Proceedings of the 1988 Winter Simulation Conference
M. Abrams, P. Haigh, and J. Comfort (eds.)

Simulation modeling and analysis with INSIGHT: A tutorial

Stephen D. Roberts
Mary Ann Flanigan
SysTech, Inc.
P.O. Box 509203
Indianapolis, IN 46250, U.S.A.

ABSTRACT

The INSIGHT simulation language describes systems in a quick, simple, and compact fashion using a network representation. This description can be entered and simulated using novel interactive facilities that free the user from learning specific syntax. Statistics summarizing the simulation are produced automatically, but can be greatly enhanced by various input models and output analysis mechanisms. Use of the language does not require programming and complex models use the descriptive features of simple ones, incorporating more elaborate specifications and more sophisticated concepts. INSIGHT is available for most computers and is portable across machines. The language has been extensively applied and its scope of applications has ranged from manufacturing to service environments. Using INSIGHT, the process of simulation modeling and the results from the simulations combine to provide "insight" into problem solving.

1. INTRODUCTION

The INSIGHT (INS) simulation language is a high level, general purpose, discrete event simulation language that allows simulation models to be described quickly and compactly. Its fundamental concepts are easy-to-learn and easy-to-use. The language does not depend on any special competence with computer programming and yet the models can be easily extended though a rich variety of specifications. The emphasis on nonprocedural facilities and high level concepts makes INSIGHT a *simulation modeling language* rather than either a simulation programming language (where the user must program) or a special purpose simulation model (which is applicable to only a specific set of problems). Because INSIGHT is a simulation modeling language, simulation models can be built rapidly and the time consuming activities of debugging and remodeling are minimized, allowing users to focus on **problem-solving** rather than simulation mechanics.

As important as the simulation language is to describing systems, its use in problem-solving can be greatly enhanced by the *simulation environment*. On microcomputers running DOS/OS-2, like the IBM PC/PS and their compatibles, or on minicomputers running UNIX, such as the DEC VAX, the HP 9000, etc., INSIGHT is implemented in a fully **interactive** environment. Models are constructed and analyzed by *interacting* directly with the computer. On-Line help is immediately available and automated support exists for selecting input distributions, writing specifications, and diagnosing errors. Interactive simulation means that modelers obtain immediate feedback on the acceptability of their model and they can participate directly during the simulation to study both the dynamic and statistical behavior of the system.

1.1 Specific Features

INSIGHT differs from other simulation languages in several specific and important ways:

1. *Incorporating many general and unique modeling concepts which can be imitated in other languages only by resorting to programming.* These include reneging, free queues, algorithmic resource decision making, multiple and simultaneous resource requirements, activity abortion, multilevel preemption, early/late arrivals, process synchronization, arbitrary gather grouping and queue departure processing, queue capture, set identification and attribute inheritance, etc. INSIGHT is the only network-based language to offer full object-oriented facilities for both transactions and resources.

2. *Using a specification language that permits run-time expressions so that specifications can be arbitrary functions of the entities, the system status, and the statistics.* All information describing the simulation is directly available without programming calls to subroutines or procedures. Expressions are not limited to arithmetic, conditional, and logical operations, but can incorporate assignments, decisions, and iterations to generalize and extend the modeling concepts and features.

3. *Providing nonprocedural methods of statistics collection and display of simulation information.* In addition to automatically produced statistics, which can be modified, a broad range of other statistics and displays are available. For these, the modeler simply specifies what is needed and INSIGHT determines how to collect and display the results. Advanced statistical procedures for constructing confidence intervals and employing variance reduction are directly available without special routines or user-developed procedures.

4. *Possessing a wide variety of statistical input mechanisms for reflecting a broad range of input models.* INSIGHT has a full set of standard built-in distributions, and facilities for arbitrary discrete and continuous distributions, such as those obtained from data. Additionally, a time-varying Poisson process generator is available to model time-dependent processes and a multivariate Johnson system generator can be used for arbitrarily related multivariate input. The INSIGHT "help" facility aids in identifying and specifying appropriate input models.

5. *Being portable between mainframe, mini, and micro computers and running similarly in different environments and on different machines.* INSIGHT uses common random number and variate generators for all implementations and all versions are completely compatible, maintaining thirty-two bit accuracy. The micro and mini computer environment is fully interactive and can be used to construct models which can be uploaded when greater execution speed is needed.

6. *Is fully supported and extensively tested, by being used in practice and in classroom.* There is a textbook, *Simulation Modeling and Analysis with INSIGHT* by Stephen D. Roberts (1983), and a user's manual exists for the mainframe version (SysTech, Inc. 1985) and the interactive INSIGHT version (SysTech, Inc. 1988). These documents provide specific information on implementation details, error recovery, time and space use, and statistical features. The language and its environment are distributed and supported by SysTech, Inc.

1.2 Background

The INSIGHT language and its simulation environment have evolved from extensive actual experience over the past twelve years. For the past ten years it has been used at Purdue University in the senior course in Industrial Engineering in Systems Analysis and Design where students make extensive use of it in their projects (Roberts 1982). Much of the evolution of the language and its concepts and features have been motivated by an interest in an easy to learn and use simulation language which has general applicability and does not demand special programming or computer expertise. The primary emphasis has been the use of simulation in problem-solving.

Because the INSIGHT user deals with a direct interpretation of the system, attention is focused on modeling issues. INSIGHT models are visually appealing and easy to document. These models provide an excellent communication medium between the modeler and the client. Participation by the client in the modeling activity greatly enhances the credibility of the work and increases the chances that the findings will be implemented. INSIGHT has had routine application to a variety of problems involving production planning, scheduling and dispatching, staffing, bottleneck analysis, material handling, robotics, inventory control, facilities planning, resource balancing, cost analysis, and productivity improvement in a variety of industrial and service environments.

2. BASIC INSIGHT CONCEPTS AND FACILITIES

When modeling with INSIGHT (INS), the modeler (user) graphically conceives of the system to be simulated as a network of elemental processes. INSIGHT provides a set of **modeling symbols** for creating a representation of the system and a **vocabulary** for describing the system. Building the simulation model involves connecting modeling symbols, summarized in silhouette as Figure 1, into a network that corresponds to the system being studied.

