

SIMULATION WITH INSIGHT

Stephen D. Roberts, Ph.D.
Regenstrief Institute for Health Care
1001 West Tenth Street
Indianapolis, IN 46202

ABSTRACT

INSIGHT (INS) is a computer simulation language for describing systems in a quick, simple, and compact fashion. Their description is executed by a computer and statistics summarizing the simulation are automatically provided. Use of the language does not require any special programming or statistical expertise. Complex models use the descriptive features of simple ones but incorporate more elaborate specifications. Likewise sophisticated statistical and simulation procedures available in INSIGHT are additions rather than revisions of the model. Simulations can be executed on mainframes when execution speed is needed or on a micro where interaction with the simulation and model development is facilitated. The net result is that the process of simulation modeling and the results from the simulations combine to provide "insight" into problem solving.

INTRODUCTION

Simulation languages can be generally classified into two categories, general simulation programming languages and special parameterized simulation models. The simulation programming languages are rooted in general purpose programming and require the modeler to code simulation procedures using some version of a general purpose language like Fortran, Algol, or Pascal. Although highly flexible, considerable programming and simulation knowledge is necessary to construct even simple models. More recent simulation languages attempt to incorporate higher level built-in procedures with network or process facilities but they fundamentally rely on general purpose programming procedures written by the modeler for extensive models. Parameterized simulation models such as those for assembly line balancing or flexible manufacturing or warehousing offer greater ease of use and usually require no programming. However, only highly specific systems can be studied, with little possibility of generalization.

INSIGHT (INS) has been designed to be a high level, general purpose, discrete event simulation language. It does not depend on any special competence with general purpose programming and modelers need not code any procedures. Its few fundamental concepts are easy-to-learn and easy-to-use, making simulation modeling accessible to a greater number of persons. There is little sacrifice of generality because of the rich variety of INSIGHT specifications which extends the modeling concepts. The emphasis on nonprocedural facilities and high level concepts makes INSIGHT a simulation modeling language rather than either a simulation programming language or a parameterized simulation model. Using a modeling language,

simulation models can be built with much less time than previously possible and the time consuming activities of debugging and remodeling are minimized.

INSIGHT differs from current simulation languages in several important ways by:

- *incorporating many general modeling concepts that can be initiated in other languages only by resorting to programming. Such concepts include renegeing, free queues, algorithmic resource decision making, multiple and simultaneous resource requirements, activity abortion, preempt-resume activities, early/late arrival processes, alternative attribute scope, array attributes, process synchronization, general gather grouping and queue departure processing, queue capture, set identification, etc. These general concepts subsume the network and process features of existing languages.

- *using a specification language that permits extensive run-time evaluation so that specifications can be state-dependent and general. All the information known about the simulation is available directly to the modeler without calls to subroutines or procedures. This information can be combined with any model-specific attributes in very flexible expressions to generalize the behavior of the modeling concepts.

- *providing nonprocedural methods of statistics collection and display of simulation information. The modeler only specifies what information is needed and INS 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 procedures.

- *being portable between mainframe, mini, and micro computers and being capable of not only executing in a variety of environments but also running similarly using common random number and random variate generators. Models can be constructed and executed interactively on a desktop PC and then uploaded to a mainframe or mini when execution demands the efficiency. Furthermore, all versions of INSIGHT are completely compatible and maintain 32 bit accuracy.

- *being fully supported and extensively (classroom) tested. There is both a textbook (1) describing the entire language including numerous examples and a user's manual (2) which provides specific information on implementation details, error recovery, time and space use, and statistical features.

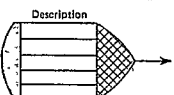
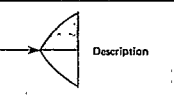
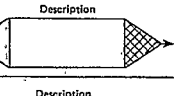
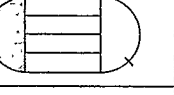
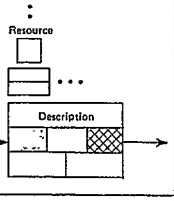
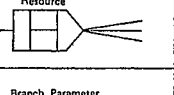
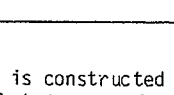
Because the modeler deals with direct interpretation of the system, attention is focused on modeling issues. INSIGHT models are concise, easy to document, and visually appealing. They make an excellent communication medium between the modeler and the client. Participation by the client in the modeling

activity greatly enhances the credibility of the work and increases the probability of eventual implementation. 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.

