analysis. When a thorough analysis of input data has been completed, the analyst is then prepared to construct the simulation model.

VII. VERIFICATION AND VALIDATION

During the process of model-building, the analyst must employ procedures to ascertain the credibility of the simulation model. The first of these steps is called verification. The purpose of model verification is to assure that the conceptual model is accurately reflected in the computer code. The first step in this procedure is to develop a flow chart of the model. This helps the analyst organize the computer code as it is being developed. The unfinished computer code should provide for printing numerous traces and intermediate results so that the analyst can finds flawed results easily should they occur.

Model validation is the process of comparing the behavior of the model to that of the system it is intended to represent. The first goal of the simulation modeler is to construct a model that appears reasonable on its face to prospective model users. He must then examine the structural assumptions and the data assumptions of the model. Structural assumptions are those that relate to how the system operates. Data assumptions are those related to the form of distributions, parameter estimates, goodness-of-fit tests, etc. We have already seen how to perform this analysis. The ultimate test of the model is its ability to predict future behavior of a real system, including perhaps under conditions which may be only proposed. To accomplish this, it is necessary to compare the behavior of the model to that of the system under conditions at which the system can be observed. An important means of model validation is to compare the performance of the model to that of selected, applicable analytic models. Although the assumptions of the analytic model may have rendered its use inappropriate for the system in question, by ascertaining that the simulation model gives comparable results when run under the same assumptions as the analytic model is valuable insight into the behavior of the simulation model.

Output Analysis

Once a valid simulation model is available, the next step in the simulation study is to use the model to predict the performance of the system. If the performance is measured by a parameter Θ, the result of a simulation experiment will be an estimate Θ of Θ. The accuracy of Θ can be measured by its variance. These estimates are called point estimates, and are computed from the time series of n observations $\{Y_1, Y_2, \ldots, Y_n\}$ by the relation

$$\Theta_r = \sum_{i=1}^{n_r} Yri/n_r \qquad r=1,\ldots,R$$

A second type of estimate produced by a simulation run is the interval estimate. For instance, the $100(1-\alpha)\%$ confidence interval for the parameter Θ is given by

$$\Theta - t_{\alpha/2,f}\,\sigma(\Theta) \le \Theta \le \Theta + t_{\alpha/2,f}\,\sigma(\Theta)$$

where $t_{\alpha/2,f}$ is student's t-statistic at $\alpha/2$ and f is the number of degrees of freedom, which is a function of the estimator for $\sigma(\Theta)$ If $\{Y_1, Y_2, \ldots, Y_n\}$ are statistically independent observations, then the sample variance is computed from

$$S^2 = 1/n-1 \sum_{i=1}^{n} (Y_i - \Theta)^2$$

If they are not independent, other means must be employed. An important part of the output-analysis phase of a simulation study is the design of simulation experiments. This involves identifying a set of dependent or response variables Y_j, $j = 1,\ldots,m$ which are assumed to be functions of the set of controllable or independent variables X_i, $i = 1,\ldots,n$. Some of the techniques used for this purpose include (1) designed experiments, (2) regression analysis, and (3) statistical tests of hypothesis. Kleijnen (1975) has presented a thorough treatment of the statistical techniques used in simulation output analysis.

VIII. SUMMARY AND CONCLUSIONS

This paper has reviewed the concepts, principles and techniques of discrete-event simulation. This discussion should be viewed as a guide to the approaches one should follow in conducting a simulation study, but should not be construed as an exhaustive treatment of the subject. The following references include several that enable the simulationist to select a modeling language, several that are useful in input and output analysis, and several that are important in overall simulation philosophy. Anyone engaged in serious simulation modeling efforts should have many of these references at hand.

Finally, it should be stressed once again that computer simulation cannot replace sound analytic modeling in those instances where analytic models are applicable. In more complex real-world situations, however, computer simulation is an indispensible tool for successful analysis.

REFERENCES

Banks, J. and Carson, J. S., (1984) Discrete-Event System Simulation, Prentice-Hall, Inc., Englewood Cliffs, NJ

Biles, W. E. and Bathina, V. R., (1986) "PASAMS: Pascal Simulation and Analysis of Manufacturing Systems," Material Flow, Vol. 3, pp 99-112

CACI, Inc., (1976) SIMSCRIPT II.5 Reference Handbook, Los Angeles, CA

Conway, R., Maxwell, W. L. and Worons, S. L. (1986), User's Guide to XCELL Factory Modeling System, The Scientific Press, Palo Alto, CA

Fishman, G. S., (1973) Concepts and Methods in Discrete-Event Digital Simulation, John Wiley, New York

Hadley, G., and Whitin, T. M. (1963) Analysis of Inventory Systems, Prentice-Hall, Englewood Cliffs, NJ

Kiviat, P. J., Villanueva, R. and Markowitz, H. M., (1973) SIMSCRIPT 11.5 Programming Language, CACI, Los Angeles, CA

Kleijnen, J. P. C., (1975) Statistical Techniques in Simulation, Parts I and II, Dekker, New York

Law, A. M., and Kelton, W. D., (1982) Simulation Modeling and Analysis, McGraw-Hill, New York

Phillips, D. H., (1980) "GEMS: Generalized Manufacturing Simulator," Industrial Engineering Department, Texas A&M University, College Station, TX

Pegden, CC., (1986) Introduction to SIMAN, Systems Modeling Corporation, State College, PA

Pritsker, A.A.B., (1974) The GASP-IV Simulation Language, John Wiley, New York

Pritsker, A. A. B., (1985) Introduction to Simulation and SLAM-II, John Wiley and Sons, New York

Schriber, J. J., (1974) Simulation Using GPSS, John Wiley, New York

WILLIAM E. BILES

Dr. Biles is currently Professor and Chairman of Industrial Engineering at Louisiana State University. He received a BS degree in Chemical Engineering in 1960 from Auburn University, an MSE in Industrial Engineering in 1969 from the University of Alabama in Huntsville, and a PhD in Industrial Engineering and Operations Research in 1971 from Virginia Polytechnic Institute and State University. His primary research interests are in distributed simulation, optimization in simulation, and the development of efficient simulation languages for manufacturing systems. Dr. Biles is Department Editor for Simulation of Transactions of IIE.

William E. Biles
Louisiana State University
Industrial Engineering Department
3128 CEBA
Baton Rouge, LA 70803
(504)388-5339