MEMORANDUM
RM-5378-PR
AUGUST 1967

DIGITAL COMPUTER SIMULATION: MODELING CONCEPTS

P. J. Kiviat

This research is supported by the United States Air Force under Project RAND—Contract No. F44620-67-C-0045—monitored by the Directorate of Operational Requirements and Development Plans, Deputy Chief of Staff, Research and Development, Hq USAF. Views or conclusions contained in this study should not be interpreted as representing the official opinion or policy of the United States Air Force.

The RAND Corporation
1700 MAIN ST. • SANTA MONICA • CALIFORNIA • 90406

Published by The RAND Corporation

PREFACE

This RAND Memorandum is one in a continuing series on the techniques of digital computer simulation. Each Memorandum covers a selected topic or subject area in considerable detail. This study discusses basic concepts. It provides a rationale for simulation, discusses the design and construction of simulation models, and relates simulation as a technique to current problems in simulation technology.

The Memoranda are being written so that they build upon one another and provide an integrated coverage of all aspects of simulation. They should be of particular interest to personnel of the AFLC Logistics Simulation Center, Wright-Patterson Air Force Base, and to Air Force systems designers and analysts. Persons concerned with computer applications and computer programming in general should also find the series useful.

Computer simulation techniques have had a brief but impressive history to which The RAND Corporation has contributed, starting with one of the earliest uses of simulation for the analysis of large-scale logistics systems [1]. RAND has also published Memoranda on simulation programming languages [2], the analysis of simulation-generated output data [3, 4, 5], statistical aspects of modeling [6, 7], and models of maintenance and logistics systems [8-12].

Simulation is now recognized as a standard systems analysis tool. It has been, and is being, used in such diverse areas as weapon system planning, hospital design, job shop manufacturing, and election forecasting.

SUMMARY

This Memorandum is one in a series of methodological studies dealing with computer simulation techniques. While this study does not lend itself to summary, the reader should find it useful to know that it discusses the modeling process -- the steps taken in analyzing a system and designing a computer program that allows a system's operations to be reproduced and studied. Subsequent Memoranda treat other aspects of simulation methodology: computer programming languages, model verification and validation, and experimental techniques. Taken in concert, the Memoranda should provide material suitable for a graduate-level text on simulation. Taken separately, they provide useful information for system designers, applications engineers, and computer programmers.

CONTENTS

I. INTRODUCTION

PROBLEMS IN PERSPECTIVE

The world's problems are usually solved, even if the "solution" is to ignore a problem and hope it will go away. More often, people search for new or better ways of doing things. When circumstances permit, problems are solved optimally; but limitations of time, talent, knowledge, and resources ordinarily dictate that the solutions found are not "best," but merely "better" than an existing state of affairs. This is regrettable, but not entirely so -- it is always better to make some progress than no progress at all, and we need reasonable working solutions as well as optimal ones.

Workable solutions are obtained in many ways: by the use of approximations, by the use of rules of thumb, by the evocation of restrictive assumptions, and by guesswork and crystal ball gazing. Some of these "satisficing"* techniques are more scientific than others, some are more comforting to a manager than to a scientist, some have a chance of being applied intelligently, some have no hope at all. Much has been written on the art of executive decisionmaking and the pros and cons of various decision procedures. We shall not comment upon such issues here; we mention them to set the scene for our topic: the use of computer simulation techniques for problem solving.**

Two reasons that problems cannot be solved optimally are lack of time and talent. We shall gloss over these factors in our discussion because they are relative; when time and talent are in short supply for one person, they may not be for another. We shall consider limitations such as knowledge and resources, which are absolute factors that preclude anyone from obtaining optimal solutions to particular problems at given points in time.

---

*A word coined by H. A. Simon to denote a solution procedure that strives for an acceptable, satisfactory solution rather than an optimal or maximizing one.

**Throughout this Memorandum, the term problem-solving is used in its broadest sense. The reader is free to attach almost any meaning he wishes to it.

Consider first the problem of resources. Many problems can be solved experimentally by manipulating the system in which they are embedded. A common example of such a procedure is an automobile test track on which automobile manufacturers test safety devices and comfort equipment while designing and evaluating the components they plan to install on next year's cars. Unfortunately, many problems cannot be handled this way. Facilities and equipment for experimentation may not exist. Or, while facilities exist, they may not be used because of cost or prohibitions on interference with ongoing system operations. A materials supply system for a continuously operating process typifies a system that must be studied while operating to determine the effect of new supply rates, but cannot chance the possibility of being interrupted. A new auto freeway must be justified on its capacity to relieve traffic congestion and improve transportation but cannot be built to determine whether or not it should be built.

When a system cannot be studied directly (the necessary resources are not available), it can be studied with a <u>model</u>. A model, which we can loosely define as a representation of a system, can take many forms: it can be iconic like a map or a scale model car; it can be symbolic like a set of equations; or it can be an analogy like an electric circuit that behaves like a water storage basin or a flowchart that describes how a system works. Different models exist for different purposes. Iconic models make good visual aids but are usually unsuited for predicting or explaining the behavior of systems; symbolic models are good for prediction and explanation but offer little as visual aids.

We normally study a system so that we can predict or explain its behavior. We want to know how fast a new plane will fly, how much congestion a new railroad line will relieve, how a new ordering policy will affect customer service, if and why a proposed inventory review and resupply system will give better cost performance per customer served than an existing system. These questions, and others like them, cannot be answered by iconic models; they can be answered by models composed of mechanisms that are able to reproduce relevant aspects of system performance and behavior [23].

Let us now consider knowledge, which becomes a limiting factor during the construction of a model. It is the crucial factor when one attempts to formulate and solve a complex problem in analytical terms, for despite the sophistication of today's mathematics there are many complex problems that cannot be stated so that they can be solved analytically. This is a comment both on the state of mathematics, and on the complexity of the problems we want to solve and the irregularities with which we compound them that frustrate our mathematical ability. Were mathematics to advance instantaneously to a point where it could solve all of today's problems it would not be long before we would create new problems beyond its scope.

SOLUTIONS AND SIMULATION

A problem-solver resorts to numerical procedures when he has insufficient knowledge to solve a problem analytically. Such procedures provide useable solutions by replacing complex problems with simpler ones that can be solved, and that approximate solutions to the original problems. A good example of such a procedure is the numerical integration of a mathematical function for which an integration formula does not exist. One method of performing numerical integration is to fit a series of rectangles of small width into the area beneath a function; the total area of these rectangles, which can be calculated, approximates the area beneath the function. A result is obtained by replacing an analytical solution with a numerical one.

Analytical procedures are usually preferred over numerical ones, as they are more accurate and less costly to compute. When they are not available, numerical procedures are used. The development of a satisfactory numerical procedure is no small task in itself as there often are a number of similar procedures a problem-solver can choose from, each having a particular mix of qualities, such as degree of error, computational speed, skill needed to apply the procedure, and so forth. The use of numerical techniques places an added burden on a person with a problem. He must not only think about his problem, but about procedures for estimating its solution (building a model) and the errors inherent in the estimates (using the model).

Simulation is a term commonly applied to the use of models to study systems. Since we are interested here in models for explanation and prediction, we rule out iconic models and similar descriptive representations. We consider a narrower definition of simulation -- the use of numerical models for the study of systems. These models can be analog or symbolic. An example of an analog simulation model is a flowchart representing the checkout process at a supermarket. The overall system logic describes the checkout process by logical relationships and flow paths. Particular parts of the model can be symbolic if they contain mathematical or statistical procedures for determining such things as the number of packages customers purchase and the amount of time each customer spends in line.

While some writers would define studies of systems that are completely described by solvable mathematical equations as simulation studies, this is not what we are describing. Such definitions are normally found in the natural sciences and in studies of pure engineering systems. To us, simulation is the use of a numerical model to study the behavior of a system as it operates over time.* In particular, we are interested in models that are implemented on digital computers -- models that operate by advancing a system through time in discrete steps rather than continuously, as is done on analog computers. We do not discuss the concepts and techniques employed in simulating systems whose state changes continuously over time. Such systems have traditionally been studied with analog computers, although digital computers are being employed today. The technical journal Simulation is the best source of information on continuous-time simulation.

Since simulation is an experimental, numerical technique, it is usually more expensive to use than analytic solutions. It is normally considered a technique of last resort, employed only if a problem cannot be solved another way. Yet, despite this, it is also widely used, for

---

*Note the use of the word "study" in the above definition. When a simulation model is given a set of input parameters and an initial system state, it is "run" to deduce subsequent system states and estimate measures of system performance. Different parameter settings produce different system responses. These responses are studied to determine the set of parameter values that in some sense optimizes system performance. A simulation model is used in an experimental manner; it does not find or seek optimal system parameter settings by itself.

there are many problems that cannot be solved analytically. Indeed, most of today's complex engineering and management studies include simulation experiments. Let us see why this is so.

SIMULATION AND SYSTEMS ANALYSIS

Thus far we have discussed the simulation of systems without describing what a system is. We can at this point provide some intuitive feel for the kinds of systems that simulation is used to investigate while deferring a more complete and detailed description of a system until Sec. III.

Two objects operate as a system when they are integrated so that the performance of one affects that of the other. A study of one cannot be made in isolation from the other without losing effects caused by their interaction. Questions cannot be asked about the performance of an integrated system by studying its components separately; they must be studied together. When (subsystem) interactions do not exist, an environment can be considered as containing several separate and distinct systems, which can be studied independently.

More and more frequently, studies are made of complex systems composed of large numbers of objects, each interacting with the other according to (complicated) performance rules. This is due to an increasing recognition that systems must be treated as a whole and not as sums of their component parts. As our technical knowledge has increased, so has our awareness that we can unwittingly suboptimize* if we are not aware of total system interactions and effects. The modern view is to define a system in terms of its inputs, its outputs and the processes required to transform one into the other. As the awareness of problem environments has broadened, so has the scope of the systems people study. The concomitant effect has been the removal of problems from the reach of analytic study.

The tendency has been to turn to simulation as a tool for studying complex systems. In fact, it is common to find the terms system simulation and simulation used interchangeably.

---

*A term used to describe a process of local optimization where subsystems are analyzed and designed separately, often to the detriment of the total system.

When problems are viewed in a total systems perspective, it becomes clear that properties of a total system are more important than those of subsystems operating within it. An example of two subsystems operating together interactively is a supply system and a maintenance system at an Air Force base. Neither can exist without the other, for one supplies what the other requests and vice versa. If cost is the measure being used, total system cost must be the criterion, not individual maintenance and supply system costs. Reducing stock levels can degrade maintenance performance; altered maintenance policies can affect demands for stock replenishment. The cost-effectiveness of the two systems operating together must be the design criterion, since each subsystem exists for the benefit of the whole, and not for itself.

Systems do not have to be large or complex for simulation to be useful. There are other reasons for using simulation that are independent of a system's size or complexity. Two such reasons are requirements for experimental control and the presence of statistical variation.

Simulation is often preferred over real world experiments when there are difficulties in controlling parameter changes in different geographical locations or in keeping ambient environmental conditions constant throughout a test. When a carefully controlled environment is required, as might be the case during the evaluation of a materiel supply system serving many Air Force bases, simulation can provide more control than world-wide test.

In other instances, analytical solutions exist for classes of problems under restricted conditions, such as constant service facility operating times; but when other conditions exist, e.g., when quantities behave according to statistical distributions that have unpleasant analytical properties, simulation may have to be used. Certain classes of <u>structurally</u> simple analytical models, such as those for single-channel waiting lines, must often be simulated rather than solved even though they fit a system under study because their <u>statistical</u> properties do not admit analytical solutions.

SIMULATION AS A TOOL FOR SYSTEMS ANALYSIS

There are two ways an analyst can look at measures of system performance; he can look at measures of average behavior or at measures of dynamic response. The sophisticated analyst looks at both.

Measures of average performance (means, standard deviations, histograms) are the traditional outputs of systems studies. Typical performance measures that are used in analyses of industrial and military systems are average lengths of waiting lines, average durations of idle and busy periods for machines and machine operators, and average system throughputs. These measures allow systems to be compared statically; system A is usually preferred to system B if the values of its average performance measures are "better." The assessment of "better" is made by analyzing the measures, taking into account their variation over time and other statistical properties.