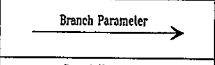
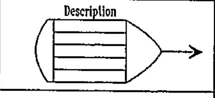
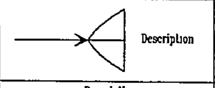
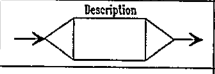
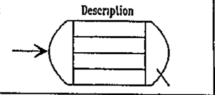
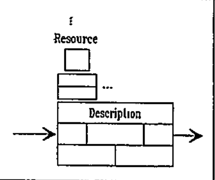
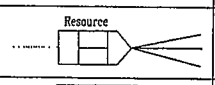
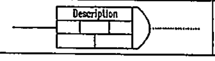
Node Type	Symbol	Processing Function
BRANCH	 Branch Parameter	Connects two nodes in a network according to their precedence and specifies how transactions branch
SOURCE	 Description	Creates transactions and schedules their arrival to the network
SINK	 Description	Removes transactions from the network
ASSIGNMENT	 Description	Assigns values to attributes
QUEUE	 Description	Delays transactions before an activity due to resource unavailability or a requirement to gate transactions
ACTIVITY	 Resource; Description	Performs an activity on transactions which may utilize resources
DECISION	 Resource	Represents the decision process used by a resource
DELAY	 Description	Represents a delay of the resource between activities

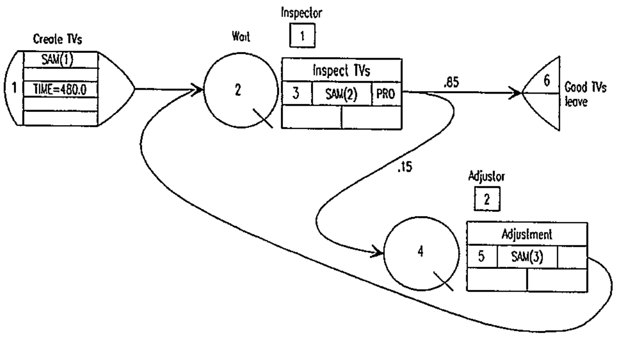
Figure 1: INSIGHT Modeling Symbols in Silhouette

The INSIGHT network is constructed about the flow of objects called *transactions*. A transaction is a general term that is interpreted by the modeler in the problem context. For example, transactions may represent TVs coming into an inspection station, people arriving for haircuts, ships entering a harbor, or parts entering a production facility. The nodes within the network are used to create transactions, assign attributes, cause queuing, perform activities, synchronize flow, and eventually remove transactions from the network. The branches route transactions from one node to another.

Transactions may require *resources* to process them at activities. The resource in INSIGHT is also a general term applied to an object that services transactions at one or more activities. Several resources may be required simultaneously at some activities. Resources may exercise independent decision making in fulfilling their service requirements throughout a network. They may be preempted by other more important service requirements or they may be unavailable for service by leaving the network from time to time. Resources may also be assigned attributes and experience delays, such as travel time, before or after performing services. Examples of resources are: inspectors who inspect TVs, barbers who cut hair, tugs which assist ships in a harbor, or machines and operators which manufacture parts.

2.1 A Simple Example: TV Inspection and Adjustment

As a portion of their production process, TV sets are sent to a final inspection station. Some TVs fail inspection and are sent to an adjusting station. After adjustment, the TVs are returned for re-inspection. The INSIGHT network corresponding to this system is shown in Figure 2.



RESOURCES		
Number	Description	Primary Service Node
1	Inspector	2
2	Adjustor	4

DISTRIBUTIONS	
Number	Description
1	Exponential interarrival time with mean 12
2	Erlang 3 inspection time with mean 10
3	Lognormal adjustment time with mean 70 and SD 5

Figure 2: Network Model of TV Inspection and Adjustment

The INSIGHT Network Model. Transactions will represent TVs since they are the units of traffic. Our resources will be an inspector who is needed at the inspection activity and an adjustor who is needed at the adjustment activity. All nodes have an identifying node number located on the left side of the node. The arrow pointing out of the node is called a *branch* and indicates where the TVs go next. The TVs enter the network at a *source node*. The source node controls when and how many TVs will arrive. At Source node 1, interarrival times are determined by SAMples from statistical distribution number 1 and TVs will be created until simulation TIME is 480. The SAMple specification is just one of the many *System-Defined Functions* (SDFs) available to help in writing specifications.

Inspection and adjustment of TVs are represented by *activity nodes*. Activities are places in the network where transactions usually receive service and are delayed in their journey through the network. Only one TV can be inspected or adjusted at a time because we have only one inspector and one adjustor. TVs that are forced to wait for service do so in *queue nodes*. Queue nodes are always adjacent to activity nodes so no branching is required from them. TVs wait at Queue 2 (shaped like a Q) until the inspector (identified as Resource number 2) can inspect them. Inspection time at Activity 3 is obtained by a SAMPLE from Distribution 2. Eighty-five percent of TVs departing Activity 3 are good and leave the network while 15% are routed to adjustment. Branching from inspection is denoted by the PRObabilistic branching method associated with the node. TVs which successfully pass inspection are no longer needed and leave the network at the *sink node*. Appended to the network are the glossaries identifying the resources available to the network and defining the set of statistical distribution references.

The INSIGHT network **visually** corresponds to an understanding of the real system. The symbols within the network not only convey individual processes but also contain relevant numerical data that control the processes. Very large or complex models can be created from such simple, basic processes by carefully assembling nodes and branches. The focus of modeling is confined to the construction of the network. Because the network has an intuitive appeal, it can be explained to decision makers in an effort to encourage their involvement in the modeling process.

Getting help. A very important dimension of the INSIGHT modeling environment is the on-line help. The *help* can be invoked directly within various INSIGHT facilities, such as asking about possible responses for network specifications within the Modeler, and the help program can be run in its own shell. Features of the help are:

* Computes distribution parameters from mean and standard deviation of observed data.
* Displays graphs of alternative distributions.
* Provides recommendations for error messages.
* Accesses pertinent information from the INSIGHT textbook.
* Issues detailed help on specifications.

For example, examining distributions is a part of the input modeling capabilities and one illustrative dialog is shown below for a Lognormal:

```
Enter the mean: 6

Enter the standard deviation: 3.0

 0.845|          *****
      |         *.....**
      |        *........*
 0.697|       *..........*
      |       *...........**                          LOGNORMAL
      |      *..............*                       (6.000,3.000)
      |     *................*
      |     *.................*
 0.486|    *...................**
      |    *.....................*
 f(x) |   *.......................**
      |   *.........................*
      |  *...........................**
 0.274|  *.............................**
      | *................................**
      | *..................................**
      |*.....................................****
      |*.........................................****
 0.063*..............................................****
      |..................................................
      +---------+---------+---------+---------+---------+
    2.000     4.600     7.200     9.800    12.400    15.000
```

Some additional help features being examined include the use of an expert system for the **diagnosis** of simulation errors; see Hill and Roberts (1987) for a prototype. Also a distribution fitting system is being installed for *input modeling* both with and without empirical data.