BASIC INS CONCEPTS

Modeling with INSIGHT (INS), the modeler graphically conceives of the system to be simulated as a network of elemental processes. INS 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.

Figure 1: INSIGHT nodes in Silhouette

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

The INS network is constructed about the flow of units of traffic 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 customers coming into a gas station. The pages 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 INS is also a general term applied to an entity 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. Examples of resources are inspectors who inspect TVs, barbers who cut hair, tugs which assist ships in a harbor, and gas station attendants who service customers.

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 reinspection. The simple INS network needed is shown in Figure 2.

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 adjuster 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 a statistical distribution numbered 1 and TVs will be created until simulation TIME is 480. The SAMPLE specification is just one of many System-Defined Functions (SDFs) available for the modeler to write 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 adjuster. TVs that are forced to wait for service do so in queue nodes. Queue nodes are always adjacent to activity nodes so no branching from them is required. TVs wait at Queue 2 (shaped like a Q) until the inspector (identified as resource 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. TVs which successfully pass inspection are no longer needed and leave the network at the sink node. Appended to the network are glossaries identifying the resources available to the network and defining the set of statistical distribution references.

The network corresponds visually to our 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 clients or decision makers in an effort to encourage

Figure 2: TV Inspection and Adjustment Network

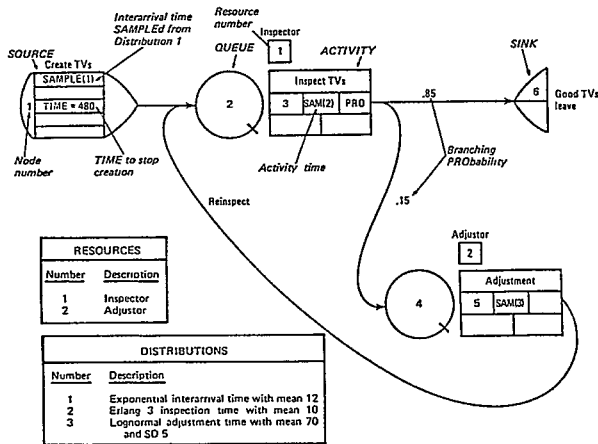


Figure 3: INS Statements for TV Inspection and Adjustment

```

SIMULATION OF = TV INSPECTION AND ADJUSTMENT,, RUNS = 10
$
RESOURCE = 1 INSPECTOR,, 2
RESOURCE = 2 ADJUSTOR,, 4
$
DISTRIBUTION = 1, EXPONENTIAL INTERARRIVAL TIME WITH MEAN = 12 MINUTES
DISTRIBUTION = 2, ERLANG INSPECT TIME WITH MEAN = 10 MINS, 3RD ORDER
DISTRIBUTION = 3, LOGNORMAL ADJUST TIME WITH MEAN = 70 MINUTES, 5 SD
$
SOURCE = 1 CREATE TVS,, SAMPLE(1),, TIME TO STOP CREATING TVS = 480
BRANCH TO, 2
$
QUEUE = 2 INSPECT WAIT
ACTIVITY = 3 INSPECT TVS, PROBABILITY BRANCH AND ACT TIME = SAMPLE(2)
SELECT,, 1 THE INSPECTOR
BRANCH TO, 6 WITH PROBABILITY = .85
BRANCH TO, 4 WITH PROBABILITY = .15
$
QUEUE = 4 ADJUST WAIT
ACTIVITY = 5 ADJUST TVS,, SAMPLE(3)
SELECT,, 2 THE ADJUSTOR
BRANCH TO, 2 FOR REINSPECTION
$
SINK = 6 GOOD TVS LEAVE
FINISH
    
```

their involvement in the modeling process.

