1981 Winter Simulation Conference Proceedings
T.I. Ören, C.M. Delfosse, C.M. Shub (Eds.)

SADT (TM) /SAINT: LARGE SCALE ANALYSIS SIMULATION METHODOLOGY

Kenneth H. Evers
SofTech, Inc., Dayton, Ohio

Robert F. Bachert
Air Force Aerospace Medical Research Laboratory
Wright-Patterson Air Force Base, Ohio

Patrick R. Santucci
SofTech, Inc., Dayton, Ohio

ABSTRACT

SADT/SAINT is a highly structured, top-down simulation methodology for defining, analyzing, communicating, and documenting large-scale systems. Structured Analysis and Design Technique (SADT), developed by SofTech, provides a functional representation and a data model of the system that is used to define and communicate the system. System Analysis of Integrated Networks of Tasks (SAINT), currently used by the USAF, is a simulation technique for designing and analyzing man-machine systems but is applicable to a wide range of systems. By linking SADT with SAINT, large-scale systems can be defined in general terms, decomposed to the necessary level of detail, translated into SAINT nomenclature, and implemented into the SAINT program. This paper describes the linking of SADT and SAINT resulting in an enhanced total simulation capability that integrates the analyst, user, and management.

INTRODUCTION

Simulation techniques are used to model or duplicate the behavioral aspects of real-world or conceptual systems without using the actual system. This is a useful method of analyzing or evaluating a system without requiring the time and cost of building or modifying the actual system. The development of a simulation model requires a system description in the form of a functional model which is combined with timing and precedence requirements to form a dynamic computer simulation model.

Current simulation techniques, however, are highly dependent upon the experience and skill of those applying them and do not provide a good communication tool. Therefore, there is not a complete understanding among the analysts, management, and the user as to what the system is, what the purpose of the simulation is, how well the model represents the system, and what the simulation results mean. This lack of communication results in the analysts having to define the system as he understands it, to develop the simulation to solve the problem as he understands it, and to make conclusion from the simulation results as he interprets them. Like most software development projects, it is the poor and incomplete definition of the problem at the start that results in a high percentage of problems encountered during the final stages.

The SADT/SAINT simulation methodology provides the techniques necessary to define or bound the problem, to develop a validated functional model of the system, to build a simulation model from the functional model, and to communicate the simulation results via the functional model. Structured Analysis and Design Technique (SADT) is a structured technique for doing system analysis and design. System Analysis of Integrated Networks of Tasks (SAINT) uses a graphic network technique and is a computer simulation tool for modeling and analyzing large-scale man-machine systems but is applicable to a broad class of systems. SADT/SAINT provides the capability to map the system information from the SADT graphic notation to the SAINT graphic notation, and to execute the SAINT model on the computer. This paper describes the linking of SADT and SAINT resulting in an enhanced total simulation capability that integrates the analyst, user, and management.

SIMULATION PROCESS

The man-machine simulation process is presented here as a guide for the analysis of currently existing and conceptual large-scale systems but is applicable to a wide range of systems. The generic process is divided into six major activities (Fig. 1). The Describe System activity uses a system description technique to construct a validated static system model. The Generate Performance Data Base activity uses the static system