Often people are concerned with other than static comparisons, they not only want to know what <u>level</u> a certain measure achieves but also how it achieves it. They are concerned with <u>system dynamics</u>, the way a system responds to different shocks and disturbances. Typical dynamic performance measures are the sample correlogram and sample spectrum. These measures portray the time-dependent behavior of systems; they allow one to discriminate between systems that have identical average performance but different behavioral characteristics. Using them, one is able, for example, to select a system that responds fastest to peak load conditions from two systems that, on the average, perform the same.

Simulation is one of the few tools available for estimating system dynamics. Most analytical techniques are only able to determine static measures, dynamic performance measures being derivable analytically for only a few, extremely simple systems.* Simulation has emerged as a natural tool for the dynamic study of large, complex systems.

---

*This situation is not nearly as pronounced in studies of continuously changing systems where systems of partial differential equations can be solved to obtain dynamic response functions.

Many factors create a climate for simulation studies. Some that we have mentioned are an increased interest in total systems analyses, requirements for dynamic as well as static performance measures, and the presence of fewer possibilities for real-life experimentation with today's complex systems.

PLAN OF THE COMPUTER SIMULATION SERIES

Assuming that a researcher wants to do a simulation study, there is a great deal of technical material he must master before he can do a study well. Unlike many other techniques (linear programming and queueing theory, for example, which require extensive formal education), simulation programs can be written with little training, and used to produce realistic output. The trouble with casual approaches to simulation lies in their proclivity to generate answers that seem correct but actually are not, their excessive consumption of programming and computer time, and their tendency to lead people down the garden path until their moment of truth arrives. One of the greatest difficulties with simulation is the ease with which programs can be constructed that only appear to reproduce the behavior of an object system. It is one thing for a model to resemble a system, another to act like it.

This series of Memoranda presents the technical material needed to conduct efficient simulation studies. This Memorandum thus far has provided a rationale for simulation, explaining why it is needed and how it can be used to advantage. Along the way it has defined many basic terms. The remainder of the Memorandum describes the modeling process, going in some detail into the components of models and the procedures involved in their construction.

Other Memoranda in the series study simulation programming languages, input data analysis and the generation of random variables, verification and validation of models, the statistical analysis of data generated by simulation models, and the experimental use of simulation models.

II. SIMULATION

Loosely speaking, simulation is the manipulation of a model of a system in such a way that "properties" of the system can be studied. The manipulation may be by hand, by a computer (digital or analog) or by combinations of people and computers working together. While there are distinctive problems associated with different modes of manipulation, they are in a great sense operational problems, and not basic to the theory or substance of simulation itself. Topics germane to a fundamental study of simulation are those that deal with and define the key words in the definition above: model, system, properties and manipulation. This section deals with these topics.

We simulate systems because we want to understand how they work, determine the factors that influence their behavior, and observe how they react to environmental changes. We are generally interested in system response; we are interested in using simulation models to give us information that will enable us to predict or control the future of that segment of reality that a simulation model represents.

Simulation is done for a variety of reasons, some practical and some theoretical. This section describes what we do when we "simulate," pointing out in some detail those areas of critical importance to simulation and thereby illustrating its limitation and advantages. This will enable practitioners to use simulation to its greatest advantage and enable simulation theorists and researchers to place their efforts, be they broad or narrow, in a general perspective of simulation methodology.

THEORIES*

Scientists do research by developing theories and using them to test hypotheses about, gain insights into, and predict the future of the worlds these theories represent. A theory is a structured body

---

*The organization of this section was greatly influenced by Part III, "The Organization" of [13].

of knowledge about some phenomenon that allows us to make meaningful explanatory or predictive statements about it. Some theories can be proven mathematically, others require empirical validation through observation, collection and analysis of data. Until a theory has been proven, it is called a hypothesis, indicating its conjectural status. Much has been written about man's inability to validate theories in both the physical and social sciences; it is often simple to show that relationships exist but difficult to demonstrate causality. To a great extent the validation of hypotheses depends upon the measurement process. Many theories have been usefully used for many years only to be disproved later when more refined measuring instruments have shown that they are only approximations. In realistic terms, these "disproven" theories cease to be useful only when they can no longer be applied to practical situations.

Theories are generally composed of implicit and explicit components. A "theorymaker" states most of his assumptions and premises explicitly; he leaves some things unstated either because they are derivative, i.e., stem from explicit assumptions, or are factors that are so much a part of his environment that he takes them for granted. Each user of a theory must be aware of these limiting factors, of the degree with which his reality agrees with the assumptions underlying a theory and of the approximations (if any) that he is accepting by using it.

MODELS

The words theory and model are often used interchangeably. While this does not often lead to misunderstanding or difficulty, there is an important difference between the two that is crucial to an understanding of simulation. A model is formalized theory, a stylistic interpretation of a body of propositions that a theory represents. Just as there can be many theories of the working of a particular system, there can be many models that formalize a theory.

This is well illustrated in simulation by the numerous simulation programming languages that are available today [14]. Generally, simulation analysts and programmers develop an approach to a simulation study (a theory of system operation) and then code a model of the system in a particular simulation language. Each language particularizes the general approach to the problem in a unique model structure. Each language brings a different formal mechanism to bear and provides a slightly different interpretation of the theory.

ASSUMPTIONS

Just as theories rest on assumptions, so do models. The assumptions of theories are often quite general and accepted without dispute, e.g., man is a rational decisionmaker. The assumptions of models are sharper, more exposed and subject to more detailed discussion. This is not to say that assumptions are more or less important when used in a theory or a model; it is to say that the controversy that arises over assumptions formalized in models is a function of their necessary narrowness and application in particular, problem-oriented contexts.

As assumptions bound a theory and make it possible, they structure a model and make it viable. The choice of assumptions depends on many things: the nature of a model, the environment in which it exists, and the use to which it will be put--to name but a few. The more highly structured a model, the more numerous its assumptions. The narrower a model becomes, the more use it must make of assumptions to limit the world in which it is embedded and to create a suitable climate for precision. But as a model becomes more structured and more refined, as it increases its number of assumptions (both explicit and implicit), questions of value as well as technical fact enter. Value-judgment assumptions affect both the answers that a model can supply, and interpretations that can be made of them.

It is most important that assumptions for simulation models be clearly stated and not hidden in the complex morass of technique known as computer programs. For it is the assumptions, much more than the technical apparatus, that determine the purpose of a model, and the credibility that can be ascribed to its predictions.

SCOPE OF A MODEL

Simple theories (models) are broad in scope, as they contain few assumptions. Broad deductions can be derived from them. Complex theories and highly structured models, on the other hand, are narrow in scope. They are narrow because many questions that might be asked of them are inherent in their structure, i.e., have been assumed away. The more complex and structured a model becomes, the less able it is to answer new and unexpected questions. Highly structured (empirical) models are useful for answering carefully phrased, narrow questions; models with low empirical content are able to respond to broader classes of questions, but with less certainty. Scope must ultimately be determined by the purpose of a model. There can be no abstract determination of a "correct model," there can only be a determination that a model is broad enough to answer questions asked of it. Thus, as a theory is devised to explain something, a model is constructed to answer certain classes of questions. The often heard statement "Build a simulation model of system X for me," is as meaningless as the statement, "John, I need a theory of the atmosphere." Without a statement of purpose, there can be no theory, no model.

THE VALUE OF DETAIL

Since a model's value lies in the way it can be used, detail is necessary only to the extent that it contributes to the precision of model predictions or estimates without limiting the variety of questions that can be asked. A general, aggregative model may have greater value than a detailed and highly parameterized one if the model is intended to explore possibilities and not determine system operating characteristics. Value must be placed on the utility of answers and not on inherent model characteristics.

The subject of the proper congruence between model and object system is one of long standing. Must a model bear a one-to-one resemblance to its object system to be "useful." or need it only act like it? And if it is just supposed to "act like it," exactly what does this mean? Once again, the answer to this question lies in the purpose of a model. If the model is to be used as an analog, e.g., as a small-scale operating system, then it must work much like the real system; if it is to be used as a predictive device, then it need only produce the correct outputs when it is given input parameters, without regard for the mechanism used in transforming the input values to the output results. Most often, simulation models are used both as explanatory (descriptive) and predictive devices. Given a set of input conditions, a model is used to predict results and describe the way the results are determined, i.e., the researcher is concerned with both system response and system dynamics. Since people usually enter into explanatory models without knowing exactly what it is they are trying to explain, the pressure is to make everything as detailed as possible. As a general principle, this is incorrect. A model should only be as detailed as is necessary to answer the questions at hand; it should be so designed, however, so it can be expanded to include more detail without inordinate cost in those model areas which have a high probability of becoming subsequent subjects of concern.

One more topic must be covered in connection with model detail--uncertainty. A decisionmaker in search of a fact uses estimates of the fact; he is concerned with the confidence he can place in any estimate he is given. Since it costs money to ferret out facts, and generally costs more as confidence requirements are increased, the decisionmaker asks, "How much is it worth for me to know, with a certain degree of confidence, that fact X is true (or false)?" The degree of confidence a decisionmaker requires is of course beyond the scope of this Memorandum. The problems of achieving stated degrees of confidence are not. A simulation model designer must consider, from the points of view of structure and data, the statistical aspects of the system he is modeling.

Structural uncertainties arise from our imprecise knowledge of how systems function, from our (occasional) inability to separate real and apparent causes, and from the common error of confusing correlation and causation. Any model that is predicated on a certain structure that has been arrived at through observation rather than theory runs the risk of making predictions that must be qualified by a statement, "If the system really acts this way, then...." A confidence statement about structure might be: "We are 95 percent certain that components fail in this way; if they do, then we can support our level of operations with six aircraft." Accepting an uncertain structure is equivalent to making an assumption, and, as with all assumptions, limits a model's scope and utility.

Uncertainties in simulation results caused by data elements are due to difficulties in parameter estimation, in the selection of correct statistical sampling distributions and in generating truly "random" sample data. Two types of uncertainty exist: uncertainty caused by input data errors (estimation) and uncertainty due to the statistical nature of simulation models (randomness). The second type shows itself in problems associated with simulation model run length and output data analysis.

Uncertainty and detail are therefore intimately related. We may reduce the uncertainty in a model by improving its structure, i.e., by perfecting the mechanisms that comprise it, but by doing so we may increase uncertainty in the other sense, i.e., by replacing structural uncertainty with statistical uncertainty. Uncertainty can almost always be reduced; the interesting question is how does one arrive at a model that has maximum utility and minimum uncertainty at lowest cost? While this Memorandum cannot answer this question, it hopefully points out important concepts and factors that enable a simulation user to move in this direction.

SUMMARY

Before a simulation model is designed, two important questions must be asked and answered: (1) what use will be made of the model (what questions will be asked); and (2) what are the requirements of accuracy and precision? Answers to these questions determine the structure of a model, as they demand that certain assumptions be made, that certain boundaries be imposed and respected, that certain types of questions can and cannot be asked, that certain territories cannot be explored, and that certain realities cannot be predicted.

To be sure, it is almost always possible to create an extremely general model, so highly parameterized and loosely structured that almost anything can be asked of it. To do so places primary emphasis on flexibility and secondary emphasis on efficiency; it implicitly states that the overriding cost is reprogramming, e.g., adapting an existing model to new conditions, and that costs such as initial programming effort and time, computer execution time, and data analysis and preparation time are secondary. There are cases where such an approach is justified, and naturally there are cases where it is not.

III. MODELING CONCEPTS

STATE DESCRIPTION OF A SYSTEM

Simulation models are constructed for the analysis of systems, which we broadly define as bounded sectors of reality. One of the first assumptions made in a simulation study is the boundary of the world to be included in a model. The fact that models are selective requires that system boundaries be defined and assumptions be made concerning the way the enclosed system interacts with the world that lies outside its boundaries.

Once a system has been defined, the purpose of the simulation study determines what a model of the system will look like, for there can be many models of the same system. Models differ from one another because (1) the theories that they formalize are different and (2) because they employ different technical mechanisms. It is the first of these causes that we are concerned with here.