The network must be translated into a form suitable for the computer. Nodes and branches along with the glossaries are expressed by INS statements called the statement model, given in Figure 3. INS statements contain fields separated by commas or equal signs. A dollar sign (\$) precedes comments on a line. Comments are permitted within a field after the field's information has been specified. The use of comments throughout not only documents and clarifies the statements but also makes the input readable.

The SIMULATION statement introduces the simulation problem and supplies identifying information. The glossaries define three statistical DISTRIBUTIONS and two RESOURCES. INS has fourteen built-in statistical distributions including those constructed from user-supplied data and time-dependent processes. Resources

and nodes may be given names in the second field. The node and branch statements are written as they appear in the network with the BRANCH statement following its node. The third field of the SOURCE and ACTIVITY statements is used to control the method of branching. TVs branch PROBABILISTICALLY from Activity 3; at other nodes, TVs go directly to the indicated node. The SELECT statement tells INS which resource services transactions at the activity. Multiple commas in a statement are used to skip unused fields for which INS automatically provides defaults.

From the statement model, INS automatically compiles and executes the simulation and provides output. The modeler 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 a next event simulation or some other process structure.

Figure 4: Tracing the Behavior of the TV Model

TIME	ENTITY	NUMBER	ACTION	NODE	TYPE	NUMBER	RELATED INFORMATION
0.000	INSPECTOR	1	HAS ARRIVED IS IDLE				
0.000	ADJUSTOR	2	HAS ARRIVED IS IDLE				
0.000	TRANSACTION	1	CREATED AT	CREATE TVS		1	
			ENTERING	INSPECT WAIT		2	NUM(QUE,*) = 0
			DEPARTING	INSPECT WAIT		2	TIME IN QUEUE = 0.000
	INSPECTOR	1	BUSY AT	INSPECT TVS		3	ON TRANSACTION 1
	TRANSACTION	1	BEGINNING AT	INSPECT TVS		3	
10.242	TRANSACTION	1	COMPLETED AT	INSPECT TVS		3	ACTIVITY TIME = 10.242
	INSPECTOR	1	IN BUFFER STORAGE				
	TRANSACTION	1	DESTROYED AT	GOOD TVS LEA		6	CUR(RTC) = 0
	INSPECTOR	1	IS IDLE				
20.008	TRANSACTION	2	CREATED AT	CREATE TVS		1	
			ENTERING	INSPECT WAIT		2	NUM(QUE,*) = 0
			DEPARTING	INSPECT WAIT		2	TIME IN QUEUE = 0.000
	INSPECTOR	1	BUSY AT	INSPECT TVS		3	ON TRANSACTION 2
	TRANSACTION	2	BEGINNING AT	INSPECT TVS		3	
28.067	TRANSACTION	2	COMPLETED AT	INSPECT TVS		3	ACTIVITY TIME = 8.079
	INSPECTOR	1	IN BUFFER STORAGE				
	TRANSACTION	2	DESTROYED AT	GOOD TVS LEA		6	CUR(RTC) = 0
	INSPECTOR	1	IS IDLE				
36.199	TRANSACTION	3	CREATED AT	CREATE TVS		1	
			ENTERING	INSPECT WAIT		2	NUM(QUE,*) = 0
			DEPARTING	INSPECT WAIT		2	TIME IN QUEUE = 0.000
	INSPECTOR	1	BUSY AT	INSPECT TVS		3	ON TRANSACTION 3
	TRANSACTION	3	BEGINNING AT	INSPECT TVS		3	

Figure 5: Summary Report for the TV Model

```

*****
NETWORK STATUS
*****
TIME = CUR(TIM) = 680.247      NODE STATISTICS LAST STARTED AT TIME = 0.000
RUN = CUR(RUN) = 10          RESOURCE STATISTICS LAST STARTED AT TIME = 0.000
RTC = CUR(RTC) = 0          NODE STATISTICS COLLECTION TIME SINCE LAST CLEAR = 680.247
NODE = CUR(NOD) = 0        RESOURCE STATISTICS COLLECTION TIME SINCE LAST CLEAR = 680.247
COMPLETED ACTIVITY = CUR(ACT) = 0
TRANSACTION = CUR(TRA) = 0  TRANSACTIONS IN NETWORK = NUM(NET) = 0
RESOURCE = CUR(RES) = 0    TRANSACTIONS CREATED = NUM(CRE) = 40
                           TRANSACTIONS DERIVED = NUM(DER) = 0

MAXIMUM SPACE UTILIZED = 725/ 5000 OR 14.50 PERCENT
    
```