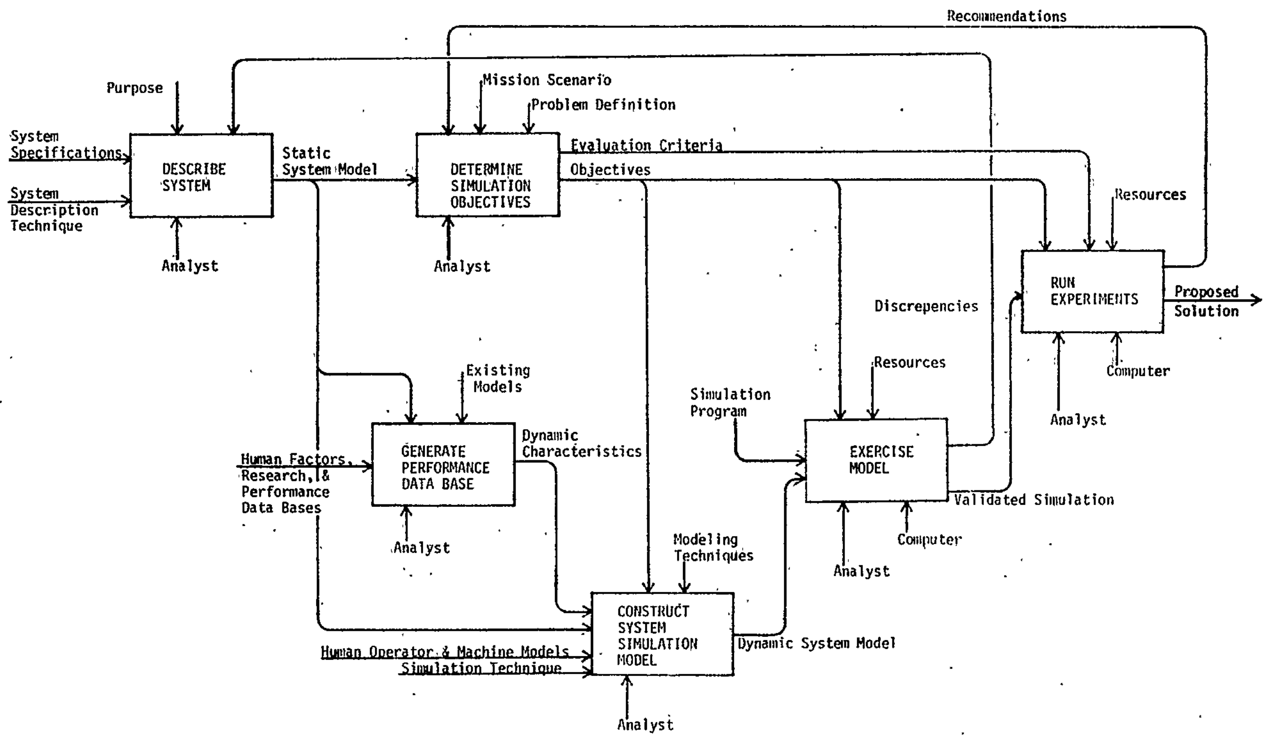


Figure 1. Man-Machine Simulation Process

model as a guide to select the dynamic characteristics of the system. The Determine Simulation Objectives activity determines what is to be simulated within the system and the criteria for evaluating the system. The Construct System Simulation Model activity develops a dynamic model of the system by using a simulation technique and by using the static system model as a guide. The static model defines the elements of the system and the characteristics of the elements, selects the dynamic characteristics from the performance data base, and combines them to form the dynamic model. This model defines the way in which the elements of the system interact to cause changes to the state of the system over time. The Exercise Model activity verifies and validates the dynamic model. Verification determines that the model executes on the computer as a modeler intended. Validation determines that the model is a reasonable representation of the system.

The Run Experiments activity exercises the model on the computer and interprets the results. The analysis of these results produces a proposed solution to the problem.

SAINT fits into the simulation process as the simulation technique used to construct the system dynamic model. SAINT is a computer simulation tool for modeling and analyzing large-scale, man-machine system and is potentially applicable to a broad class of problems. The SAINT program is the simulation program that executes on the computer and uses the dynamic model as input. The success of the simulation is largely dependent upon the accuracy to which the modeler performed the Describe System activity. If the modeler and the user worked closely together and understood the model and its outputs, then the project will likely be successful. However, if the model formation and assumptions are not effectively communicated, then the project will have limited value. Experience has shown that the graphic network notation of SAINT and other simulation models do not fully communicate a system description, system boundaries, and various levels of functional specifications. Thus, the Describe System activity is the weakest point in any simulation process because the dynamic models can be no better than the static functional model from which they are derived.

To perform this activity, a systematic, highly structured, top-down technique is needed for performing and planning requirements definition, functional analysis, and system design. A comparison of methodologies, existing or under development, indicates that only SADT has the combined attributes for being applicable to requirements study of large complex systems [1]. SADT is currently in use by the U.S. Air Force in the form of IDEF; it is thoroughly documented; it is easy to understand; and it provides the capability to model the resources in a man-machine system. Therefore, SADT was chosen as the system description technique used to produce the functional model by the Describe System activity.