An example of a system that can have many models, i.e., theories about how it operates, is an Air Force base. For any particular base or for a generalized base structure there can be maintenance models, personnel models, operations models, and so forth. Each model is different, viewing the total system from a different point of view, and yet including the same system elements. This is done by means of varying assumptions about how subsystems interact and by varying degrees of detail in specifying system structure.

Systems are distinguished from one another by their static and dynamic structures. The entities (objects such as people and machines) that make up a system, along with their associated attributes (characteristics such as age, weight and mental status) and membership relationships (connections between entities, such as being a member of a family, the Air Force, or the Masons) define its static structure. The activities in which these entities engage specify its dynamic structure. There can be systems with identical static structures and different dynamic structures and vice versa. A model's ultimate use determines its structure. A model used for comparative statics may have no dynamic structure; it may merely project changes in system static structure on the basis of statistical or logical relationships. A model used to analyze dynamic behavior leans heavily on a structure that explains how a system moves ahead in time from static state to static state.

A system is said to be in a certain state when its entities have properties unique to the state. These properties are such things as numerical attributes of temperature, value and color, and logical relationships of membership in groups or sets. Depending on the view taken toward activities, whether they interact at discrete points or over periods of time, there can be attributes associated with dynamic as well as static system constructs. That is, an activity can be fully or partially completed, be in progress or terminated, be waiting for another activity to occur or be interrupting another activity, etc. Viewed in a very general sense, there is no conflict in the description of state as a static or dynamic phenomenon. For at all times, whether between state changes as described in a static sense, or at system activity intersections, the concept of a system state is completely defined.

Viewed at a point in time, a system model is always in one of a (perhaps enormous) number of states. Viewed over a period of time, a system model passes through a succession of states as its entities undergo system activities, change their attribute values and relationships, and become eligible for subsequent activities and status positions.

Simulation is the manipulation of a model to reproduce the operations of a system as it moves through time. As such, a simulation analyst is concerned with techniques that move a system from state to state, and with techniques that draw inferences from these movements. As we have previously pointed out, some simulations concern only static aspects of system change; they concern the final values of entity and system attributes and not how these values are obtained. Other studies concern system dynamics; they are interested in the ways in which systems arrive at different states. These different types of simulation studies require different model structures. Models that concentrate on static aspects of systems tend to be less detailed than those that concentrate on dynamics; they tend to be more statistical and less mechanistic. The degree to which a model is able to serve both purposes efficiently is a function of the assumptions made about its internal operations and the techniques used in its implementation.

SIMULATION MODEL STRUCTURE

A simulation model can be viewed as a system state generator. Given an initial system state, it moves a system to new states using information contained in the system, extracted from previous state changes, and communicated from outside the system boundaries. A simulation model is portrayed structurally in Fig. 1.

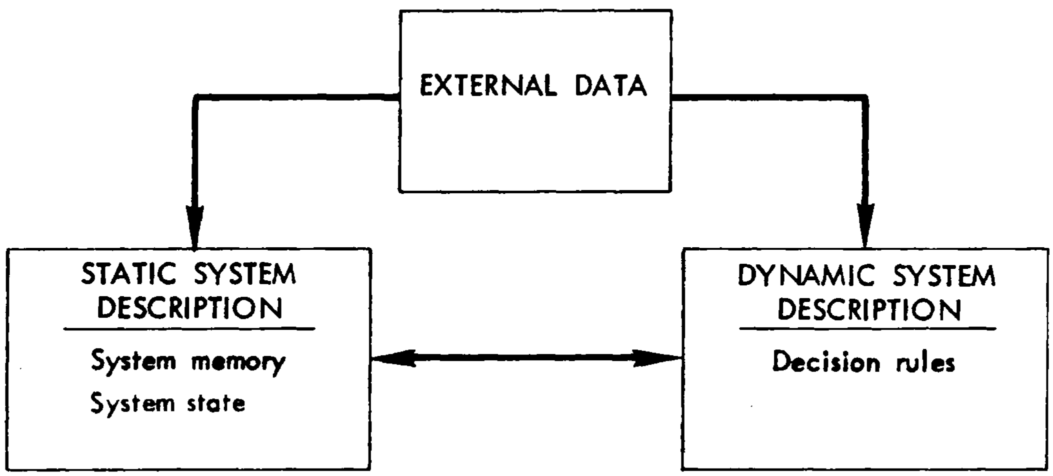

Fig. 1 -- Structure of a simulation model

The static and dynamic system descriptions define a model's current state. The system decision rules use the state data to determine new system states. In so doing, they may use data external to the system and information extracted from previous state changes. Models that make use of such "remembered" information are called adaptive in recognition of their ability to "learn" from previous experience.

A model's final structure is affected by more factors than one might suppose. To name a few, it is affected by:

The purpose of the model.

The accuracy and precision required of the output.

The detail required in the model to achieve the required precision.

The assumptions required at the system boundaries.

The assumptions required within the system boundaries for

status representation

decision parameters

decision rules.

The availability of necessary data.

A model's design is thus complicated by a number of theoretical and practical factors. Theoretical factors determine such things as system boundary interactions and decision rules; practical factors modify theoretical decisions, such as the fineness of detail incorporated in a model. For this reason there must be a feedback loop in the modeling process.

THE MODELING PROCESS

Viewed as an iterative process, modeling takes into consideration the requirements of the model builder and the limitations of his environment. The final model, in both structure and implementation, reflects: the influences of the system being studied, the questions that are to be asked about the system, and the environment in which the model is to perform. Modeling is a constant balancing of costs: data collection costs are balanced against costs (and benefits) of precision, computer program execution costs are balanced against the costs of model reprogramming.

A five-stage iterative sequence describes the modeling process:

Stage 1: Statement of a problem in general system terms. Definition of gross system boundaries. Statement of output(s) needed to solve the problem.

Stage 2: Statement of (initial) assumptions. Definition of static and dynamic system structure. Construction of minimal system model. Assessment of assumptions in light of Stage 1 goals.

Stage 3: Determination of input data requirements and availability. If input data required are not available, modify assumptions and model structure by returning to Stage 2.

Stage 4: Determination of output possibilities. If output is insufficient, modify assumptions and model structure by returning to Stage 2.

Stage 5: Prepare precise specifications for final model. Select a modeling and programming language. Reassess the implications of all assumptions for the future. Prepare a detailed plan for use of the model.

In Stage 1, a problem is discussed in problem-oriented terminology to define a situation as the person with the problem sees it. The tasks at this stage are (1) to define a problem well enough so that it can be expressed in concrete terms and (2) to create an understanding of a problem, as it is seen and as it will be solved, between the person with the problem and the "problem-solver" (who may be the same person). Problems at this level are generally stated first as "Something has to be done about the way the shop is operating," and then, after some discussion, refined to "We have a problem with too large an investment in work-in-process inventory," and finally, after considerable more discussion, narrowed to "What is the right mix of equipment for our workload in the custom-built widget shop?" By a process of gradual narrowing, a problem can be phrased in terms of a limited system (the custom-built widget shop), and stated objectives (find an equipment mix that minimizes work-in-process inventory). When an objective and a gross model are clear, outputs can be described that, when produced by the model, enable realization of the objective.

Stage 2 takes the general system defined in Stage 1 and shapes it into a workable simulation model. As a first step, all recognizable assumptions are clearly stated. These are not all the assumptions that will be present in the finished model since (1) this stage aims at a minimal model, i.e., one containing as little detail as possible, hence minimal assumptions, and (2) assumptions may be present that are only recognizable as such when they are formalized in mechanisms of the final model.

Next, given the object system, the problem statement and the list of limiting assumptions, the structure of the model is developed. The static structure dissects the object system into its component parts, giving each part a unique name and set of characteristics. The dynamic structure describes the way these components act and interact in the system in performing their assigned functions.

Finally, the newly created static and dynamic structures are examined to see if any new assumptions have been made that conflict with the original goals of the system study. If there are conflicts, either the model (static and dynamic structure or both) or the problem statement must be revised. If there are not, the next stage of modeling can begin.

At this stage the model is examined to determine its input data requirements. Until now, data have been ignored, not letting preconceived data structures or systems prejudice the form of the model. Now the model must be examined to see whether it calls for data that (1) do not exist, (2) exist but cannot be collected and analyzed for one reason or another (usually cost or time), or (3) exist but are not usable (incomplete, biased, error-ridden). If this is so, the model must be redesigned to do without these data; usually this calls for new assumptions about the system's behavior. Keeping each data element in mind, the model builder returns to Stage 2 where he can make such changes that are compatible with the total system.

Given a model structure and a source of "good" data, the model must next be examined to "see what it can do." It must be analyzed to see what output it generates, and to determine its characteristics. Usually the output is sufficient and of the proper kind. This is not surprising, since it has been the goal of the modeling project. But often the output is not correct. Due to structural assumptions made along the way or modifications necessitated by input data difficulties, the model may have been changed sufficiently to preclude the generation of certain kinds of data. Also, it is not at all unusual at this point to ask new questions of the model, questions that were not thought of previously and for which output provisions have not been made. The very existence of the model, the fact that people are working with it and thinking about its object system in concrete terms makes this almost a certainty. Again a return must be made to Step 2 to remedy these deficiencies.

In the end either of two things happen, a final model emerges or the original problem (perhaps augmented by now) is declared unsolvable. The latter is not usual, but it can happen if questions are asked that require data that do not exist and cannot be synthesized or that concern system functions whose mechanisms cannot be modeled.

Normally a model emerges. If it is not exactly what was first asked for, it is close to it. It has the property of being minimal for its task, i.e., it has evolved from a simple beginning, with detail and complexity added only when necessary. It has the maximum generality (minimal assumptions built in) for the task for which it is to be used.

This is important, for it is difficult enough to analyze a simple system without complicating the task with extraneous detail.

This is the model that will be programmed for a computer, and, after it is checked out and verified, run under different conditions, studied and treated as if it were a small scale replica of the real system. But before it is programmed, it is best reviewed in the light of future as well as present plans. That this is necessary is a comment on the pragmatics of model building and computer programming. This is a stage where economic reality steps in and has its say on the shape of the final product.

It costs money to program a model and run it on a computer. This money is spent on system analysts, model building, data collection and analysis personnel, computer programmers and statisticians. It is also spent on computer running time, and on peripheral equipment support. Generally, models that are specific and directed to narrow goals are more efficient (with respect to computer running time) than general-purpose models; they are often also cheaper to design. This is due to modeling efficiencies made possible by taking advantage of the structural properties of particular situations. But when new questions are asked of narrow models, when new data elements must be recognized and inserted into model structures, redesign and reprogramming costs must be paid. These costs are often so steep that they are considered as penalties; they are especially steep when new personnel must be recruited for the task.

And so a wise model builder looks to the future. He asks "What might these investigators like to study (look at or alter) next?," and designs models with as much flexibility as he can, wherever he can, without paying expensive computer run time penalties. This cannot always be done, and it is possible that a wrong guess will cost money in the long run, but the nature of simulation and the way models are used, suggests that it is a profitable general principle.

SUMMARY

Simulation is a flexible management tool, enabling almost any system or situation to be studied in great detail. It is also a relatively new and unstudied tool, promising great rewards at the (possible) cost of great penalties.

Hopefully, Secs. I and II have defined simulation well enough so that the reader now sees where it fits in management's toolkit. It is a decisionmaking aid, as are all management science or operations research techniques. Two of its greatest advantages are its almost complete lack of dependence on particular methodological assumptions about the systems, and the close look at system operations its methodology requires. Many a dollar has been made on improvements made to existing systems after careful analysis for simulation model building and data collection brought design faults to light.

IV. AN EXAMPLE OF SIMULATION MODELING

A SYSTEM

The system we use as an illustration is an office in which a number of executives and secretaries work. It might be a real estate office, a division of a life insurance company, a branch of the Federal Government, or a small town newspaper. If the executives are doctors, it might be a medical center or a records room in a large hospital. The executives process incoming correspondence, place and answer telephone calls, and hold conferences throughout a normal working day. Periodically they call upon the secretaries to take dictation, do typing and take charge of miscellaneous affairs.