Using the INSIGHT Modeler. The INSIGHT Modeler facilitates model construction by conversing with the user during the model construction process. The user simply tells the Modeler what general elements are in the model, like an activity node, and the Modeler then queries the user about its characteristics, such as the activity time and resource requirements. Some of the prominent features of the INSIGHT Modeler include:

* Friendly interface -- no knowledge of syntax or programming required
* Prompts for needed information
* Suggests defaults for mosts specifications
* Provides appropriate help
* Input error detection and immediate notification
* Finds logical errors or incomplete model components by inspecting the entire model
* Produces an executable INSIGHT statement model
* Includes a full-featured expression editor

The following illustrates only a portion of an introductory session with the Modeler. User input occurs only after a ":" or a "?" prompt and is shown in full caps, although lower case is acceptable. The <CR> is used to denote a blank response, where there is only a carriage return.

```
Filename for Model? TV
[creating file tv.mod]

type in knowledge level (range 1-7) = 2

:SOURCE

        node number = 1? <CR>
        node name = SOURCE1? CREATE TVS
        interarrival time = 0.0? SAMPLE(1)
        time to begin creation = 0.0 <CR>
        termination criterion = NUM? ;HELP

        Alphanumeric beginning with:
          NUM: create the NUMber of blocks of transactions
               specified by termination value
          TIM: create blocks of transactions until the
               TIMe specified by termination value

        termination criterion = NUM? TIME
[checking interarrival time value ...]
        termination value = 0.0? 480
        block size = 1? ;SKIP
        number of branches = <unresolved>? 1
            branch sub-entity #1
                branch number = 1? <CR>
                destination node = <unresolved>? 2
[warning -- nonexistent node 2]
                branch condition = [true]? <CR>
                *where next = ;CO? <CR>
```

The Modeler operates by responding to commands or accepting responses to questions. At the *command* level, the user issues instructions to the Modeler. Most of these instructions designate elements of the model, such as a resource, queue, or activity. Some instructions are housekeeping or status commands which enable the user to manipulate the model and produce various files. At the *response* level, the user answers questions posed by the Modeler with respect to model elements. For instance, the Modeler will question the user about the node number or node name or some characteristic of the node. Generally, the questions provide a "default" answer, so you can enter a specific answer or accept the default answer by a carriage return. There are a set of special responses which allow the user to issue commands at the response level. For instance the ";help" obtains help with a question while the ";co" returns the user to command level.

As you answer questions about the model, the Modeler checks the acceptability of your response and will inform you if the response is inconsistent with the rest of the model or if the response is erroneous. You can change your response and alter the model immediately, without any more complex manipulations. The Modeler keeps track of all specifications and its advice is based not only on its general knowledge about INSIGHT, but also its growing knowledge about the model being built, typical of an *expert system*. The Modeler will even accept complex, multi-line expressions and edit them conveniently while checking them for inconsistencies or misuse.

A special advantage in using the Modeler is that users need only be knowledgeable about the INSIGHT modeling concepts. This approach is particularly valuable to modelers who do not want to be involved in the extensive detail, so often required of simulation users. As a special aide to learning and use, the Modeler has a "knowledge level" that stipulates the level of INSIGHT knowledge required, so that more advanced concepts can be hidden from routine use, and the Modeler won't request more advanced information from a user whose knowledge is less.

The model can be saved during any stage of its development and restored later to be completed. When the model is complete, the INSIGHT *statement model* can be created. The Modeler will perform this function directly.

The INSIGHT Statement Model. The statement model is the intermediate, computer readable form of the simulation model, that is portable across a wide variety of machines. From the statement model, INSIGHT automatically compiles and executes the simulation and provides output. The user is free of troublesome details such as event handling, statistics collection, and report writing. Furthermore, it is not necessary for the problem to be interpreted into some restricted simulation programming structure.

The Modeler will automatically generate the statement model from its interaction with the user, however the user can create and edit the statement model using a text editor. You can work with both the Modeler and a text editor by using a special program, called the Translator, which converts a statement model produced by a text editor into a form that is readable by the Modeler.

The statement model is submitted to the Compiler to construct the form of the executing simulation. INSIGHT uses a two-pass compiler to construct an efficient executing simulation. The compiler acts quickly. In Interactive INSIGHT, you can observe the interpretation of the statement model and optionally obtain an *echo* of the model. The echo of the model includes the values of all specifications, both those given by the user and those assumed by INSIGHT, and can be very useful in debugging and model verification.

Executing the Simulation. The simulation is menu-driven and executed interactively. Some of its prominent features are:

* View simulation as it executes
* Step through the simulation in one-event mode
* Can immediately halt and schedule future simulation interrupts
* Can tailor detailed trace elements
* Configure specific debug messages
* Save current simulation state
* Interfaces with Analyzer

The display of the current run and current time will appear as the simulation is executed. The simulation may be interrogated at any time during its execution by scheduling an **interrupt** or striking a carriage return. All these actions are directed within the following menu:

```
THE CURRENT RUN =   1   THE CURRENT TIME =      .00000

YOU MAY NOW:
    1. SIMULATE
    2. SIMULATE IN ONE EVENT MODE
    3. SCHEDULE THE NEXT INTERRUPT
    4. CHANGE THE FREQUENCY OF TIME UPDATES
    5. TRACE/DEBUG SYSTEM BEHAVIOR
    6. ENTER THE ANALYZER
    7. UNLOAD
    8. LOAD ANOTHER MODEL
    9. STOP
```

Having freedom to configure the trace allows you to view the behavior of the model at different levels of detail. Tracing model entities and attributes is especially helpful in **debugging.**

Any time during the simulation, the current state of the simulation may be *unloaded*. When a model is unloaded, its entire status, including statistics collected, is saved. This state may be reloaded at some later time and the simulation continued or the state of the system examined. This interaction can give a user new perspectives on the system being examined.

Analyzing the Simulation. By choosing the analyzer option from the main simulation menu, you can examine and possibly alter the simulation model. Within the analyzer, you can:

* Review or print standard reports
* Review the status of the entire model
* Examine all the transactions at a particular node
* View the status of resources
* Obtain specific statistics, including confidence intervals (where appropriate)
* Look at any specific transaction or resource and alter its characteristics
* Cause actions that alter the execution
* Edit the model without re-compilation

The analyzer provides interactive access to all information provided by INSIGHT including the SDFs, attributes, and statistics. For instance, if during the simulation you wanted a confidence interval on the number of transactions in Queue 4, you could invoke the following menu for node statistics:

```
THE FOLLOWING STATISTICS TYPES ARE AVAILABLE:
    1. NUMber of transactions in a specific queue
    2. time in a specific queue INCluding zero waits
    3. time in a specific queue EXCluding zero waits
    4. NUMber of transactions in a specific activity
    5. activity TIMe in a specific activity
    6. total time in the NETwork when leaving at a specific sink
    7. total time in QUEues when leaving at a specific sink
    8. total time in ACTivities when leaving at a specific sink node

CHOOSE ONE? 1

WHICH NODE NUMBER? 4

NUMBER OF TRANSACTIONS IN QUEUE     4
MEAN OF ALL RUN MEANS   =       .5843
STANDARD DEVIATION OF ALL RUN MEANS   =       .5152
SMALLEST OBSERVATION WITHIN ALL RUNS   =       .0000
LARGEST OBSERVATION WITHIN ALL RUNS   =     4.0000
OBSERVATION PERIOD (A TIME-WEIGHTED STATISTIC) =   6148.9010
STANDARD ERROR OF ALL RUN MEANS   =       .1717
MEAN OF CURRENT RUN   =       .0000
OBSERVATION PERIOD IN CURRENT RUN   =   126.5212
SUM (MEAN*OBSERVATION) OF CURRENT RUN   =       .0000

DO YOU WANT A CONFIDENCE INTERVAL FOR YOUR MEAN? y

THE FOLLOWING CONFIDENCE LEVELS ARE AVAILABLE:
     1. 99 PERCENT
     2. 95 PERCENT
     3. 90 PERCENT

CHOOSE ONE? 2

A 95% CONFIDENCE INTERVAL HAVING    8 d.f. = [  .1882,  .9803]
```