	NODE NAME	NODE NUMBER	NODE COUNT COU(*)	NUMBER IN NODE	ZERO WAIT ZER(*)
QUEUES	INSPECT WAIT	2	449	0	122
	ADJUST WAIT	4	67	0	23
ACTIVITIES	INSPECT TVS	3	449	0	
	ADJUST TVS	5	67	0	
SINKS	GOOD TVS LEA	6	382		
SOURCES	CREATE TVS	1	382		

** CURRENT SEEDS **

```

SYSTEM = 610234094
DISTRIBUTION 1 = 44620978
DISTRIBUTION 2 = 406037205
DISTRIBUTION 3 = 192317993
    
```

NODE STATISTICS

#QUEUE NODES* NUMBER OF TRANSACTIONS IN QUEUE

NODE NUMBER	NAME	MEAN MEA(NUM,*)	OBSERVATIONS TIME OF OBS(NUM,*)	OF THE MEANS STANDARD DEVIATION STD(NUM,*)	STANDARD ERROR STE(NUM,*)	SMALLEST SMA(NUM,*)	LARGEST LAR(NUM,*)	CURRENT RUN OR BATCH ONLY MEAN CME(NUM,*)	OR BATCH ONLY TIME OF OBSERVATION COB(NUM,*)
2	INSPECT WAIT	1.42561	6829.15	1.64405	0.519893	0	16	0.487982	680.247
4	ADJUST WAIT	0.674818	6829.15	0.564294	0.178445	0	4	1.49339	680.247

#QUEUE NODES* TIME IN QUEUE INCLUDING ZERO WAITING TIMES

2	INSPECT WAIT	21.6831	10	22.1288	6.99773	0.0000E+00	197.3	6.91559	48
4	ADJUST WAIT	68.7826	10	46.7156	14.7728	0.0000E+00	272.8	126.984	8

#ACTIVITY NODES* NUMBER OF TRANSACTIONS IN ACTIVITY

3	INSPECT TVS	0.663770	6829.15	0.117624	0.371959E-01	0	1	0.666321	680.247
5	ADJUST TVS	0.683530	6829.15	0.163073	0.515683E-01	0	1	0.848396	680.247

#ACTIVITY NODES* ACTIVITY TIME

3	INSPECT TVS	10.0957	10	0.868443	0.274626	0.7268	33.72	9.44297	48
5	ADJUST TVS	69.6705	10	1.97769	0.625402	59.25	82.07	72.1399	8