The office manager has called for a review of the office, with an eye toward personnel reallocations, changes in procedures, and possible automation of selected clerical functions. He has requested that a simulation study be conducted to answer certain specific questions. Being a wise man, he realizes that the performance of the office system must be judged as a whole and has established some system performance criteria that he will use in rating various alternative office system designs. He plans to evaluate these designs through questions he will ask of the simulation model.

The system performance criteria specify that system design A is preferred over system design B if:

(1) Design A allows a greater volume of business to be transacted; or

(2) Design A reduces the time taken to process business transactions for a given volume of business; or

(3) Design A requires fewer secretaries for a given volume of business and transaction processing time.

The ordering of the criteria implies that the ability to accommodate business demands comes first, maximization of executive and secretary discretionary time comes second, and reduction of the secretarial staff comes last.

The office can be viewed as a closed system receiving requests for service from outside its boundaries, processing these requests, and responding to their originators. It is pictured abstractly in Fig. 2. This abstract model, given a frame of reference by a set of questions, provides a starting point for developing a simulation model of the office system.

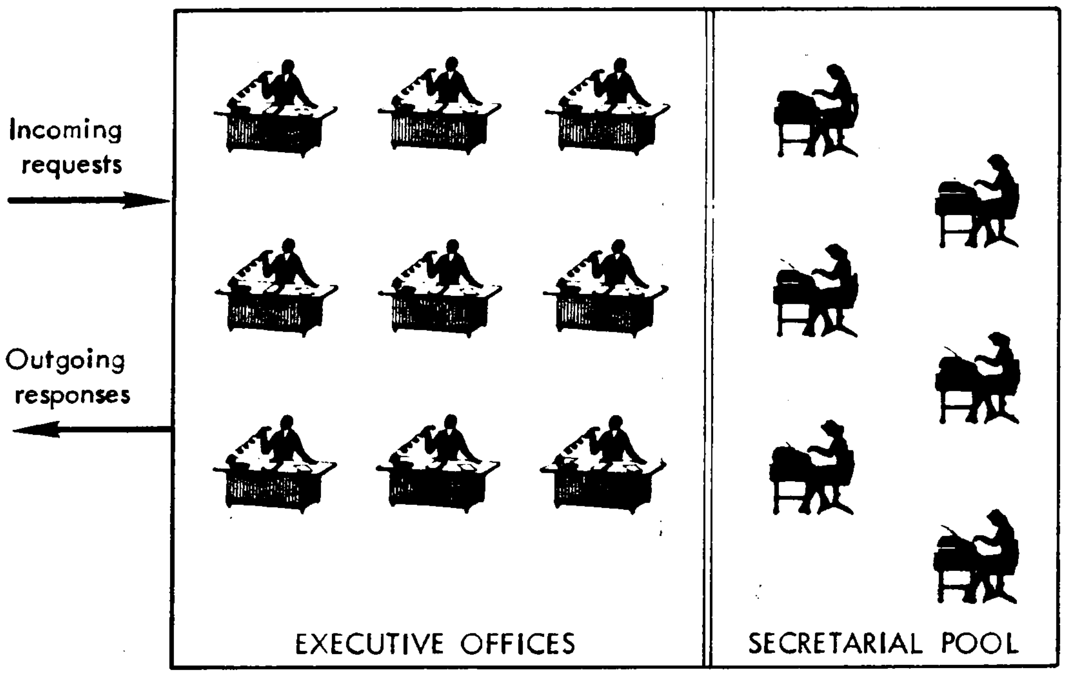

Fig. 2 -- Executive-secretary system

SOME QUESTIONS

There are three general classes of questions that can be asked of any simulation model; these questions relate to:

(1) Demands the environment makes on the model,

(2) The structure of the model,

(3) The parameters of the model.

A system exists to perform certain tasks, whose nature and intensity drive the system. In our office system, the tasks imposed on the executives are determined by incoming transactions. By varying the characteristics of these transactions one can test the system's ability to respond to different workloads. Reasonable <u>environmental questions</u> the office manager might ask are:

> What will happen to system performance if the incoming workload is increased by 10 percent?
>
> What will happen to system performance if the office takes on new kinds of tasks or changes the mix of existing tasks?

Another class of questions relates to the logical structure of a system, its overall layout, information and material flows, and controlling decision rules. Questions asked of simulation models fall into this category when people use simulation to design new facilities, or redesign existing ones. The structure of a model is its most constraining (as well as liberating) force, and questions about system structure are difficult to answer. Some reasonable <u>structural questions</u> the office manager might ask are:

> Can the paperwork flow be redesigned to reduce, by 10 percent, the time an average transaction stays in the office?
>
> What will be the effect of switching from a single pool of secretaries to several small pools assigned to particular men?
>
> Will system performance be improved or impaired by adding an additional afternoon coffee break for the secretaries?

The third class of questions concerns itself with variations in existing system structures. This class is important as it concerns the assessment of maximum system performance levels within existing system design constraints. Exploration of system performance by varying its design parameters is known as sensitivity analysis. Some reasonable <u>parametric questions</u> the office manager might ask are:

> What increase in system performance can be expected from only hiring new secretaries who type at least 100 words per minute?
>
> What will be the effect of trading the existing copying machine for a newer, high-speed model?

Similar questions suggest themselves in most systems. Our aim here is not to suggest particular questions, but to describe the class of questions simulation models can be used to answer, and to impart a flavor for the way a simulation model is constructed. The reader should note how the questions influence the model. For example, one of the above questions assumes that a secretary's efficiency is a function of the time since her last break. A model of secretarial performance must take this factor into account if it is to answer this question.

A MODEL

As we have seen, a model is a formalized theory of how a system operates. Since any formal mechanism used to build a system model exerts its own independent influence we must state at the outset that the model we present is only intended to illustrate the way a model is constructed. Other formal mechanisms will produce other models, some more general, some narrower, some harder to deal with, some easier. All would be constructed the same way, from a dissection of a system into component parts and a formalization of its mechanisms and how they work.

The formal modeling scheme that we use in this example is the logical flowchart. The simulation theory that underlies the model structure is that of discrete-change interaction. Other modeling schemes such as decision tables [15] or relationship graphs [16] could have been used; other simulation theories such as transaction flows, activity cycles or processes [17] could have been utilized. The reader is urged to view this section with an open mind, accepting the example as an illustration and not a prescription. A number of modeling and simulation concepts are discussed in succeeding Memoranda in this series, and the interested reader can refer to these or to the referenced works for additional information. Discussion of them at this point is beyond the scope of this text.

A flowchart is a graphical display of functional activities that take place in a system. Generic functions are indicated by standardized block shapes, with specific operations written inside each block. Some common function blocks used in simulation modeling are pictured in Fig. 3.

a. Computation:

b. Decision flow:

c. Transfer:

d. Flow chart entry:

e. Scheduling of event in simulated future:

f. Selection of previously scheduled event:

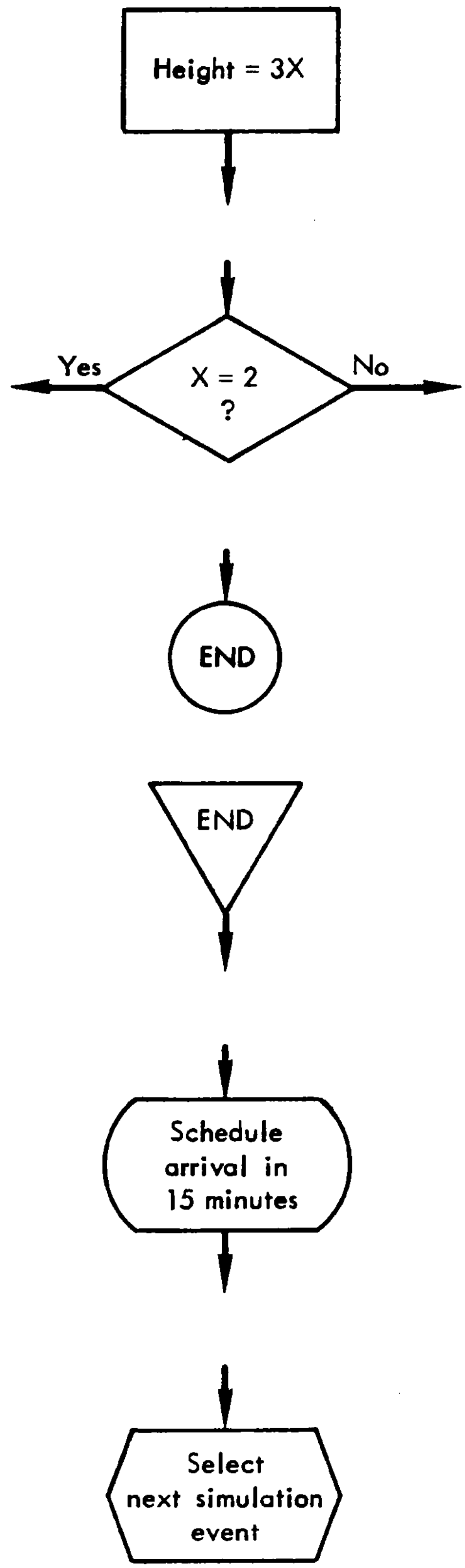

Fig. 3 -- Typical flowchart symbols used in simulation modeling

We use these flowchart symbols in our example. Computations that change the system state directly or are used in changing state or moving the system through time appear in computation blocks (Fig. 3a), comparisons appear in decision blocks (Fig. 3b). If a comparison is true, flow proceeds down the path labeled YES; if false, down the path labeled NO. Transfers between sections of a flowchart and among flowcharts are indicated by transfer blocks (Fig. 3c). Transfers are made when computations do not proceed in sequence within a flowchart or when different flowcharts refer to one another. Flowcharts are entered at entry blocks (Fig. 3d) that indicate the beginning of a flowchart-- they start the path a particular logical flow can take. Scheduling blocks (Fig. 3e) are points in a flowchart where references are made to entry blocks that are not entered immediately, as through a transfer block, but after the passage of an indicated amount of simulated time. When such a block is encountered a "memo" is made in a list that enables the indicated flowchart to be entered at the appropriate simulated time. A "timing mechanism" permits simulated activities to proceed in parallel or in series as the logic of the model indicates. This concept is central to simulation, and is elaborated as we proceed through the steps of model building.

There is only one event block (Fig. 3f) in each flowchart; it acts like a transfer block with one important difference. It does not transfer to an indicated event, but to an event selected from a list that ranks previously scheduled events according to the simulated time when they are supposed to "occur." When an event block is entered, control passes from the flowchart (event) it is in to the flowchart (event) with the smallest scheduled value of simulation time. If this time is different from the current simulation time it is used to advance the "simulation clock." Events that change the clock happen one after another in simulated time; events that leave the clock alone happen in parallel, at the same simulated time. That is, although they are processed sequentially, they are thought of as occurring at the same time.

Overall Model Structure

The model consists of five "events":

(1) A request to perform a task,

(2) Initiation of a task,

(3) Review of a secretarial task,

(4) Availability of an executive at the end of a task,

(5) Availability of a secretary at the end of a task.

Each event is either a starting or ending point of an activity that takes place in the simulated system. Event flowcharts describe time-independent actions that take place in an activity, and call upon one another as simulated time elapses. A basic assumption of discrete-change simulation models is that all system state changes take place at activity boundaries. This is different from continuous-change simulation models that permit state changes to take place continuously as simulated time advances. There are ways to model continuous change in discrete change models, but these are "tricks of the trade" that are not discussed here.*

Simulated time is advanced by what we have called a timing mechanism. Whenever this mechanism receives an instruction to do so, it updates a simulation clock, and transfers to a selected event entry block, taking actions that are supposed to occur at this point in time. As events are processed, state changes take place, the model undergoes dynamic revisions -- the system is simulated.

The remainder of this section is devoted to modeling the office system through these five events, clearing up the notions of simulation, and putting certain important technical considerations in perspective. The discussion is very detailed and the reader is encouraged to get a pencil and paper and work along as he reads.

A Request to Perform a Task

In most systems, jobs (tasks, requests) are initiated in either of

---

*In fact it is hard to find them recorded anywhere and they are usually passed on by word of mouth among professional programmers.

two ways, by mechanisms outside the system or by mechanisms within it. When jobs are initiated externally, they can be viewed as the outputs of a black-box;* these outputs can be regular or periodic, deterministic or predictable in a statistical sense, or irregular and completely indeterminate. When they are initiated internally, it is usually through some logical mechanism whose operations are known to or determined by the system.