INSIGHT automatically collects a comprehensive set of statistics about the simulation. A variety of reports based on these or additional statistics can be generated and printed to the screen or to a file. The *Summary Report* presents the entire set of statistics that could be examined at any time. Without any specific instruction from the user, the Summary Report consists of three reports, representing three views of the system performance. The *Network Status Report* describes the current state of the network. The Network Status Report is very helpful in obtaining a general understanding of system activity. In addition, it informs us of the values of critical variables when the simulation terminates. The *Node Statistics* describe: the number of transactions and time spent in each queue (including and excluding zero times); the number of transactions and the time spent in each activity; and the time in the network, in queues, and in activities for the transactions which exit the network at each sink node. The Node Statistics reflects the experience of transactions as they flow through the network. The sink node statistics can be used as general productivity measures. *Resource Statistics* include utilization and availability information. These statistics inform us about the experience of the resources and their service.

2.2 INSIGHT Expressions

Much of the generalization in INSIGHT modeling is due to the powerful specification expressions. *Expressions* are combinations of primitive elements including constants, SDFs, attributes, and functions (which are themselves expressions) which are evaluated as the simulation executes. The SDFs represent system-defined elements, while the attributes and functions are user-defined. In addition to **arithmetic** (+, -, *, /, **), **relational** (.GT., .LT., .GE., .LE., .EQ., .NE.), and **logical** expressions (.AND., .OR., .NOT.), INSIGHT will accept **assignment**, **decision**, and **iteration** constructs. For example, the expression

```
(ACT = MAX(CUR(TIM),15-SAM(3)))
```

causes the attribute ACT to be assigned the MAXimum of the CURrent TIMe and 15 less a SAMple from Distribution 3. Furthermore, the value of the expression itself is the assigned value and can be used to specify, for instance, an activity time. A decision expression as

```
.IF.TYPE.EQ.1 .OR. NUM(QUE,4).GT.5
    .THEN. 5
    .ELSE. SIZE
```

has as its value either 5 or the value of SIZE depending on whether the condition, TYPE equal to 1 and NUMber in QUEue 4 greater than 5, is true or false. An iteration expression such as

```
.WHILE.(I=I+1).LT.NUMBER
        .DO.(TOT=TOT+TIM(BUSY,1))
```

adds to the present value of TOT the value of TIMe BUSy for resource I iterating from I to NUMBER.

2.3 Some Noteworthy Features

A few special features of INSIGHT should be mentioned. *Attributes* are an expressive feature in INSIGHT. They are named and assigned values anywhere within the model. Attributes apply to transactions and resources. Transaction attributes may refer to a **single** transaction (INDividual), to an entire **run** (RUN), the complete **simulation** (SIMulation), or to a **family** of transactions (SHAred). The SHAred transaction attributes have an inheritance property in that all members of the common "family" have access to these attributes while no transaction outside the family can access them. Such family properties are especially useful in containing common information like vendors or assembly numbers and can pass information among members for their use such as insuring that a welder that welded the cab also welds the frame. The resource attributes may apply to an individual RESource or to a TYPe of resource. Resource attributes are especially useful in allowing resources to remember breaks, locations, and past actions.

INSIGHT offers a wide variety of statistical *input distributions* to accommodate a broad range of random behaviors. A sample from a distribution is obtained from the SAMple SDF whose argument is a declared distribution. For example, SAM(SAM(3)) obtains a sample from a distribution that is identified by a sample from Distribution 3. You can declare distributions by requesting them within the Modeler. It will prompt you for the needed parameters and check their values. The standard statistical distributions available include most of the well-known discrete and continuous distributions including the Beta, Binomial, Erlang, Exponential, Gamma, Poisson, Lognormal, Normal, Weibull, Triangular, and Uniform. There are also the empirical discrete and continuous distributions that can accommodate an arbitrary mass function and a piecewise constant density function. Empirical distributions are typically those obtained directly from data. In contrast to these time-invariant distributions, a time-dependent distribution, called the time-varying Poisson, is available to provide a fundamentally different statistical process, most useful for specifying changing arrival rates that depend on time. The "help" facility, described earlier, will assist the user in examining the distributions available and perform several useful functions in selecting input models for the simulation.

A more flexible family of distributions is also available in INSIGHT called the **Johnson System.** This family can accommodate any range of distributions measured by their mean, variance, skewness, and kurtosis. Because of its flexibility, the Johnson System provides the broadest possible input models available in any simulation facility. Further, they have been extended to multi-variate models. The availability of this extension allows you to model dependent input variables. For example, the activity time may depend on the transaction's length, width, height, and weight. Previously, you would have to create some logical sequence of tests to evaluate such complex dependencies, but now you can model this input more naturally with a four-variate distribution. There is a special SDF option for sampling from multi-variate distributions that will fill an array of values so you can have access to the individual values immediately.

There are two special SDFs which allow the user to obtain or set information for any specific transaction or resource. These are referred to as the *Internal Transaction Pointer* (ITP) and the *Internal Resource Pointer* (IRP) which can be employed anytime an expression is evaluated. The ITP and IRP play an analogous role in modeling to pointer variables in modern programming languages.