#SINK NODES* TIME IN SYSTEM

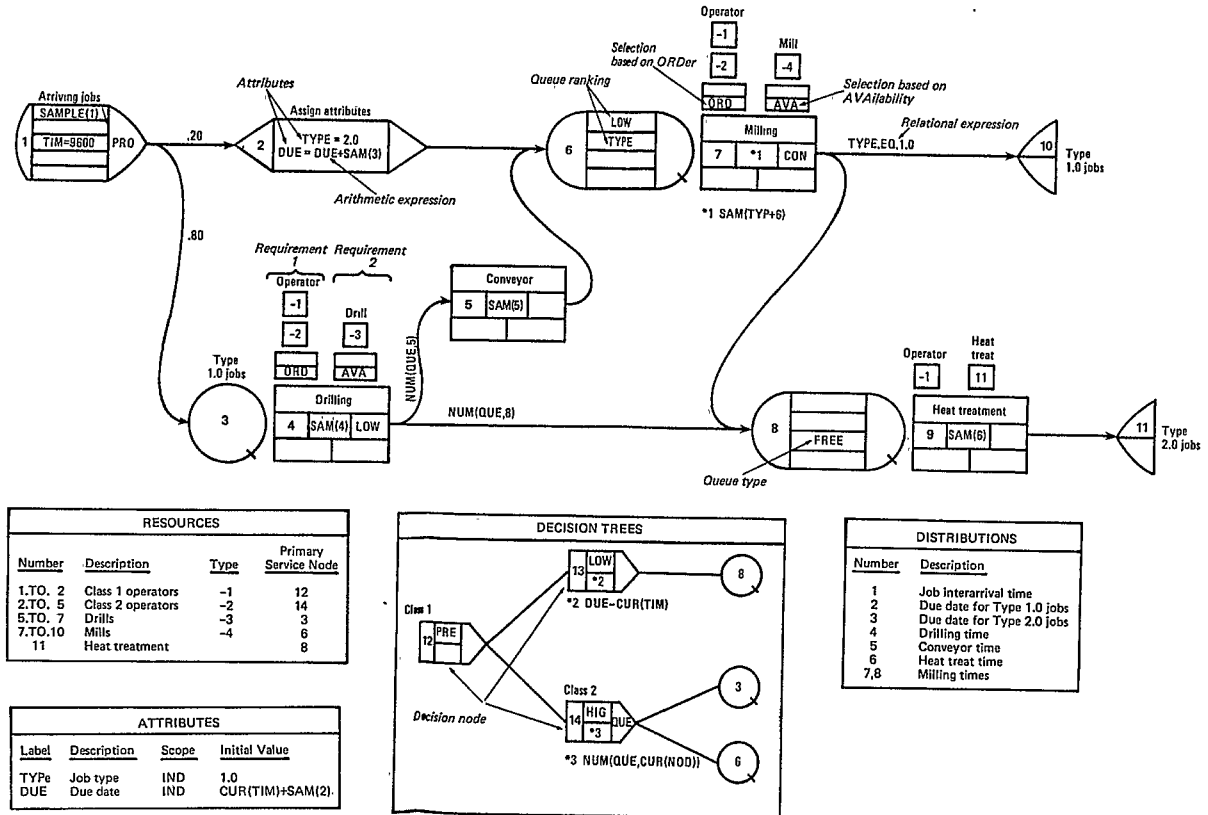
6	GOOD TVS LEA	61.6362	10	28.9446	9.15309	1.550	527.0	59.4551	40
---	--------------	---------	----	---------	---------	-------	-------	---------	----

RESOURCE STATISTICS

----- OBSERVATIONS OVER RUNS OR BATCHES -----
TIME EACH RESOURCE WAS OBSERVED = 6829.15

NUMBER(S) OR TYPE	NAME	MEAN UTILIZATION MEA(UTI,*)	NUMBER OF OBSERVATIONS OBS(UTI,*)	STANDARD DEVIATION STD(UTI,*)	STANDARD ERROR STE(UTI,*)	SMALLEST SMA(UTI,*)	LARGEST LAR(UTI,*)
1	INSPECTOR	0.66377	10	0.11762	0.03720	0.46288	0.83045
2	ADJUSTOR	0.68353	10	0.16307	0.05157	0.35170	0.84840

Figure 7: Flexible Production Department Network



important material handling occurs between the mills and the drills using a conveyor.

A model of the department would use transactions to represent the jobs and resources to represent the operators and the machines. The INS network model is given in Figure 7. Activities 4, 7, and 9 represent the drilling, milling, and heat treatment activities respectively. Jobs are created at Source 1 until the simulation TIME is 9600 minutes (20 eight-hour days). Each transaction is assigned a TYPE and DUE date by associating these values with transactions through INDIVIDUAL transaction attributes. The attributes are declared in a glossary to the network. Each transaction initially created has a TYPE of 1.0 and DUE date set to the CURrent simulation TIME plus a SAMple from Distribution 2. The specification of CUR(TIM) is just one of several SIFs available in INS which may supply information about the network known by INS.