When external inputs are regular and known, a mechanism can be built into a simulation model that simulates the regularity. In a sense, the simulation model is constructed so it contains a simulation model of the relevant world beyond its boundaries. In a large number of simulation models, statistical regularities are observed in model inputs (requests, jobs, tasks) and statistical methods used to simulate them. A common method is to define the time between successive arrivals (requests, jobs, tasks) as having some known statistical distribution and to sample within the model to generate arrivals which, though they are unequally spaced and might seem to occur at chance times, observe the arrival pattern either present in the "real world" or deducible from theory. It is sufficient here to recognize the necessity for mechanisms that can produce such data without dwelling on the details of how they do it [18,19].

When system inputs are irregular they can be transmitted to a simulation model directly. This is usually done by putting data on punched cards, paper or magnetic tape and incorporating a mechanism in the simulation model for reading the data. Simulation models are often run with two types of input: real world data to validate a model by testing its behavior in known situations, and generated data to observe its performance in new situations.

---

*A "black-box" is a closed system whose behavior is known, but whose mechanisms are not. We know what it does, but not how it does it. And, by definition, we don't care.

We assume that executives in our office system have two types of tasks: they process incoming communications (invoices, requests for bids, price queries) and handle interoffice correspondence. These tasks are not independent of one another; the former are produced by mechanisms external to the office system, the latter are induced during the course of daily operations. As they result in similar actions we treat both in the same event (flowchart). The event concerns itself with "discovering" a request, and, for requests that have come from outside the office system, determining the time of the next request (either by reading it from a punched card or generating, by statistical means, a time when it will **arrive**). But this is getting ahead of ourselves, since before we start building an office model we must define the system. We do this by defining the objects that "live" in the system and assigning attributes to them. Table 1 lists these objects, which we shall call entities from here on, and their attributes. Once these entities and attributes have been defined, one can almost visualize how the system will be simulated.

Table 1

SYSTEM ENTITIES AND THEIR ATTRIBUTES

Executive	Secretary	Task
Position	Skill in typing	Type
Manager	words/minute	Invoice
Senior	errors/100 words	Price quotation
Junior	Skill in dictation	Bid
	words/minute	Telephone
	errors/100 words	Dictation
	Skill in office work	Typing
	General rating 1-100	
State	State	Characteristics
Busy	Busy	Time
Available	Available	Length
On-break	On-break	

Given the static structure defined in Table 1, the nature of the request event, and some logic not yet described, we can construct a flowchart model of the actions that take place when a request enters the system. This model is illustrated in Fig. 4. Numbers to the left of each flowchart block refer to comments in the body of the text that describe the operations that take place within the block.

Block 1 is the entry point to the flowchart. It contains a name that will be used in subsequent flowcharts to refer to the "task request" event. The directed arrow leading from it is a symbol commonly used to indicate a path and direction of flow.

Block 2 is a decision block that splits the logical flow depending on the kind of request that has just occurred. To understand how this block operates we must understand the concept of an event occurrence.

An event occurs when its "time arrives," the time having been previously recorded by an internal scheduling block or observed on an input data card. The precise mechanisms that accomplish these tasks differ among simulation programming packages and need not be stated here. It suffices if the reader understands that there is some mechanism operating in the background of a simulation program, observing data cards and previously scheduled events, ordering them by their event times and "popping them up" when their time arrives. This in fact is the function of the event selection block. The reader will notice that every event terminates with a direct transfer to another event, or with an event selection block. It is in event selection blocks that time discriminations are made, events sequenced properly in time and the simulation clock advanced.

When a request event is popped up the simulation program has access to information associated with it, e.g. how it was caused. The model is able to look at this information and take action on it.

If the request is for an internally generated task, the flowchart leads directly to Block 5 where a question is asked to see if office workers are available to process the request. If the request is for an externally generated task, the program pauses in Block 3 to compute (according to some statistical time distribution) or read (from a data card) information about the next arrival.

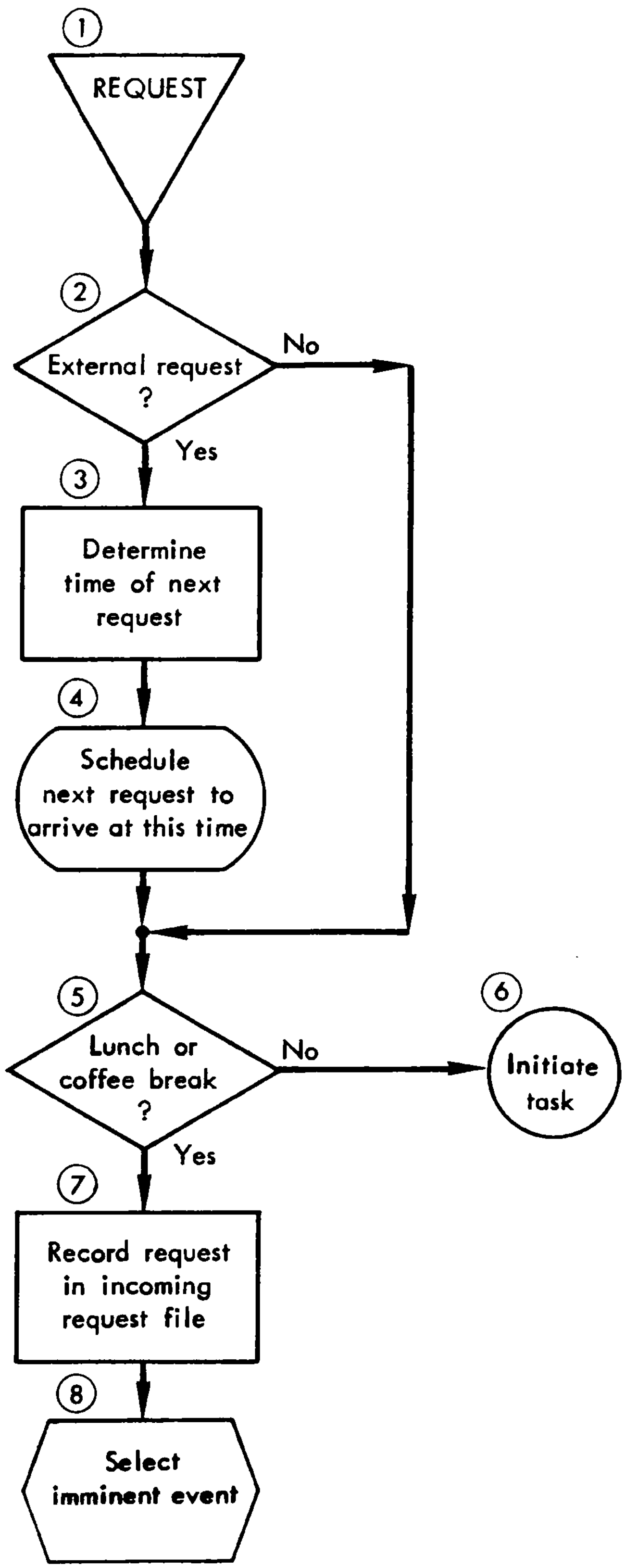

Fig. 4 -- Event Number 1: Request to perform a task

Block 4 schedules the arrival of the next externally generated request. When it does so it records a memo of a request arrival and its time on a calendar of events scheduled to occur. This calendar is part of the selection mechanism employed in sequencing events and advancing simulation time.

By the time the program arrives at Block 5 it is through with scheduling future events and is concerned with processing the request that has just arrived. Since real offices do not work continuously, but pause for lunch and coffee breaks during the day, the model asks in Block 5 if such a period is in progress. If it is, the request cannot be processed immediately but must be filed for later handling. If the request can be processed, Block 6 transfers control to the event that does so.

Block 7 records a request that cannot be handled in a backlog file; it might be an in-basket in real life and a table or list structure in a computer program. The file entry is made so that when the office workers return to their desks they see that tasks accumulated while they were gone.

Block 8 directs the simulation program to select an event from the time-ordered file of scheduled events. It might be another request or the completion of a previous task. When the next event is selected it may or may not indicate a simulation time advance. If it does not, we think of it and the event just completed as occurring simultaneously; although they are processed in series on the computer there is no time advance and they are considered as happening at the same time.

Initiating a Task

Once the system has accepted a request, a match must be made between it and the resources needed to fill it. A search is first made for an executive. If one is found who is free and can handle the request, a secretary is procured if needed. The logic for this event is shown in Fig. 5.

Block 1 as always is an entry block giving the symbolic name of the event.

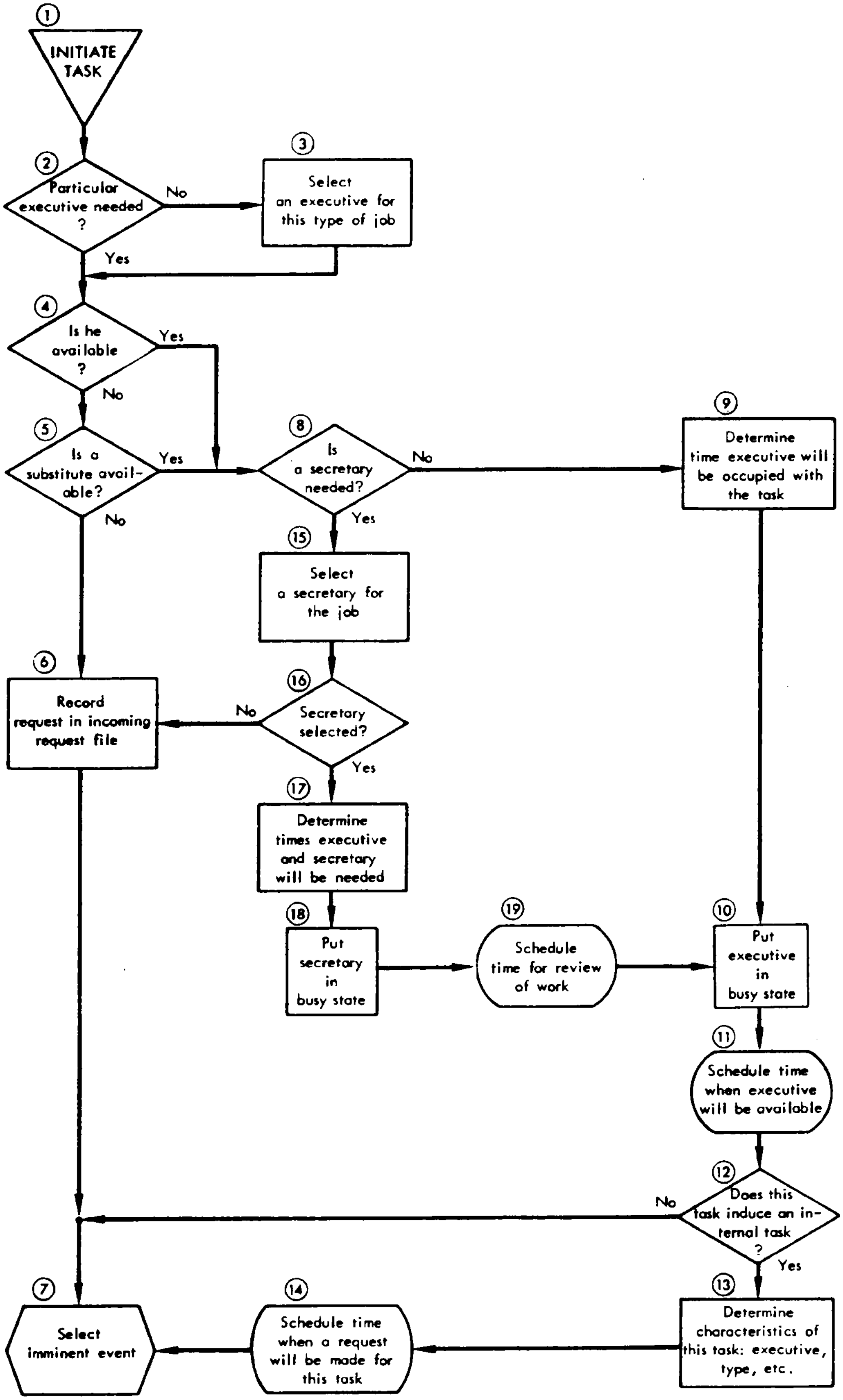

Fig. 5 -- Event Number 2: Initiation of a task

Block 2 starts the match between a task and its resources by asking if the request just entered calls for a particular executive, e.g., a telephone call for a certain person or a request for a price quotation from a specialist in a certain area. If no particular executive is called for, Block 1 passes flow to Block 3 where an executive is selected. If a certain person is requested, flow proceeds to Block 4 where a test is made to see if this person is available.