The resultant technique, SADT/SAINT, provides the capability to translate from one graphic technique to another, from one model to another, and from a static system model to a dynamic model. The discussion presented here provides the reader with a brief explanation of how a SADT activity diagram communicates a system description, how a SAINT model is represented, and how to transition from SADT to SAINT.

SADT

SADT significantly increases the productivity and effectiveness of teams of people involved in a system project. Specifically, it provides methods for: 1) thinking in a structured way about large and complex problems; 2) working as a team with effective division and coordination of effort; 3) communicating analysis and design results in clear, precise notation; 4) documenting current results and decisions in a way that provides a complete audit of model design history; 5) controlling accuracy, completeness, and quality through frequent use of review and approval cycles; and 6) planning, managing, and assessing progress of the team effort [2].

An SADT model is a graphic representation of the system's hierarchic structure decomposed with a specific purpose in mind. A model is structured so that it gradually exposes more and more detail with the level of detail being dictated by the analysis requirements [3]. SADT uses a series of diagrams (figure 1 is an example) that define the system boundaries and illustrate the decomposition of the system structure in a top-down manner to the required level of detail. In order to provide a system description as a guide to creating a SAINT model, the system analyst develops an SADT activity (function) model.

An SADT activity model is an organized sequence of diagrams consisting of boxes (defining system activities) and of data arrows (defining relationships among the activities). The relationships, shown by the placement of the boxes and arrows, represent a description of the system for the reader.

The first diagram in a model is a single box that is a general description of the whole system. The General activity described by the diagram is further defined on the next diagram as three to six detailed activity boxes connected by arrows representing system data. This decomposition process continues until the system is described at the level of detail required.

The upper level or less detailed diagram defines boundaries for the lower level or more detailed diagram. The arrows around the box on the upper level diagram are the same as those entering and leaving the lower level diagram, i.e., the lower level is the same part of the same system in more detail (Fig. 2).

The relationship between the diagrams in a model is defined by using node numbers. The top diagram of a model is node A0. Each box on this diagram is numbered. The diagram representing the decomposition of box 3 of the A0 diagram would be called node A3. The decomposition of box 3 on node A3 would be node A33 (Fig. 2). Not all of the boxes of every diagram must be decomposed but all diagrams will be defined by a node number.

The activities performed by a system are represented by boxes on a diagram. An active verb phrase is written in each box to define the activity. The arrows that enter or exit a box represent the information or objects needed by or pro-

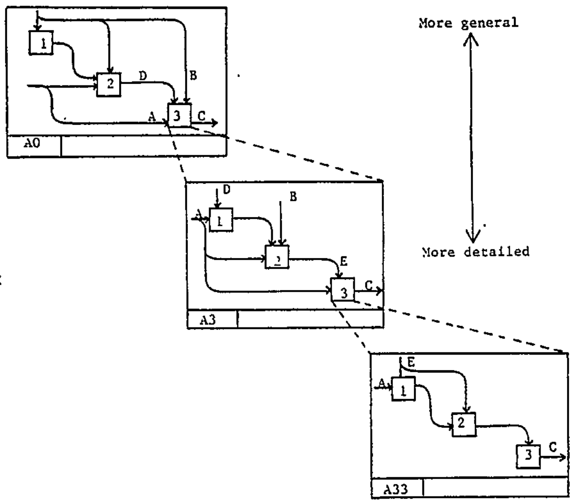


Figure 2. System Decomposition Using SADT Notation

duced by that activity. The side of the box the arrow enters determines its role. Arrows that enter the box are either input, control or mechanisms. Arrows that exit a box are outputs (Fig.3).

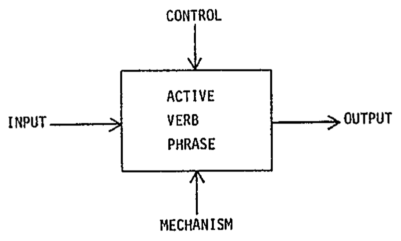


Figure 3. Relationship Between Activity Boxes and Data Arrows

Input arrows at the top and left of the box represent all the data that is needed for the box to fulfill its role. Input data entering on the left is transformed into output data on the right by the activity. A control entering the top of the box governs the way the transformation is done. A box with its inputs, controls and outputs represent WHAT the system does. Arrows entering the bottom of the box are called mechanisms and show HOW the activity is accomplished. Mechanisms represent the person, device, or process which carries out the activity and can be described by just a name on the arrow or by a reference to a separate SADT model.