Although 80 percent of the transactions move into Queue 3, 20 percent enter Assignment node 2 where their TYPE and DUE date are changed to reflect a different (the second) job type. Assignment nodes are used to change the values of attributes in the network. Notice an arithmetic expression can be used to compute specific values. Attribute values may be used throughout the network. At Queue 6, transactions are ranked by putting those with the LOWest value of TYPE first (i.e., jobs from drilling are ranked ahead of jobs just arriving). The activity time for milling is SAMpled from the distribution computed as the value

of TYPE plus 6 (SIF arguments may be expressions). Branching from the milling activity (Activity 7) is CONDitioned on the value of TYPE. If TYPE is equal to 1.0 (a relational expressions), then the job exits the department at Sink 10. Jobs of TYPE 2.0 branch to Queue 8 to wait for heat treatment.

Branching from the drills (Activity 4) is based on the LOWest value of the branching expressions following the activity. The SIF referring to NUMBER in QUEUE is used to supply the size of Queues 6 and 8. The shortest queue length consequently specifies where transactions branch. If movement is from Activity 4 to Queue 6, then transactions are delayed in Activity 5 to represent the movement on the conveyor. No resources are required for this activity.

Activities 4, 7, and 9 each require two resources, an operator and a machine, before the activity can begin. Both must be available simultaneously and the requirements are represented by the two columns of symbols above each activity. However, there are choices among alternatives for each requirement so the selection method is specified. Resources may be referred to by resource type using negative numbers. Thus resources of type -2 (class 2 operators) are in preferred ORDER to resources of type -1 (class 1 operators). This means that whenever a choice among resources is possible, Type -2 is considered before Type -1. Resources of the same type are chosen based on whichever has been AVAILable for selection (idle) the longest.

However, when an operator finishes an activity, its next service must be determined. This decision is accomplished by reference to the primary service nodes of the resources. Each machine group services a single queue but the operators may serve more. Their choices are represented by one (or more) decision trees, composed of decision nodes and queues. The decision trees are found in a network glossary. In this case, the class 2 operators serve a subset of the class 1 operators. Class 1 operators PREfer service at Queue 8. However, the service of a particular job from Queue 8 is based on LOWest slack time, given as the DUE date less the CURrent simulation TIME. Notice that the relative dispatching of each job can change dynamically and therefore, Queue 8 is designated as being FREE to permit any job to be served regardless of its position in the queue. If queue 8 is empty when considered or if the resource is Type -2 (class 2 operator) then Decision node 14 is employed. Queues 3 and 6 are examined and that QUEE having the HIGhest NUMber in the QUEE will be serviced.

Therefore, INS easily accommodates some of the most complex aspects of modeling job shops. Multiple and simultaneous resource requirements at activities are possible. Resource decision making can be incorporated. Branching involving conditions and expressions can be employed. Queue ranking and dynamic dispatching are easily handled relying on the attributes of transactions. The importance of these extensions is that they did not clutter the network and the modeling framework retains its visual and communication benefits. The specifications clearly document the specific actions as well as depict the general system under investigation. Furthermore, the modeler did no programming.

Embellishing the Models

These examples illustrate only the simplest features. Many other behaviors can be modeled similarly. Reneging and balking at queues can be directly incorporated. The definition of a queue can be broadened to include waiting for a gate condition and/or gather requirements to be satisfied. Transactions may be gathered and grouped at queues and then processed together in the associated activity.

Important realistic complications like resource preemption and its impact on other resources and transactions can be specified. Transactions can select resources in many ways and capture resources at a queue or activity. Many other features are incorporated, all within the context of the six INSIGHT modeling symbols (see (1)). Although such embellishments greatly extend the modeling potential, they are readily incorporated into the network without sacrificing its visual clarity.

Much of the generalization in INSIGHT modeling is due to the powerful specification expressions. Expressions are combinations of primitive elements including constants, SIFs, attributes, and functions (which are themselves expressions) which are evaluated as the simulation executes. The SIFs represent system-defined elements while the attributes and functions are user-defined. In addition to arithmetic, relational, and logical expressions, INS 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 upon whether the condition, TYPE equal to 1 or NUMBER in QUEUE 4 is greater than 5, is true or false. An iteration expression such as

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

adds to the present value of TOT the value of BUSY TIME for resource I iterating from I to NUMBER.