Block 3 is typical of a functional block whose description is short but whose programming content might be large. A procedure to select an executive can be brief, e.g., managers can do everything, senior executives can do everything except give price quotations, junior executives can only answer the telephone; or it can be long and elaborate, e.g., an executive is selected whose personal qualifications as listed in his personnel file match the requirements of the task according to a complex and computationally intricate formula. Many of a simulation model's key assumptions are built into blocks such as this. In this report we leave this block in its present "macro" level; in a subsequent Memorandum we present a computer program for simulating the office system and expand the block into detailed logical elements.

When an executive is selected, Block 3 transfers to Block 4, the block to which control is passed if a particular executive is called for.

Block 4 asks if the executive requested in Block 2 or selected in Block 3 is available. It does so by examining the executive's state (status code); if the code is "available" the executive is free to handle the request, if it is "busy" or "out on break" he is not. Once again, as in Block 2, the flow logic is split depending on the answer to this question.

If the selected executive is available, flow passes to Block 8 where processing of the task begins. Before we consider these actions we should discuss what happens if the executive is not available.

Block 5 asks if a substitute is available for a busy executive, implying that a substitution can be made and that a procedure exists for finding one. This situation is a little like that of Block 3 where an executive is selected for a particular type of task. Block 5 could be expanded to a series of blocks describing a procedure for selecting a substitute, testing for his availability, selecting another substitute if necessary and so on until all possible candidates are tried and accepted or rejected. In our simplified model we do not state this logic. We only indicate that if a substitute cannot be found, control passes to Block 6 which files the unprocessed task.

Block 6 of this event is identical to Block 6 of the request event; it files information about the request for later processing. This block appears in the simulation model whenever a request cannot be processed and must be "remembered."

In Block 7 control is passed via an event selection block back to the "timekeeping" mechanism of the simulation program. Since the current request cannot be processed, the model must look at its calendar of scheduled events to determine what to do next.

Returning to the case where an executive is available to process a request, we ask next in Block 8 if a secretary is also needed. This will be true if the request is for dictation or for some task where instructions must be given; it will not be true if the task is simply answering a telephone call. This question can be answered in a number of ways in an operating computer program; as with most questions of this type we leave the description of decisionmaking at the macro level, namely, that a decision has to be made.

Block 9 starts the flow path for the case where a request can be honored by an executive alone; it determines the amount of time he will spend on the task. As with arrival times this can be computed, read in as a constant value for all tasks of a given type, or read in along with the request for the task. These are options that are left to the operational computer program.

Block 10 puts the executive in the "busy" state so that he cannot be called on to do another task while he is working on this one. He will remain in this state until the "executive available" event occurs; this is scheduled in Block 11 to happen after the lapse of the previously determined amount of time.

Before proceeding with the simulation, the model must ask if processing this task, e.g., answering a phone call, induces another task, e.g., writing a memo. This is done in Block 12. If a task is not induced, flow passes to Block 7 where the model is instructed to select another event and proceed with the simulation. If a task is induced, Block 13 determines its characteristics and passes them on to Block 14 where the induced task is scheduled to be requested. Flow then proceeds to Block 7.

If, back in Block 8 we found that a secretary was needed to work along with the executive, control would have passed to Block 15 where a secretary must be selected before a task starts. Block 15 probably employs logic similar to that used in selecting executives for tasks. This logic can pair a particular secretary with an executive, pool all secretaries so that they are available to all executives, or employ some intermediate scheme. As was done in selecting an executive, when a secretary is chosen her status code must be tested to see if she is available.

Block 16 performs this test. It, like Block 5, can be considered as a macro block in which alternatives and availabilities are tested until a decision is reached. If a secretary is not available, a request cannot be processed and must be filed along with other unprocessed requests.

Once a secretary is found, Blocks 17, 18 and 19 determine the time the executive and secretary will spend on the job, put the secretary in the "busy" state, and schedule the time when her work will be reviewed. It is not necessary that the executive and the secretary work together on the task for the same period of time; separate events are provided to schedule their release from the task at different times. The release times can, however, be the same if the task is a cooperative effort.

Block 19 transfers control to Block 10 after completing its function, picking up at a part of the flowchart that we have already seen. The reader should be able to see why and how this was done.

Review of a Secretarial Task

One of our office rules is that every task a secretary performs must be reviewed. When a secretary finishes a task she brings it to the attention of the executive who initiated it. If he is not available, she waits.* If he is available, he reviews the work and either accepts it or notes corrections that must be made before another review. The logic of the review event is shown in Fig. 6.

Block 1, as usual, names the event. Block 2 asks a question about executive availability and transfers to Blocks 3 or 5 depending on the answer.

Block 3 records the review task in the executive's incoming work file if he is busy. The task is filed along with incoming requests that were filed for reasons we saw in previous flowcharts. Block 4 calls on the simulation timing mechanism to select the next scheduled event. The secretary is not assigned an "available" state, but remains "busy," waiting for the executive to become free and review her work.**

Block 5 is another macro block, hiding what might be an enormous amount of logic behind the label "executive review of secretary's work." This block will most likely contain different review criteria for each kind of task a secretary can perform, one for dictation and typing, one for filing, etc. The result of a review will be Yes if the work is satisfactory or No with a list of corrections if it is unsatisfactory.

Block 6 branches on the previously computed Yes or No. If the task has been done satisfactorily, the secretary is put into the "available" state in Block 9, making her eligible for new tasks, and scheduled to come available immediately (Block 10). An event scheduled to occur with no increase in simulation time will probably be executed at once when the event returns control to the timing mechanism (Block 4). At least it will compete for immediate processing with other events scheduled at the same time, i.e. occurring in parallel.

---

*This may not be good office practice, but is a feature of our example.

**Cf. above footnote.

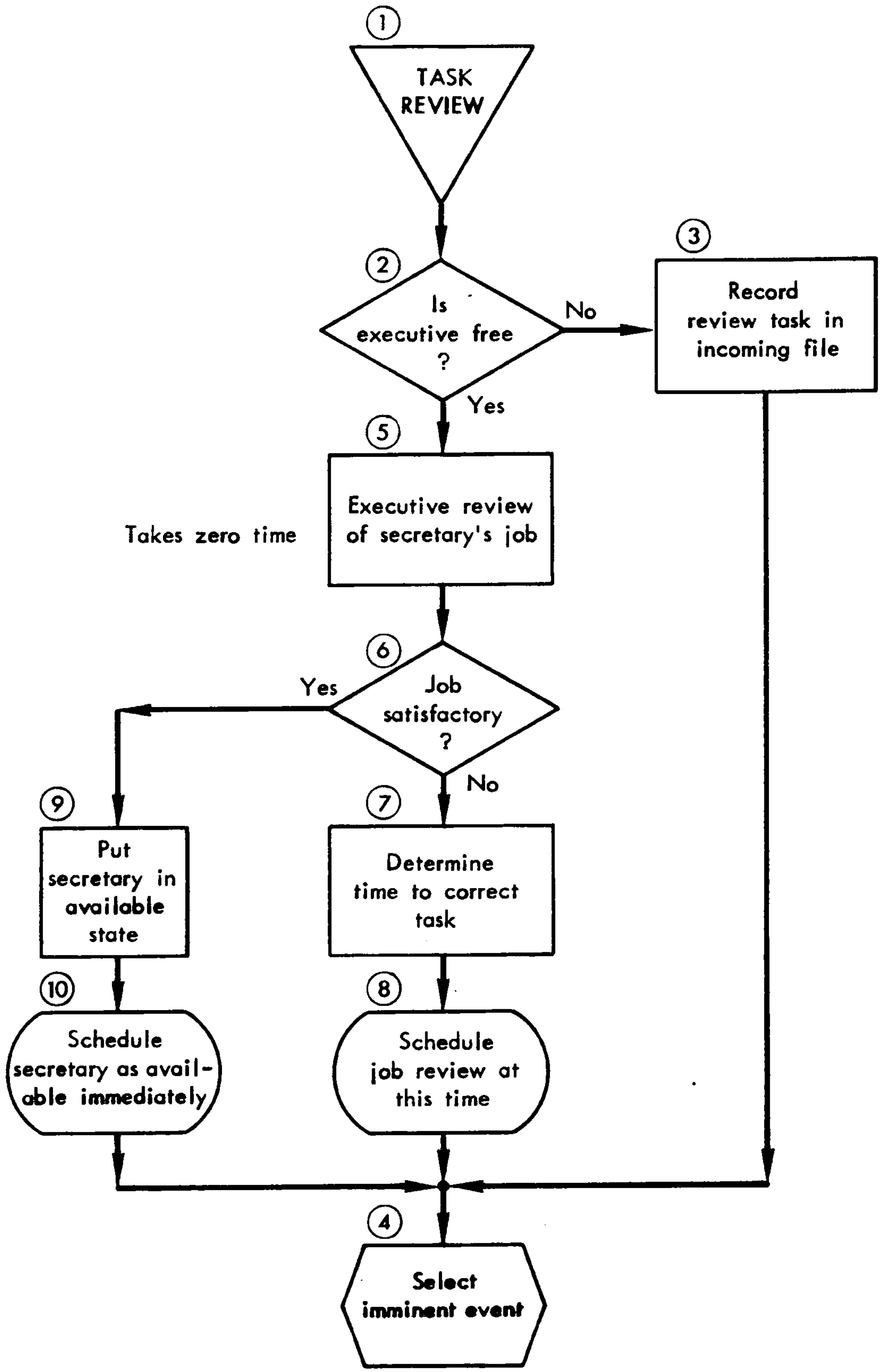

Fig. 6 -- Event Number 3: Review of a secretarial task

An unsatisfactory task has its correction time computed in Block 7 and review rescheduled in Block 8.

Important things to notice about this event are its hidden basic assumptions--a review task takes no time and a secretary stays with a job until it gets reviewed. These assumptions can be easily changed to allow secretaries to do other work while waiting for reviews, cause executives to spend time making large-scale corrections, etc.

Executive Available at the End of a Task

This event marks the completion of an executive activity. It returns an executive to an "available" state and determines his next action: another task, a break for coffee or lunch or an idle (discretionary time) period. The event logic is shown in Fig. 7.

Block 1 names the event. Block 2 puts the executive in an available state and asks questions about the next executive action. These questions are asked in a specific order and imply certain things. From the logic of the event we see that a lunch or coffee break cannot start until a current job is completed, but will be taken when it is due regardless of task backlogs. This is important as it assumes a priority sequence imposed by the order in which questions are asked and not by explicit priority statements.

Block 3, which decides if a break is due, also contains hidden logic. When one considers the connections between events and the way in which the model operates, he sees that if an executive is idle (in the "available" state) and a break time occurs, there is no mechanism that alerts him of this. By the way the model is constructed, breaks can only be taken after the completion of jobs. This will have little practical effect if: (a) the work rate is high in the office so that there are no long periods of idle time possible or (b) the logic of Block 3 looks ahead and starts a break early if one is almost due. This small difficulty has been put in the model to acquaint the reader with problems that can occur when one sets out to build a model from scratch.