The arrows connecting boxes on a diagram show a constraint relationship (Fig. 4). The box receiving the data is constrained since the activity cannot be performed until the data is available. Several activities can be performed at the same time on a diagram if all the constraints are satisfied. All or part of the data represented by the inputs may be required to satisfy the constraints. Sequence and time is not explicit in an activity diagram. Feedback represents update information to a previous activity.

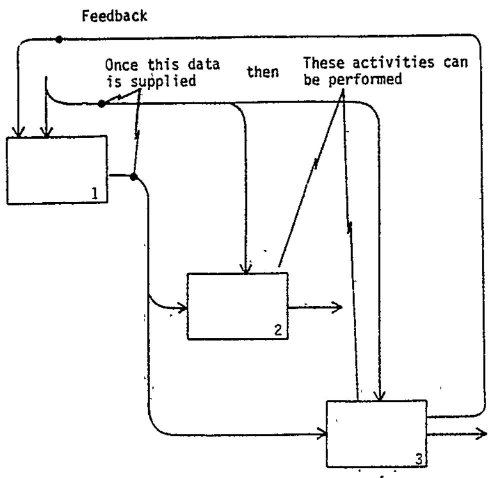


Figure 4. Simultaneous Activities & Feedback

Arrows represent categories of data. The higher the level of the diagram the more general these categories are. As the system is decomposed, each of the arrows is also decomposed into more detailed data category. Arrows may also join or aggregate similar data from different sources to form a general data category (Fig. 5).

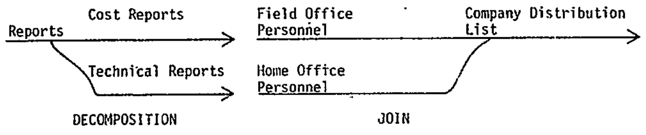


Figure 5. Arrow Decomposition and Join

The Author of SADT diagrams is responsible for studying a system and developing models based on SADT rules. He defines the orientation of the model by specifying the context, viewpoint, and purpose [3]. As the diagrams are developed, they are circulated for comments among individuals familiar with the system. The author updates and circulates the diagrams until they have been agreed upon and their required level of detail reached.

The SADT diagrams are easy to understand and can be used to communicate results to managers and technical personnel as the system is modeled. The final model report is published in an easy to read standard format that includes text and glossary of definitions. This report is a system description that is well coordinated, factual, and easy to communicate.

SAINT

SAINT is currently being used by organizations in industry, academia, and government to assist in the design and statistical analysis of complex man-machine systems. Models are synthesized using the SAINT symbol set and terminology resulting in a graphical network representation. These models are then exercised using the SAINT simulation computer program. SAINT, which has been alternately called a simulation technique [5], a tool[4], and a language [6], is potentially applicable to a broad class of systems in which discrete and/or continuous components and queues that exhibit time varying properties are portrayed. The history and evolution of SAINT may be found in [4].

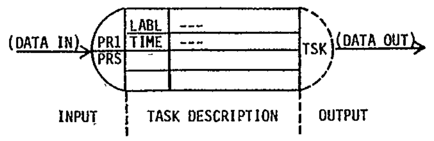
When SAINT graphical networks are synthesized, a subsystem of the system structure can be represented by a single network task or by a network of many tasks. The graphic features help communicate and describe the system by showing criticality or priority, timing, precedence relationship and sequencing probabilities. This is lacking in many modeling and simulation languages for system description.