To employ information about any transaction in the network, INS provides an Internal Transaction Pointer (ITP) which the modeler can employ anytime an expression is evaluated. For example

```
ITP(3,5)+ABC
```

yields the value of ABC for the third transaction in node 5. The ITP plays an analogous role in modeling to the role pointer variables play in modern programming languages.

Expressions may be made arbitrarily complex to reflect sophisticated specifications and behaviors without changing the visual structure of the model. Because they are evaluated at run time, their specification can be symbolic to take advantage of state-dependent information. Thus the network serves primarily as a structuring tool while the specifications actually prescribe the model behavior. Note that with these considerable complications, there remains no need for programming.

STATISTICAL ANALYSIS

In addition to providing a high level approach to simulation modeling, INS 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 INS estimate variances by replications (separate simulation runs) or by batches (division of observations into subintervals). INS does not automatically provide a standard error from observations collected during a run because of their known lack of independence. This is why the output described in the previous section uses 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.

Start-up issues requiring special initial conditions can be specified directly within INS by a PRERUN statement that initializes variables and attributes

and inserts transactions in nodes. Truncating data during start-up is accomplished by SDFs 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 INS by direct specification. By default each distribution in the model has its own separate random number stream. There is no practical limit to the number of streams in INS. 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, INS employs inverse transform variate generators so that random numbers are inherently synchronized. By including these features in INS, statistical analysis can become an integral part of simulation modeling.

Special attention has been given to random number and random variate generators in INS. The random number generator (3) is completely portable to any machine having a 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 INS 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 new time-varying Poisson process (4). More discussion of statistical issues in INS is found in (5).

INTERACTIVE INSIGHT

The mini and micro computer implementations of INSIGHT have additional facilities to promote interactive modeling and analysis during four phases of computer involvement -- model input (MODELER), compilation (COMPILER), execution (SIMULATOR), and analysis (ANALYZER). These features eliminate the need for a modeler to know about system editors, compilers, linkers, etc. and from knowing the input and output conventions like statement order, fields, and SDFs in the INSIGHT language. Instead the system prompts for responses and uses its general knowledge base about INSIGHT and its understanding of the model to guide and respond to modeling requests giving error messages immediately, warning the modeler of potential problems, and offering general information. All the information is provided in 80 column format (the mainframe version presumes 132 column line printer for output) and tailored for display on a CRT.

Modeler

The MODELER facilitates the input of the INSIGHT model by generating the statement model from interaction with the modeler. When completed, the resulting model will be free of syntax and semantic errors. The principal value of the MODELER is that the user need

not know how INS statements are formed and instead can concentrate on modeling concepts. Figure 8 illustrates only a portion of a session. The MODELER

Figure 8: Session with MODELER

```
#model tvs
[reading file tvs.mod]

:llst node
-----
TYPE           NUMBER      NAME                BRANCHING INFORMATION
-----
source node    1          CREATE TVS          2
queue node     2          INSPECT WAIT        3 (assoc. activity node)
activity node  3          INSPECT TVS         6, 4
sink node      6          GOOD TVS LEA        none

:activity
node number = 4? 5
node name = ACTIVITY5? adjust_tvS
activity time = 0.0? smf(3)
number of resource requirements = 0? 1
select sub-entity #1
requirement number = 1? skip
resource alternatives = <unresolved?> 2
*where next = ;co? 1
abort input node = 0? icc

:queue
node number = 4? 1
node name = QUEUE4? adjust_wait
associated activity node = <unresolved?> 5
[Node 6 not an activity node]
associated activity node = <unresolved?> 5
queue ranking method = POS? icc

:create
[writing to file tvs.ins]
:exit
[writing to file tvs.mod]
```

questions the user for information about the model and interprets the response. If necessary, the MODELER can supply "help" indicating possible responses. Inappropriate or inconsistent responses will be detected and errors cited for reentry. Incomplete or flawed models can be stored and edited at a later time. The MODELER keeps track of all specifications and at any time can remind the user of deficiencies or incomplete models. The MODELER's advice is based on both its general knowledge of INSIGHT and the growing information about the model being built. A unique and important feature of the MODELER is its ability to accept and understand complex, even multi-line, expressions and to edit them conveniently. The expression parser checks the expression elements and questions any inconsistencies or misusage.