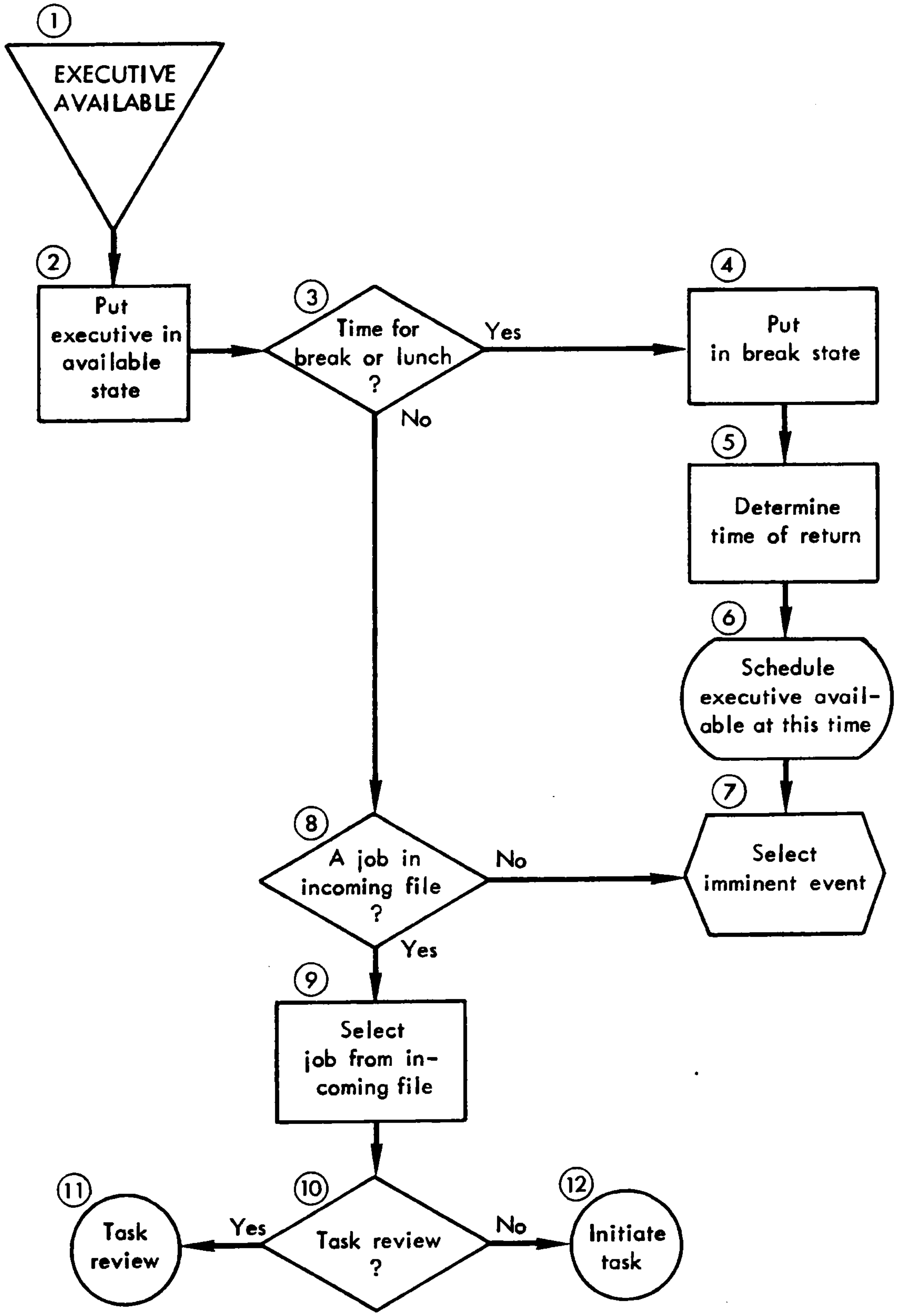

Fig. 7 -- Event Number 4: Availability of an executive at the end of a task

If a break is due, Block 4 places the executive in the "break" state, Block 5 determines its duration, Block 6 schedules the executive's return to an availability condition (by executing this same event some time in the future after the simulation clock has advanced past the break point), and Block 7 returns control to the event selection mechanism.

If a break is not due, the model must decide whether the executive should be left in the available state or assigned to a waiting task. It does this by looking, in Block 8, at the file in which we have been putting requests that could not be processed. If the file is empty the executive is left alone and control passed to Block 7 to select the next event.

If the file is not empty the model must select a job. If there is only one job in the file there is no problem. If there is more than one there is a conflict situation that must be resolved. Conflict is usually resolved by priority rules that assign values to different types of jobs; a job is selected that has the highest (or perhaps lowest) value. In cases with ties, multiple ranking criteria are used. Possible criteria that might be used in this model are: time a job arrives in the system, skill level required to process a job, etc. The issue of selection rules is a complex one and a model that merely says "select a job" hides a lot of work that must be done to develop an operating model. For example, there are few organizations that have well articulated and formalized priority rules, and a modeler may have his hands full merely trying to find out the "rules of the game."

Once a task has been selected, however, it is a relatively simple thing to route the executive to the proper flowchart to process it. This is shown in Blocks 10, 11 and 12.

<u>Secretary Available at the End of a Task</u>

This event is similar to event 4 in both its intent and its form. When a secretary is released from a task she becomes available and is either sent on a break, put on a backlogged job or left idle, depending

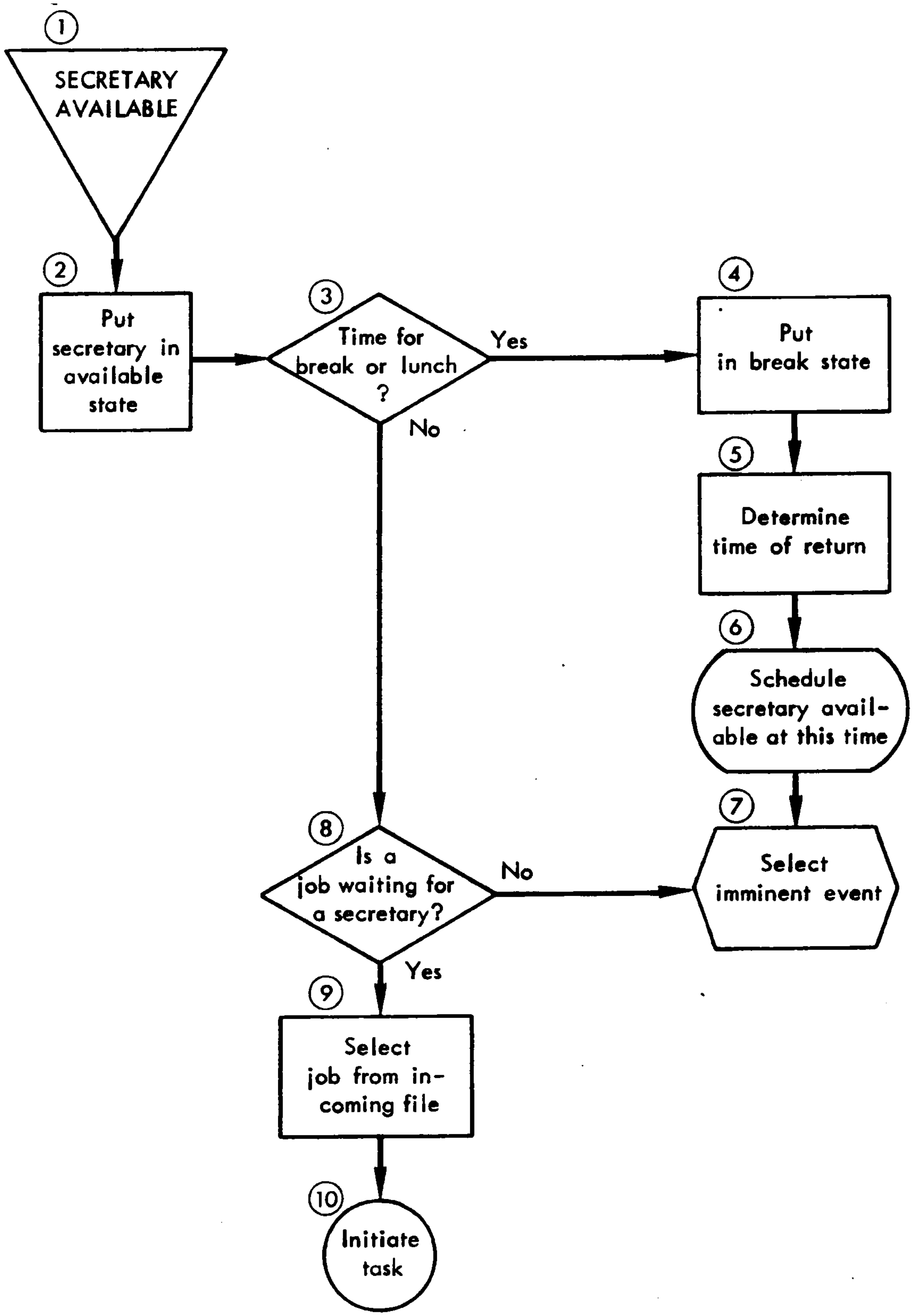

Fig. 8 -- Event Number 5: Availability of a secretary at the end of a task

on current conditions. Blocks 1 through 10 in the flowchart of Fig. 8 correspond to similar blocks in Fig. 7 and need not be commented upon.

SUMMARY

A model is a mechanism for reproducing a system's performance. When described in a formal manner, e.g., through flowcharts, a model takes on a descriptive flavor that aids in understanding the processes at work in a system. For this reason people are constantly at work devising new modeling schemes [15,16,20].

In this section we have presented a simple system, posed some questions that might be asked about its behavior and presented a model that, with some elaboration, can be implemented on a digital computer. We have indicated that this model is primarily illustrative, one of many that could be designed and formalized to serve the same ends. We have tried to keep the model simple, hoping to preserve the overall system structure in the reader's mind, since over-elaboration can prevent a reader from seeing an underlying model structure. Yet we have incorporated a moderate amount of complexity to keep the model from being completely trivial.

At this point we hope that an interested reader will take some time out to think about the model, about the various technical problems implementing it might present, about the kinds of data he would need to use it, and about the way in which he would go about doing so. We strongly suggest that he make up a set of sample data, e.g. a set of executives and secretaries of various capabilities and a list of requests to be input at different times, and work through the model. It is only by attempting a simulation, even for a few events, that a true feeling of "what it is all about" is formed. And it is only through such an exercise that all of a model's hidden and unstated assumptions are exposed. We have left some for the reader to find.

V. USING A SIMULATION MODEL

The basic concepts and techniques of simulation should now be clear. A Glossary of simulation terminology is provided in the Appendix for readers who wish to review the concepts we have discussed.

While it is now possible for a reader with little or no previous background to understand, evaluate and possibly construct a systems simulation model, he will find when he attempts to do so that his difficulties have just begun. While he has mastered the "concept barrier" he has an even higher "technical barrier" before him. This section highlights some technical simulation problems. We will not discuss them in any detail, for that is the purpose of other Memoranda in this series, but we do make them known and draw the reader's attention to their importance.

SIMULATION PROGRAMMING LANGUAGES

Traditionally, it has been a difficult and expensive task to go from a flowchart model, such as we have presented here, to a running computer simulation program. It has been difficult because communications are difficult between model builders and computer programmers, because there are few widely available computer programming languages rich enough to express simulation concepts in a natural way, and because simulation requires special mechanisms that ordinary programming languages do not have. It has been a long road for most large scale simulations and a tribute to their creators' patience and perseverence that they have been done at all.

The situation is better today. A number of special-purpose simulation programming languages exist that are designed for aiding model design, reducing computer programming difficulties and guiding model builders in the use of models. Most of these languages have been designed with both the nonprofessional programmer and the computer specialist in mind. Simulation programs written in these languages are intelligible to wide audiences and serve as model documentations as well as computer codes [21].

A person interested in doing computer simulation must therefore know about computer programming languages. If he does not plan to do programming himself he must still be able to understand different simulation language concepts, for, as we have pointed out, it is these concepts that determine the final structure of a simulation model.

DATA ANALYSIS

We have seen the impact of data on a model. Not only do they determine the accuracy of a model, but their presence (or absence) determines a model's structure. Before a simulation model can be completed, all the data elements used in it must be examined to determine their availability and quality. If they are available they must be collected and analyzed, and hypotheses developed about them. Is a certain data item a constant or should it be sampled from a statistical distribution? If it is a constant what is its value? If it is a random variable, what is its distribution? How does one sample from such a distribution? These and similar questions manage to make data analysis one of the most important tasks in "getting a simulation to market."