Two extremely important factors in the SAINT modeling approach are the system description and the modeler's experience, knowledge, and understanding of the system. The modeler uses the system description and SAINT symbol set to synthesize a model. Therefore, the simulation results are only as good as the system description. Realizing that the system description is also a model, the validity of the simulation results depends upon the validity of both the descriptive model and the SAINT model and the equivalence of the two. The importance of the system description is exemplified by the following excerpt:

> The level of detail at which a system or system segment should be modeled cannot be specified a <u>priori</u>. It is the analyst's responsibility to determine the level of detail to be included in the network model based upon the nature of the problem he is trying to solve and an analysis of the task components and their interrelationships. He must decide if it is sufficient to model a task as a single unit, or if it necessary to model each component individually [5].

SAINT provides the framework for modeling systems in terms of discrete task elements, continuous system status variables or state variables, and the interactions between them. The fundamental elements of the discrete component are tasks, resources, and the concepts of data flow, events, conditions, intervals, system parameters, and system boundaries. Each task represents a function, process or activity that may be performed by resources (mechanisms). Branches connecting the tasks indicate precedence and successor relations for sequencing and looping among tasks.

The symbol used to model a task and branches are shown in Figure 6. The task symbol is divided into the input side, output side, and task description. The input side designates precedence requirements for the task release. The output side contains a task number and can be one of four possible shapes, each indicating a type of branching operation. The task description describes what is to occur during the performance of the task.



PRI	NUMBER OF PREDECESSOR COMPLETIONS REQUIRED FOR FIRST TASK RELEASE
PRS	NUMBER OF PREDECESSOR COMPLETIONS REQUIRED FOR SUCCEEDING TASK RELEASES
TSK	TASK NUMBER

Figure 6. Task Symbol

In processing a task, the SAINT simulation computer program releases, starts, and completes a task depending upon precedence, branching, resources, and other techniques available. The task description portion of the task symbol contains information associated with task performance, Statistics collection, attribute assignment, and branching. Each row contains a task description code and any necessary descriptive information [4]. Examples of task description codes are the following. LABL allows a description of each task. TIME signifies the task performance time and may be specified by a constant or by a sample from any one of eleven probability distributions. DMOD permits the substitution of one distribution set for another. MODF designates which of one or more functions accessible from a user-written subprogram a value is to be obtained. RESR specified the resources necessary for task starts. When criticality is important, PRTY can be used to designate initial priority which can then be modified dynamically. When more than one information packet arrives at a task, it is usually necessary to designate by INCH which one will be utilized and passed on to the next task. When different predecessor task completions are necessary for a task release, the DIFF code must be used. If it is not used, the repeated looping through any predecessor will cause the task to be released. PREC is used to designate ranking or order of task completions when tasks have identical completion times. Assignments to information, resource, and system attributes at the release, start, or completion of tasks can be made via ATAS. UTCH allows specification of additional modifiable task characteristics, e.g., environmental factors required for optimum performance.

The information within the model is contained in one of three types of attributes: information, system, and resource. Data flow along the branches is modeled by using groups of information attributes called information packets and is local data. These attributes can be assigned or modified at any task and are transmitted to successor tasks. System parameters and boundaries are global and do not flow through the network. A set of system attributes is used to designate these values and may be dynamically monitored and changed at any task. Each resource may be assigned a set of resource attributes containing characteristic information about the resource. This information is also global and may be dynamically monitored and changed at any task. All attributes may be used to dynamically determine task performance time, successor relations, or task priority. This provides control of data flow, changing events and conditions, contingencies, decision making, and control.

SAINT has the capability to connect the discrete task element and the continuous component. This interaction is achieved by programming the appropriate state equations in a FORTRAN subroutine or user function.

SADT/SAINT COMBINATION

By analyzing the capabilities of SADT and SAINT, it becomes apparent that each fulfills a specific function within the simulation process. SADT is a technique for building a functional model to design, understand, communicate, and document the system. SAINT is a detailed, graphical network technique for building a simulation model containing the necessary input information for the SAINT computer program.

Translating from SADT to SAINT involves the identification of details related to the chronology of the SADT activities and associating them with the SAINT nomenclature. This translation involves three major activities: 1) develop the static SAINT network, 2) generate the performance data base, and 3) add the dynamic characteristics to the SAINT network. The following paragraphs describe the process and ideas necessary to perform this translation.

Once the SADT functional model has been generated and validated, it is used as a pattern for generating the SAINT simulation model and as a guide for generating the performance data base. The SAINT model includes the discrete task-oriented model, the user written routines, and the continuous state variable equations.