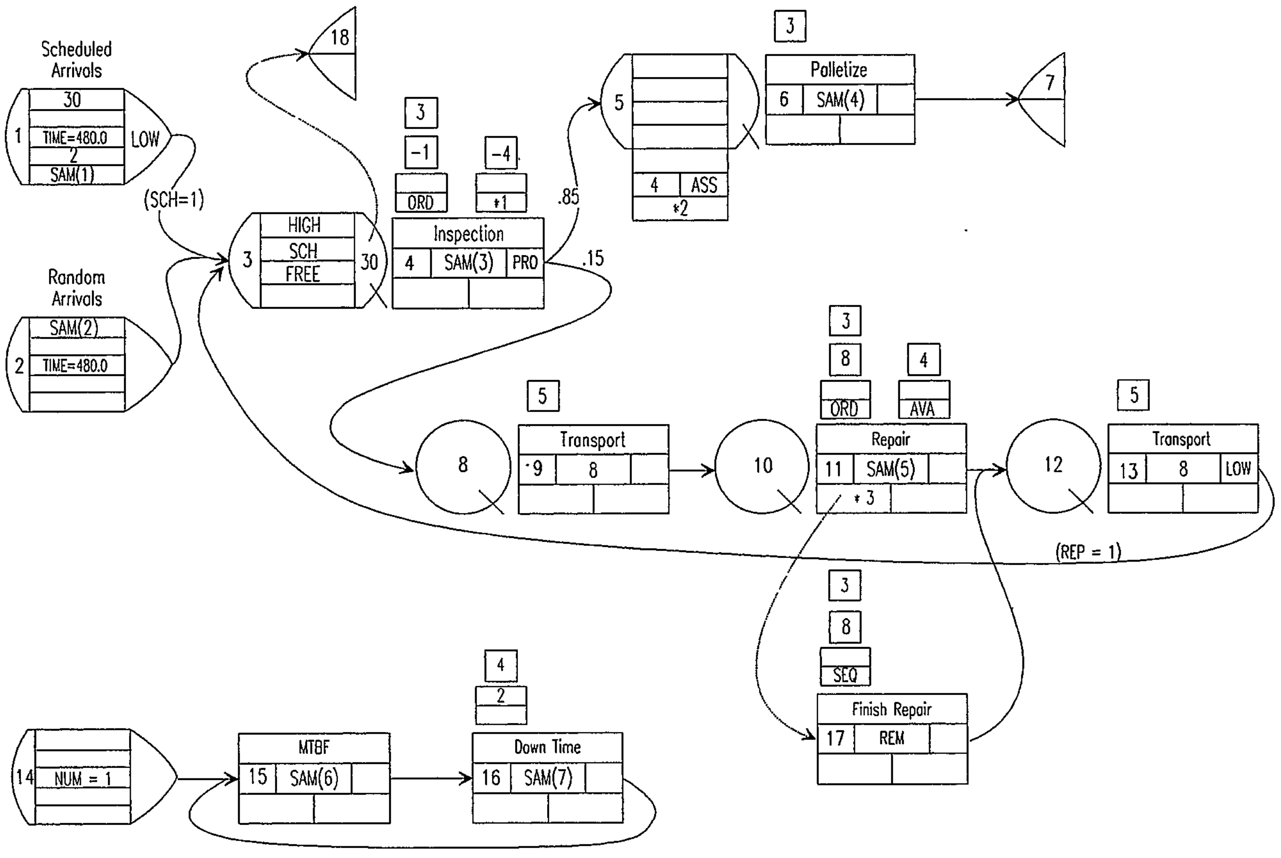
Other special SDFs exist within INSIGHT to control many other actions. For example, statistics may be cleared or started/stopped, special reports can be generated during the simulation, activity times may be changed, the simulation run can be stopped, resources can be made to arrive differently, etc. These functions can themselves satisfy specifications or become part of expressions which specify the model.

3. MODELING SOME PRACTICAL PROBLEMS

To illustrate the INSIGHT approach to modeling some common problems, we will embellish the TV Inspection and Repair problem described in Section 2. Each embellishment will be treated individually and we will highlight the approaches involved and comment on the related INSIGHT concepts. The complete model with all embellishments is shown in Figure 3, which should also serve as an intermediate reference. In this tutorial we can illustrate only a few INSIGHT concepts and features. You are referred to the textbook and other references if you desire more detail about the other language facilities.

3.1 Arrivals and Priorities

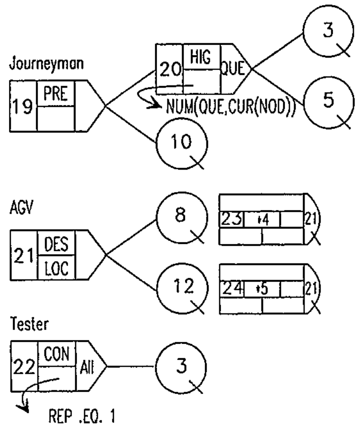
Problem. The TVs arrive in two broad patterns. Two TVs are regularly brought to the department by material handlers who are scheduled to arrive every 30 minutes. The actual arrival varies around the half hour schedule. The second pattern of arrivals is much less predictable and varies by the time of day. The scheduled TVs are given priority over the randomly arriving TVs.



RESOURCES			
Number	Description	Type	Primary Service Node
1.TO.2	Inspectors	-1	2
3	Journeyman		19
4	Adjustor		4
5	AGV		21
6.TO.7	Testers	-4	22
8	Repairman		4

DISTRIBUTIONS	
Number	Description
1	Normal with mean = -2 and SD = 4
2	Time Varying Poisson -- see graph
3	Erlang 3 with mean = 10
4	Gamma with shape = 25 and scale = .6
5	Lognormal with mean 70 and SD = 5
6	Exponential with mean 255
7	Johnson SB with xi= 0, lambda = 20, gamma = .647, and delta = 1.135

DECISION TREES



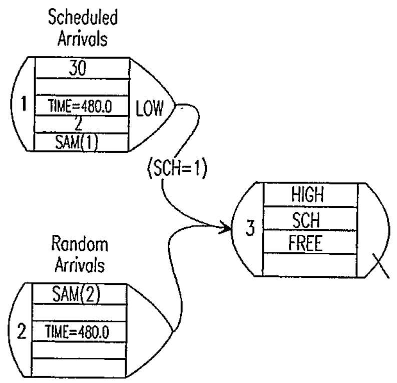
EXPRESSIONS

```
*1     .IF. REP .EQ. 1
          .THEN. -4
          .ELSE. 0
*2     SCH .EQ. ITP(CLA) + SCH
*3     17 + 0*(REM = TRA(REM) + 10)
*4     ( .IF. NUM(QUE,14) .EQ. 0
            .THEN. 6
            .ELSE. 0 )  + 0*(LOC=12)
*5     ( .IF. NUM(QUE,12) .EQ. 0
            .THEN. 6
            .ELSE. 0 ) +0*(LOC=14)
```

ATTRIBUTES			
Label	Description	Scope	Initial Value
SCH	Scheduled	IND	0
REP	Repair TV	IND	0
REM	Remaining Time	RUN	0
LOC	AGV Location	RES	12

Figure 3: Fully Embellished TV Inspection and Adjustment Problem

Approach. The modeling approach involves using two source nodes. Source 1 models the scheduled arrivals. The expected interarrival time is 30 minutes, but this time is modified by an early/late arrival time SAMpled from Distribution 1. The random arrivals are modeled with Source 2 with the interarrival time being SAMpled from a time-VARying Poisson process, defined in Distribution 2. An INDividual transaction attribute, labeled SCHeduled, is assigned on the branch from Source 1 to Queue 2 using an assignment expression. Subsequently, Queue 2 is ranked on the HIGhest value of SCHedule to give the scheduled TVs priority at the inspection.