Surprisingly little has been written on this subject. Much can and should be said. A model is only as good as its data; one cannot underestimate the importance of data or treat them casually.

MODEL VALIDATION

A model must not only contain sound data when it is run, it must also be sound structurally. Before a model can be used it must be tested to see that it does conform to the system for which it was designed. It must be examined to insure that it responds as it should, and that it performs as the real system would under different stresses, data inputs and subsystem configurations.

Since real systems operate dynamically in time, simulation models must be tested to insure that they behave the same way. It is rarely enough that a model gives answers that one can observe in the real world. Unless one is certain that a model behaves like the real world, he cannot be sure that answers obtained under slightly different conditions will be usable.

A model of an existing system must be validated to be used. A validated model is one whose results bear a measurable (absolute or relative) relationship to results obtained in the real world. Analysis of dynamic response and calibration go hand-in-hand; the former looks at "how you got there," the latter at "where 'there' is."

Model validation requires the application of a substantial amount of statistical know-how. The typical simulation model is a complex of interlocking and interacting statistical relationships; a model user must sort out and understand them. It is not often an easy task. A body of literature is just forming that gives a model builder the knowledge necessary to do it well [5].

EXPERIMENTATION AND ANALYSIS

Given a working, verified and validated program, one must decide how to use it. In the main there are two types of questions people are concerned with: (1) comparisons of alternatives and (2) studies of system response over a range of parameter settings.

Comparisons are made when people want to discriminate between the performance of a system under different operating policies, decision rules or parameter settings. Does a first-come, first-served service policy result in greater customer satisfaction than an earliest due-date policy? Is there a degradation in system performance if a particular service rate is reduced 10 percent?

System response studies are made: to find parameter settings that produce the best system response, to determine the sensitivity of response around these optimum settings, and to determine the shapes of response curves and surfaces. How many elevators do I need to have no prospective passenger wait more than five minutes? How fast must these elevators travel? Is the customer service a linear function of the number of elevators or is the relationship more complex?

These and other questions must be answered on sound statistical terms. In most cases classical statistical procedures cannot be used because the simulation data do not satisfy assumptions necessary for their application. For example, the presence of autocorrelation in most time series gathered by simulation models makes it difficult to determine how long to run a model to estimate, with predetermined accuracy, the average value of the series. A literature is evolving that treats these and other questions, and a simulation user must familiarize himself with certain statistical techniques before he attempts to draw inferences from simulation studies [4,22]

Appendix

GLOSSARY OF SIMULATION TERMS*

ACCURACY	Closeness of a model's response to response observed in the real world.
ACTIVITY	A system function that usually takes time to accomplish and results in a change in system state, such as moving from one place to another or repairing a defective part.
AGGREGATION	Representing several variables or factors by one combined factor.
ANALOG MODEL	A model in which one set of properties is used to represent another.
ANALOG SIMULATION	A simulation executed by an analog computer (useful, in general, for simulating models expressed as differential equations).
ASSUMPTION	A fact or statement taken for granted.
ATTRIBUTE	A characteristic of an entity, such as the age of a man or the length of a waiting line.
COMPRESSION OF TIME	Simulating a given period of time in a shorter period. (Time in a simulator does not necessarily proceed at the same rate as real time. In general, time is compressed; a unit of real time passes in much less time in the simulator; e.g., one year is compressed to, say, ten minutes.)
CONTINUOUS-CHANGE MODEL	A simulation model in which state changes occur continuously as time progresses.
CONTROL VARIABLE	See PARAMETER.
DECISION RULE	A rule that defines a method for choosing one of a number of alternatives based on the values of factors at the time the choice is made. The factors can be deterministic or stochastic.
DESCRIPTIVE MODEL	A model in the form of a narrative description of a situation.
DETERMINISTIC MODEL	A model in which every factor is uniquely determined when the factors to which it is related are determined.

*This Appendix is based on a Glossary prepared by Stanley S. Reed (IBM) and Roger L. Sisson (University of Pennsylvania) and submitted to the Workshop on Simulation held at the University of Pennsylvania, March 17, 1966.

DIGITAL SIMULATION	A simulation executed by a digital computer.
DISCRETE-CHANGE MODEL	A simulation model in which state changes occur only at discrete points in time.
DISCRETE VARIABLE	A variable whose value changes in steps, such as whole numbers, rather than continuously.
DYNAMIC SYSTEM	A system whose state changes with time during its normal operation.
DYNAMIC STRUCTURE	The time-dependent structure of a model; the rules for moving a model from one system state to another.
ENTITY	Any distinguishable item, being, or processing unit within a model.
ENVIRONMENT	The surroundings in which a model is embedded. A model usually "sees" its environment through parameter settings and assumptions made about its static and dynamic structure.
ESTIMATION	Determining the value of a system performance measure by experimentation.
EVENT	An instant in simulated time at which a change to a new system state can take place. A computer program describing how system state changes take place. An activity is always bounded by two events: start activity and stop activity.
EVENT TIME ADVANCE	A method of time advance where time is incremented from one event time to the next, which may be an increment of several units of simulated time.
FLOW CHART	A symbolism for representing sequential procedures.
INCREMENTAL TIME ADVANCE	A method of time advance in which time is incremented unit by unit in uniform steps.
MATHEMATICAL MODEL	A model formulated in accepted mathematical symbols such that there is a mathematical structure that permits the manipulation of these symbols in useful ways.
MEASURE OF PERFORMANCE	A quantity whose value can be used to judge how well a system is operating.
MINIMAL MODEL	A model of a system containing the fewest structural and data assumptions that is sufficient for a given purpose.
MODEL	A representation of an existing or proposed system.

MONTE CARLO MODEL	See STOCHASTIC MODEL.
OPTIMAL CONDITION	A condition in which no change in a controlled variable can effect an improvement in the measure of system performance.
PARAMETER	A value which, when altered, changes the input to a system. Parameters are system control variables.
PHYSICAL MODEL	A model whose components are represented by physical processes.
PRECISION	The degree of refinement with which a measurement is stated. Associated with the reproducibility of estimates made from samples.
PREDICTION	Determining a possible future value of a system state or performance measure.
PROCESS	An instance of an activity during a simulation that can last for a period of time.
PSEUDO-RANDOM NUMBERS	Numbers produced by a deterministic method that in many respects behave like random numbers.
RANDOM NUMBERS	Numbers, usually uniformly distributed between 0 and 1, that occur in such a way as to be completely unpredictable.
SIMULATED TIME	Time as represented within a simulation; usually expressed in terms of a basic unit such as a second, a day or a week.
SIMULATION	The manipulation of a system's model to reproduce its operations as it moves through time.
SIMULATION CLOCK	A counter used in a simulation model that acts as a clock as simulation time advances.
SIMULATION LANGUAGE	A formal terminology and set of programming statements that can be used for, and that facilitates, the construction of a simulation model and computer program.
SIMULATION PROGRAM	A computer program representing a specific simulation model or, by parameters, a class of models.
SIMULATION SOFTWARE	A computer program package that translates a model expressed in a simulation language into a computer program that can be executed on a computer. (The software may include a translator, statistical generators, libraries of preprogrammed routines, input and output editing routines, etc.)

SIMULATOR	A device for executing a simulation model dynamically; a Simulation Program in a computer ready to operate is a Simulator, so is a scale model in a wind tunnel.
STATE, STATUS	It is useful to think of the world as being in one of a number of possible situations, i.e., all data entries describing a system have particular values. Each of these situations is called a state. A simulator is a device for determining a future state of a system given its present and past states. Creating a model is, in part, a process of determining possible states.
STATIC STRUCTURE	The time-independent structure of a model; the framework within which system states are defined.
STOCHASTIC MODEL	A model that has stochastic variables.
STOCHASTIC VARIABLE	A random variable, i.e., a variable whose value is not uniquely determined by factors to which it is related but that varies in some statistical way.
SYMBOLIC MODEL	A model defined by arbitrary symbols and rules for manipulating them.
SYSTEM	An interacting collection of processes in a closed environment, the boundaries of which are clearly stated.
SYSTEM RESPONSE	A value or sequence of values determined by the behavior of a system as it operates over time.
VALIDATION	The process of assuring oneself that a model is a reasonable and satisfactory representation of a system.

REFERENCES

1. Geisler, M. A., and J. W. Petersen, "The Costs of Alternative Air Base Stocking and Requisitioning Policies," Naval Research Logistics Quarterly, Vol. 2, Nos. 1 and 2, March-June, 1955, pp. 69-81.
2. Markowitz, H. M., B. Hausner and H. W. Karr, SIMSCRIPT: A Simulation Programming Language, The RAND Corporation, RM-3310-PR (DDC No. AD 291806), October 1962.
3. Conway, R. W., Some Tactical Problems in Simulation Method, The RAND Corporation, RM-3244-PR (DDC No. AD 287527), October 1962.
4. Fishman, G. S., Problems in the Statistical Analysis of Simulation Experiments: The Comparison of Means and Lengths of Sample Records, The RAND Corporation, RM-4880-PR (DDC No. AD 628626), February 1966.
5. -----, and P. J. Kiviat, Spectral Analysis of Time Series Generated By Simulation Models, The RAND Corporation, RM-4393-PR (DDC No. AD 612281), February 1965.
6. Kamins, M., Two Notes on the Lognormal Distribution, The RAND Corporation, RM-3781-PR (DDC No. AD 415360), August 1963.
7. Sherbrooke, C. C., Discrete Compound Poisson Processes and Tables of the Geometric Poisson Distribution, The RAND Corporation, RM-4831-PR (DDC No. AD 637591), July 1966.
8. Ginsberg, A. S., and B. A. King, Base Operations in Maintenance Simulation, The RAND Corporation, RM-4072-PR (DDC No. AD 605248), August 1964.
9. Ginsberg, A. S., H. M. Markowitz and P. M. Oldfather, Programming By Questionnaire, The RAND Corporation, RM-4460-PR (DDC No. AD 613976) April 1965.
10. Sharpe, W. F., The Army Deployment Simulator, The RAND Corporation, Abridged RM-4219-ISA, March 1965.
11. Smith, T. C., SAMSOM: Support Availability Multi-System Operations Model, The RAND Corporation, RM-4077-PR (DDC No. AD 601813), June 1964.
12. Voosen, B. J., PLANET: Planned Logistics Analysis and Evaluation Technique, The RAND Corporation, RM-4950-PR, January 1967.
13. Krupp, S., Pattern in Organizational Analysis, Holt, Rinehart and Winston, Inc., New York, 1961.
14. Teichroew, D., and J. Lubin, "Computer Simulation -- Discussion of the Technique and Comparison of Languages," Communications of the ACM, Vol. 9., No. 10, October 1966, pp. 723-741.
15. DETAB-X, Preliminary Specification for A Decision Table Structural Language, CODASYL Systems Group, 1962.
16. Frye, W. H., The Interrelationship Graph (IRG) as a Technique for Computer Model Development, Navy Electronics Laboratory, SF 001 0102, Task 11276 (NEL Z21761), November 1965.

17. Krasnow, H. S., and G. P. Blunden, The Process Concept as a Basis for Simulation Modeling, The IBM Corporation, ASDD Technical Report 17-181, November 1965.

18. Tocher, K. D., The Art of Simulation, D. Van Nostrand Company, Inc. Princeton, New Jersey, 1963.

19. Naylor, T. H., J. L. Balintfy, D. S. Burdick and K. Chu, Computer Simulation Techniques, John Wiley and Sons, Inc., New York, 1966.

20. Holt, A. W., and R. M. Shapiro, N-Theory, Technical Documentary Report #1, A Mathematical Method for the Description and Analysis of Discrete, Finite Information Systems, Applied Data Research, Rome Air Development Center Contract AF 30(602)-3324.

21. Kiviat, P. J., "Development of Discrete Digital Simulation Languages," Simulation, Vol. 8, No. 2, February 1967, pp. 65-70.

22. Fishman, G. S., Digital Computer Simulation: The Allocation of Computer Time in Comparing Simulation Experiments, The RAND Corporation, RM-5288-PR, April 1967.

23. Churchman, C.W., R. L. Ackoff and E. L. Arnoff, Introduction to Operations Research, John Wiley and Sons, Inc., New York, 1957.