The discrete task-oriented model is generated by translating the activities of the SADT model into the tasks of the SAINT flow network with each activity having a corresponding task. The SAINT tasks can represent different levels of detail. By proper selection of the SADT activities, it is possible to have a detailed model for a part of the system and a general model for the remainder of the system. For example, using Figure 7, the following multilevel model could be developed [3].

The initial SAINT model would be developed from activities 1, 2, 3, 4 on diagram AO. A more detailed model of activity 2 would be obtained by replacing activity 2 with activities 1, 2, 3 on diagram A2. In turn, a more detailed model of activity 2 on diagram A2 would be obtained by replacing it with activities 1, 2, 3, 4 on diagram A22. The diagram level and activity represented by the SAINT task can be described using SADT notation. For example, A.2 and A22.1 represent activity 2 on diagram AO and activity 1 on diagram A22 respectively. Using this notation, the resultant SAINT model would contain tasks corresponding to the following activities from three levels of detail: A1, A2.1, A22.1, A22.2, A22.3, A22.4, A2.3, A3, A4 [7].

This notation can be used as a reference from the SAINT tasks to the corresponding SADT activities by adding an additional SAINT task descriptor, called SADT, to the SAINT notation. The direct connection from SAINT to SADT is made by assigning the activity number to the task descriptor.

Each activity's title and mechanisms are translated to the task's LABL and RESR description codes respectively. The title and the mechanism names are translated directly from the SADT activity to the LABL and RESR description codes respectively of the SAINT task. As mentioned earlier, however, mechanisms represent the person, device, or process which carries out the activity and can be represented by just a name or by a reference to a separate model. The capability to represent a resource as a separate model allows a sensitivity analysis to be performed on the resources without making changes to the system.

The interactions among the SADT activities are the same that must exist among the SAINT tasks. However,

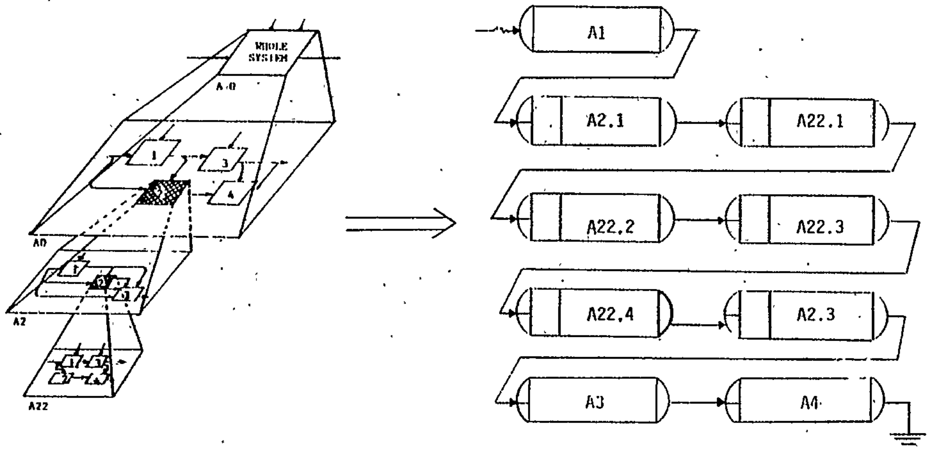


Figure 7. SADT-to-SAINT Model Transition

the SADT interactions do not show the conditional branches that data must sometimes take. Therefore, each activity must be considered and a decision made as to whether or not the data generated within that task will branch to subsequent activities based on a condition or whether it will flow for all conditions. When the latter condition exists, a deterministic branch appears on the task. An extra task may be inserted into the network to make the necessary checks and to control the data flow as necessary. Since these extra tasks are strictly for decision branching and not an actual system activity, the task time will be zero.

Within the SAINT flow network established, the task priorities for realistically controlling the data flow must be generated. Priorities are needed when tasks require the same resources and execute in parallel. Therefore, an analysis of the network is needed to determine when possible conflicts will arise and to determine what the priorities of those tasks must be.