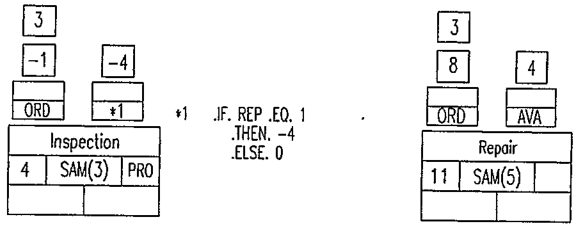


Observations. Notice that two entirely different processes are being modeled by the two source nodes. In Source 1 an arrival time about a schedule is employed, rather than an interarrival time, since the arrival time yields a more natural representation of the input process. In Source 2 the interarrival time is described by a time-dependent process rather than a time-independent model (such as an Exponential with constant parameters) and the approach involves a simple sampling, rather than the usual approach of dividing the day into a series of sources.

3.2 People and Machines -- Simultaneous Resource Usage

Problem. Multiple resources are required at the inspect and repair activities. Inspection needs an inspector and occasionally a tester, while the repair needs a repairman and an adjustor. We also note that there are two inspectors and categorize them as resource type -1. Further there is a journeyman who can also inspect and repair when needed. The tester is not needed for newly arriving TVs but is needed for TVs which have been returned from repair. There are two testers. At the repair, the repairman and the journeyman can satisfy the requirement for a person. There is, however, only one adjustor.

Approach. In INSIGHT, simultaneous resource requirements are modeled by simply constructing multiple stacks of symbols above the activity. Since both activities each require two resources simultaneously, there are two stacks above each. The alternative resources which can satisfy the requirement are identified by the symbols within a requirement stack. Where there is a choice of resources to satisfy a requirement, the selection is represented in the first symbol. For example, at the Inspection, the inspectors (resources of type -1) are preferred over Resource 3 (the journeyman) by the ORDer of the resource symbols. Where the preference among alternatives is not an issue, then other selection methods are available. Because the tester has only part-time use, its selection is given by a conditional expression that yields -4 (a tester) when the TV has been repaired (REP .EQ. 1).

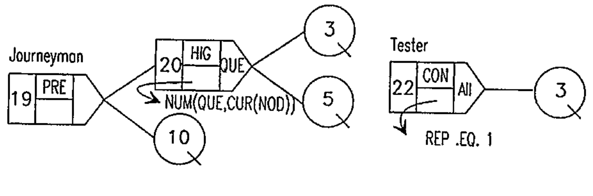


Observations. Simultaneous resource requirements are modeled directly, not serially as done in other simulation languages. Further, complex choices within resource requirements, which frequently arise in practice, are easily managed. The choice of resources can further influence the activity time, as can other network elements. For example, one inspector may become faster as the number of TVs waiting increase.

3.3 Resource Decision Making -- What to do next?

Problem. The journeyman works at three locations, inspection, repair, and palletizing. When he finishes an activity, he will first prefer either palletizing or inspection, choosing between the two based on the number of TVs backlogged in the queue. The tester's decision process must also be restricted to those TVs which have been repaired.

Approach. Resource decision making is modeled by decision trees. A decision tree represents the algorithms that resources use in deciding among the activities they must service (actually they serve transactions in the queues in front of the activities). Decision trees use decision nodes with queues referenced in the leaves to form decision processes which one or more resources may use. Each decision node contains a decision mode that describes how the associated nodes of the tree are to be evaluated. For example Decision 19 uses a PREferred mode, meaning look first at Decision 20 before examining Queue 10. Decision 20 contains additional specifications that cause Queues 3 and 5 to be evaluated, however that QUEue with the HIGhest Number will be examined first. Likewise the decision process for the Tester looks at ALL transactions in Queue 3, seeking one that satisfies the CONditional expression requiring that the TV be REPaired. Note that the REPair attribute is established following Activity 13. Also because certain TVs lower in the queue may be served first, due to the varying resource requirements, Queue 3 is a FREe queue. FREe queues permit any eligible transaction to be serviced, regardless of their position in the queue (however those who are first are first considered).

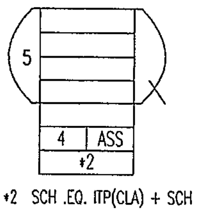


Observations. The decision tree gives special intelligence and identity to the resource and equips it with authority over its actions which parallel that of the transaction. In INSIGHT, transactions and resources form the two major classes in an object-oriented environment.

3.4 Gathering/Gating

Problem. TVs are palletized in groups of four. However scheduled TVs are palletized together as are the randomly arriving TVs.

Approach. To synchronize transactions, INSIGHT has facilities for gathering (bringing groups of transactions together) and gating (causing one transaction to wait until some condition is satisfied). Palletizing, like many operations that require an accumulation of transactions is easily modeled by modifying the queue node to gather four TVs which are ASSembled into one transaction (the pallet). To insure that the proper groups of TVs are formed together, a classifying condition is used which causes arriving transactions to gather into groups whose SCHedule attribute is equal.

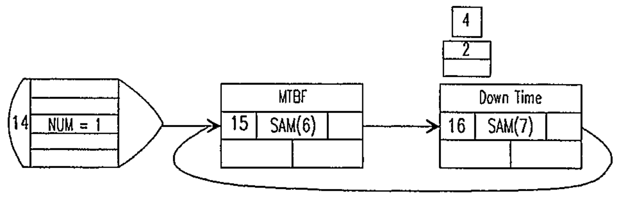


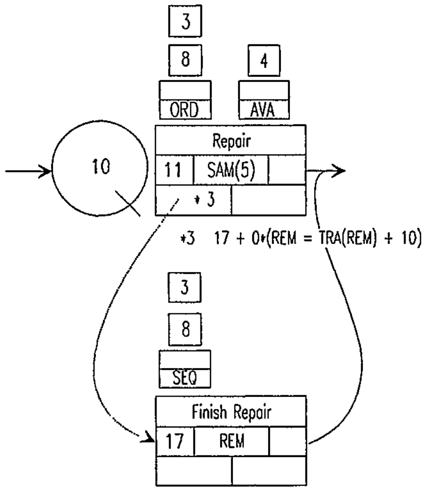
Observations. The gathering of transactions may involve many different operations and the departure specification can be expressed so that groups are BATched together, SERially processed, or NORmally handled. Also the number of transactions grouped together need not be restricted to a constant. It may be specified by a random sample or some kind of range involving a transaction attribute, such as making sure the weight exceeds some value but does not exceed another. The need for synchronizing one transaction occurs when different parts, like a truck body and its cab, are properly sequenced through a series of operations.

3.5 Breakdowns and Preemptions

Problem. We observe that the adjusting machine (Resource 4) breaks down and must be removed from service on a random basis. When this happens during a repair operation, the time to complete the activity must be extended because the work is done manually until the machine can be returned.