The number of predecessors for each task must then be assigned. Using the previously specified data flow, a study of each task must be made in order to determine how many and which data inputs are necessary to cause task release.

The final step in generating the SAINT discrete task-oriented model is to assign task numbers. This involves assigning each task an integer number and can be done in the order specified by the analyst.

The second major activity is to generate the performance data base that contains the dynamic information required by the SAINT model. By using the SADT model as a guide, this dynamic information is available from human factors research, from existing data bases, from interview with system experts, and from simulations and analysis of submodels of the total system. This information will include such things as timing, resource attributes, and time-varying system attributes.

The third and final major activity involves the selection of the dynamic characteristics from the performance data base and adding them to the SAINT network. The timing required for each task is assigned from the data base. The data for the SADT inputs, controls, and mechanisms are selected from the data base and assigned to the SAINT information, system, and resource attributes.

In general, the SADT input data are placed in the information attributes and the control data are placed in the system attributes; but decisions must be made on an individual bases to make the most effective use of the SAINT capabilities. The resource attributes are assigned to describe each resource and are obtained from the mechanism description as given by the performance data base. These can be assigned constant values or values based on parameters such as time or workload. By using the SADT structure which allows mechanisms to be represented by a separate model, routines can be written, or existing operator models can be used, to model the resources in order to assign resource attribute values.

The user written routines implement the more sophisticated dynamic characteristics of SAINT by assigning information attributes and task descriptor codes based on changing parameters such as time.

The last activity is to generate the continuous state variable model which provides the capability to change values continuously over time. The user defines these state variables by writing the algebraic, difference, or differential equations that govern their time-dependent behavior. The use of state variables in SAINT is optional.

The SAINT discrete task-oriented model, the user written routines, and the continuous state variable model are now completed and are referred to as the simulation model. The information from the model is then transformed into the format acceptable by the SAINT program. The SAINT simulation model is exercised for verification and validation and then executed under experimental conditions. During these two activities, both the static and the dynamic models are continually referred to as a basis for system modification.

CONCLUSION

SADT/SAINT provides the capabilities necessary to build a system simulation in a top-down, structured manner, in a notation that communicates and documents the system, and in a form executable on a computer. The graphical notation of both techniques aids in the debugging, testing, modification, and communication of the system, and allows the various subfunctions of the system to be modeled at different levels of detail to meet the needs of the problem statement.

Both SADT and SAINT are thoroughly documented and are being applied to large, complex systems. The SADT/SAINT combination currently requires no modifications to the individual techniques. To date, only a portion of the transition capabilities have been explored. However, as SADT/SAINT is applied to more complicated systems, additional transition capabilities will be realized and incorporated, possibly requiring minor modifications to the individual techniques. The resulting translation capabilities will bring out even more emphatically the powerful modeling and simulation capabilities possible with the SADT/SAINT combination.

REFERENCES

Pritsker, A.A.B., Pegden, C.D. (1979). "Introduction to Simulation and SLAM". New York, John Wiley & Sons, Inc., and West Lafayette, Systems Publishing Corp.

Ross, D.T., "Structured Analysis (SA): A Language For Communicating Ideas:, IEEE Transactions on Software Engineering, Vol. SE-3, No. 1, January 1977.

Ross, D.T., Schoman, Kenneth E. Jr., "Structured Analysis For Requirements Definition", IEEE Transactions on Software Engineering, Vol. SE-3, No. 1, January 1977.

Seifert, D.J., Chubb, G.P. (1978), "SAINT: A Combined Simulation Language For Modeling Large Complex Systems". Aerospace Medical Research Laboratory, Wright-Patterson Air Force Base, Ohio, AMRL-TR-78-48.

SofTech, Inc., Waltham, MA, "An Introduction to SADT, Structured Analysis and Design Technique", Document 9022-78R, November 1976.

Townsend, Dwight F., "System Analysis: Key to The Future", Datamation, pp. 145-148, October, 1980.

Wortman, D.B., Seifert, D.F., Hann R.L., Chubb, G.P. (1978). "Simulation Using SAINT: A User-Oriented Instruction Manual". Aerospace Medical Research Laboratory, Wright-Patterson Air Force Base, Ohio AMRL-TR-61.