Approach. Breakdowns are modeled as a separate transaction, requiring immediate attention from the adjustor (Resource 4). The transaction entering Activity 16 can preempt the resource from Activity 11 because its preemption level is higher (at Activity 16 the preemption level is 2 and elsewhere it defaults to 1). But when Activity 11 loses its resource, the repair can either be suspended temporarily or the activity can abort. In this case choose the latter and abort the activity to be completed at Activity 17 which has only one resource requirement. The resource is taken SEQuentially from Activity 11 and the abortion expression computes the REMaining time as a RUN attribute, increasing the remaining time from Activity 11 for the TRAnsaction by 10 minutes. The result is used as the activity time in Activity 17.



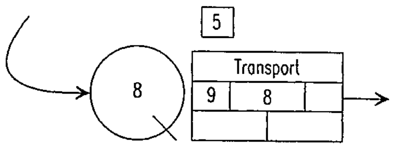


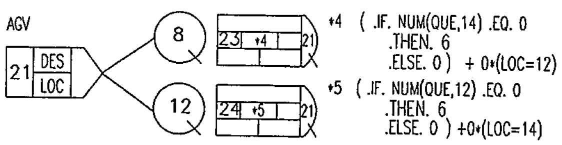
Observations. There is extensive potential for the INSIGHT concepts of preemption and abortion. Since preemption is given by an expression and the comparison of two expressions (one for the preemptor and one for the preemptee), there can be considerable flexibility. The preemption level not only causes preemptions for an activity but can prevent them, depending on the nature of the preemption level expression. Also, an activity which is preempted has the option of waiting for the return of the lost resource or it can be aborted to be completed in a different way.

3.6 Resource Delays and Materials Handling

Problem. An AGV (Automated Guided Vehicle) is positioned between inspection and repair. It moves between the two locations, transporting TVs as they are available. Transportation time is eight minutes, if the AGV is loaded and six if it is not carrying a TV.

Approach. By defining the AGV as Resource 5, it can simply serve the transport activities (Activities 9 and 13) as other resources. The activity times are eight minutes since it represents carrying a TV. However there may be nothing in the associated queues (Queues 8 and 12) when the AGV is available. Therefore the AGV continues to the next location. When a resource must be active, but not in service of a transaction, a delay node is used in the resource's decision tree. In this case the delay nodes (Delay 23 and 24) have their activity times specified by an expression conditioned on the availability of transactions in the associated queues. When there is none, then a delay of six minutes occurs until the next examination of the queues. The decision tree for the AGV uses a DESignated decision mode, so that each queue is examined in sequence.





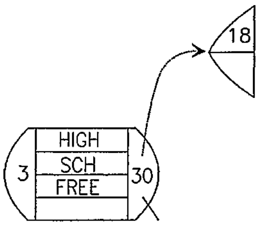
AGV
21
DES
LOC
8
23
*4
21
12
24
*5
21
4 (.IF. NUM(QUE,14) .EQ. 0 .THEN. 6 .ELSE. 0) + 0(LOC=12)
5 (.IF. NUM(QUE,12) .EQ. 0 .THEN. 6 .ELSE. 0) +0(LOC=14)

Observations. Material handling systems are modeled, rather than specified, in INSIGHT in order to provide flexibility in modeling a wide variety of systems. Elemental concepts are combined to produce more complex models so that when changes occur (such a consideration of a new material handling system), they can be modeled. Common material handling devices like conveyors, cranes, hoists, etc. are modeled using the same concepts.

3.7 Reneging

Problem. A complication in the inspect is that TVs which have been waiting for 30 minutes are frequently removed to another inspection operation. This phenomenon is called reneging, when a transaction establishes some upper bound on waiting.

Approach. Reneging can be modeled as a part of the queue specifications. The 30 minutes is the renege time and the reneging transaction is modeled a leaving at Sink 18.



Observations. Reneging gives transactions a means of departing a queue without being served at an activity. In INSIGHT, reneging can be conditional. The renege time is used simply to alert the transaction that it should consider reneging. At the renege time the transaction may choose to remain in the queue or leave. Conditional reneging provides a means of modeling jockeying, where a transaction moves from queue to queue without being served.

4. MORE SIMULATION FACILITIES

In addition to modeling concepts, INSIGHT has a number of facilities that enhance the simulation study. For example, there are extensive statistical features and I/O facilities. INSIGHT is extensible through programming.

4.1 Statistical Analysis

In addition to providing a high level approach to simulation modeling, INSIGHT also contains a number of built-in features to make accurate estimates of the variance of the sample mean and to perform variance reduction. The modeler can directly specify that INSIGHT estimate variances by replications (uniquely seeded runs of the same simulation model) or by batches (division of observations into collections of observations). INSIGHT does not automatically provide an estimate of the standard error from observations known to lack independence. This is why the output from the TV Inspect and Repair problem would use one observation per run to compute variances. Similar computations could be obtained using batches when dealing with a steady-state simulation. Tables can be used for more complex statistics collection.

Tables are a unique feature of INSIGHT in that they employ a nonprocedural format. The desired statistics are described to the Modeler and INSIGHT automatically handles the tasks of collecting, compiling, and displaying statistics. For example if within the Modeler we requested a table as:

```
:TABLE
        table number = 1? <CR>
        table name = TABLE1? NUMBER IN INSPECT QUEUE
        statistic type = <unresolved>? NUM
        number of referenced entities = <unresolved>? 1
            queue/activity node number #1
                node number = <unresolved>? 2
                *where next = ;CO?
        number of histogram cells = 0? 10
        lower limit of first cell = 0.0? 1
        cell width = VAR? 1
        collection control = COM? <CR>
        table breakdown = NO? <CR>
```

which produces a table and histogram.

Start-up issues requiring special initial conditions can be specified directly within INSIGHT by the PRERUN that initializes variables and attributes and inserts transactions in nodes. Truncating data during start-up is accomplished by System-Defined Functions that can clear specified statistics. Clearing can occur within the network or be activated at a specific time. Furthermore, run length or batch size can also be controlled within the network model by employing expressions which can terminate a run or batch interval. Statistics collection can also be stopped and started.

Variance reduction techniques employing common, antithetic, and paired random variates are available in INSIGHT by direct specification. By default each distribution in the model has its own separate random number stream. There are over 2100 individual streams in INSIGHT. Thus different experiments with a model automatically use common sources of variation. Within an experiment, some runs may have common streams and other runs may be made antithetic automatically. To maximize the applicability of these variance reduction procedures, INSIGHT employs inverse transform variate generators so that random numbers are inherently synchronized. By including these features in INSIGHT, statistical analysis can become an integral part of simulation modeling.

Special attention has been given to random number and random variate generators in INSIGHT. The random number generator (Marse and Roberts 1983) is completely portable to any machine having 32 bit or greater word length while retaining excellent statistical properties. The implementation of the generator not only permits the automatic use of common variates but in combination with the variate generators causes INSIGHT simulation models to run identically on different computers. This makes simulation results portable. The variate generation process using inverse transforms is extensive, ranging from standard distributions to a time-varying Poisson process (Klein and Roberts 1984) and a multi-variate Johnson system generator that permits up to 42 dependent variates. More discussion of statistical issues in INSIGHT is found in Roberts and Klein (1984).

4.2 File I/O

INSIGHT has the capability to read data from a file (or the terminal), and to write data to a file (or the terminal). The input/output feature is obtained through the **FORMAT** statement, and eliminates most of the need for user-written code. It allows two-way communications of information between you or a file and INSIGHT, and has a variety of useful purposes including:

* Run data-driven simulations
* Write data to a file, which could be used in other software packages
* Interact directly with the user
* Debug and verify models
* Access and select information from data files

Some examples of the FORMAT statement are:

```
FORMAT, 1, , 'ARRIVAL TIME IS'F6.2, ARR
FORMAT, 2, INPUT, F8.3, ARR, F8.3, SER
FORMAT, 3, STOREDATA, F8.3, -1, F8.3, CUR(RUN), E15.5,
                IF. CUR(TIM) .LT. 200
                   .THEN. MEA(UTI,-1)*100
                   .ELSE. MEA(UTI,-2)*100
FORMAT, 5, REPORT,//'AT TIME = 'F10.3, CUR(TIM),
     'THE TIME IN SYSTEM IN MINUTES FOR',,
    /'   TVs with NO rejects is'X3F10.3, CME(NET,7),
    /'   TVs with ONE reject is'X3F10.3, CME(NET,8),
    /'   TVs with TWO rejects is'X2F10.3, CME(NET,9),
    /'   TVs with more than two rejects is 'F10.3, CME(NET,10)
    /'OVERALL PRODUCTION RATE IS 'F10.5,
     (COU(8)+COU(9)+COU(10))/CUR(TIM),' TVs Per Minute'
```

Format statement 1 could be used to read the arrival time from the terminal (a blank filename designates the terminal), and/or it could be used to write the arrival time to the terminal. Format statement 2 is used to read input from a file called INPUT and will supply the simulation with ARRivals and SERvices. Format statement 3 writes three values to the file STOREDATA. Notice the use of INSIGHT expressions. Format 5 writes a special report to the report file. A "/" produces a carriage return and improves the appearance of the output.

To read from, or write according to a specific format statement, the REAd and WRIte SDFs are used. REAd(2) will read the values associated with Format number 2, and WRIte(5) will write the fragment/values associated with Format 5. Another SDF associated with the format statement is the FILe SDF, which can be used to determine or to move the file pointer.

The use of files allows the data that drives the simulation to be stored separately from the model. It is an especially good way to initialize large arrays and to format special reports.

4.3 Programming Interface

To allow conversation with other software, an interface is available which allows communication of information between INSIGHT and a program. Complex or frequently used procedures may be written in FORTRAN, C, or other linkable languages to reduce execution time. SDFs and attributes may be used in the routines to provide current status information, or cause simulation actions, or obtain statistical information which can be used to print reports tailored to the model. Facilities exist for user-determined events and statistics collection. Finally, a program can be written to execute a simulation many times while testing various model parameters to optimize an objective function.

5. CONCLUSIONS

INSIGHT provides an easy-to-use simulation capability that contains powerful modeling concepts that do not rely on general programming. Such an approach makes simulation modeling available to those with little prior experience and further extends the scope of possible simulation applications. Built-in procedures for statistics collection and automatic output generation mean that results are obtained easily and quickly.

INSIGHT is fully interactive and the modeling and simulation requirements are further reduced since the system can now provide so much help in modeling and simulation. Also since the simulation is interactive, the user can obtain both static and dynamic information about the simulation. INSIGHT runs under a variety of operating systems on mainframes, minis, and micros with results being fully portable. The language is supported by SysTech, Inc. at P.O. Box 509203, Indianapolis, IN 46250, (317) 842-6586, which also is responsible for the distribution and maintenance of the simulation products.

REFERENCES

Hill, T. R. and Roberts, S. D. (1987). A prototype knowledge-based simulation support system. *Simulation* **48**, 152-161.

INSIGHT User's Manual (1985). SysTech, Inc., Indianapolis, Indiana.

Interactive INSIGHT Simulation System (1988). SysTech, Inc., Indianapolis, Indiana.

Klein, R. W. and Roberts, S. D. (1984). A time-varying Poisson process generator. *Simulation* **43**, 193-195.

Marse, K. J. and Roberts, S. D. (1983). Implementing a portable FORTRAN Uniform (0,1) generator. *Simulation* **41**, 135-139.

Roberts, S. D. (1982). Teaching simulation to undergraduates. In: *Proceedings of the 1982 Winter Simulation Conference* (H. Highland, Y. Chao, O. Madrigal eds.). The Institute of Electrical and Electronics Engineers, San Diego, California, 706-707.

Roberts, S. D. (1983). *Simulation Modeling and Analysis with INSIGHT*. Regenstrief Institute, Indianapolis, Indiana. Distributed by SysTech, Inc.

Roberts, S. D. and Klein, R. W. (1984). Statistics collection display, and analysis in INSIGHT. Regenstrief Institute, Indianapolis, Indiana.

AUTHOR'S BIOGRAPHY

STEPHEN D. ROBERTS is Professor of Industrial Engineering at Purdue University and Professor of Internal Medicine at the Indiana University School of Medicine. His academic and teaching responsibilities are in simulation modeling. His methodological research is in simulation language design and includes INSIGHT (INS) a general purpose, discrete event language, and SLN for the Simulation of Logical Networks. He is also a principal in SysTech, Inc. which distributes the simulation languages and consults on their application.

He received his BSIE, MSIE, and PhD in Industrial Engineering from Purdue University and has held research and faculty positions at the University of Florida. He is active in several professional societies and in addition to making presentations and chairing sessions at conferences, he was Proceedings Editor for WSC '83, Associate Program Chairman for WSC '85, and Program Chairman for WSC '86. Presently he is Chair of SIGSIM, an Area Editor for Simulation, and WSC Board Representative for TIMS.

MARY ANN FLANIGAN is a Vice-President with SysTech, Inc. responsible for marketing and distribution of simulation products. She received her BSIE and MSIE in Industrial Engineering from Purdue University. She is also a Research Assistant with the Regenstrief Institute and an Instructor in Industrial Engineering at Purdue. Her interests are in simulation modeling and analysis, and in the development of new simulation software.

Stephen D. Roberts
Mary Ann Flanigan
SysTech, Inc.
P.O. Box 509203
Indianapolis, IN 46250, U.S.A.
(317) 842-6586