Because the output of the MODELER is the statement model, it can be uploaded to a mainframe or any other acceptable computer for further execution. The MODELER's use of knowledge bases gives it an artificial intelligence framework and removes from the user any responsibility for statement writing. With the MODELER, model construction is almost reduced to question answering.

Compiler

INSIGHT uses a two-pass compiler to construct an efficient executing simulation. The compiler acts quickly and does not require interaction except to note errors and request direction. Output from the compiler can be executed (simulated) immediately, stored for later execution, or even reviewed by the ANALYZER.

Simulator

Execution of the simulation with the SIMULATOR provides for extensive interaction during execution. The simulation can be executed directly or in "one-step" mode. In one-step mode the trace of each event

is displayed. Execution of the simulation displays the simulation run and time, and allows either a scheduled interrupt or user interrupt from the keyboard as illustrated in Figure 9. Interrupts can

Figure 9: Interruption of SIMULATOR

```

THE CURRENT RUN = 1 THE CURRENT TIME = 122.96820
YOU HAVE INTERRUPTED THE SIMULATION AT CUR(TIM) = 122.9682
YOU MAY NOW:
1. CONTINUE
2. UNLOAD
3. ENTER ONE EVENT MODE (which turns on the trace)
4. LEAVE ONE EVENT MODE (which may turn off the trace)
5. SCHEDULE THE NEXT INTERRUPT
6. STOP
CHOOSE ONE? 3
ENTERING ONE EVENT MODE
YOU MAY NOW:
1. CONTINUE
2. UNLOAD
3. ENTER ONE EVENT MODE (which turns on the trace)
4. LEAVE ONE EVENT MODE (which may turn off the trace)
5. SCHEDULE THE NEXT INTERRUPT
6. STOP
CHOOSE ONE?

```

be scheduled either by evaluation of a relevant SDF in the model or scheduled during the SIMULATOR session (see Figure 10). An important option during the

Figure 10: Scheduling Interrupts during Simulation

```

THE SIMULATION IS NOT NOW SCHEDULED TO BE
INTERRUPTED BEFORE THE END OF THE SIMULATION
YOU MAY SCHEDULE TO INTERRUPT THE SIMULATION:
1. AFTER SOME SPECIFIC NUMBER OF SUBSEQUENT EVENTS
2. AT A TIME DURING THIS RUN
3. AFTER LAST EVENT IN THIS RUN - BEFORE ENDRUN
4. BETWEEN ENDRUN AND POSTRUN
5. AFTER THE POSTRUN
6. BEFORE PRERUN OF A SUBSEQUENT RUN
7. DO NOT INTERRUPT BEFORE END OF SIMULATION
CHOOSE ONE? 2
AT WHAT TIME DO YOU WANT TO INTERRUPT? 60
NEXT INTERRUPT WILL BE AT 60.00000
Please press <return> to continue.

```

execution of the simulation is the opportunity to unload the model. An unloaded model saves the state of the simulation which can be reloaded later for continued execution or analyzed with the ANALYZER. If the execution is interrupted by an error, error information is automatically printed and the system state may be saved so that the ANALYZER can be invoked. Thus during the execution the simulation state can be saved, continued, and recovered allowing detailed examination through one-step or executed directly.

Analyzer

The ANALYZER provides an interactive means of examining the current state of the simulation and its statistical history. The micro computer implementation uses a menu-driven interaction, shown in Figure 11, that allows the user not only to review

Figure 11: ANALYZER Options

```

THE FOLLOWING OPTIONS ARE AVAILABLE:
1. INSIGHT REPORTS
2. STATE OF THE NETWORK AND NETWORK ATTRIBUTES
3. NUMBER(QUANTITY) OF TRANSACTIONS AND RESOURCES
4. STATUS OF THE CURRENT TRANSACTION
5. STATUS OF RESOURCES
6. VIEW SPECIFIC STATISTICS
7. MOVE THE TRANSACTION POINTER
8. CAUSE ACTIONS
9. EDIT MODEL
10. EXIT ANALYZER
CHOOSE ONE?

```

the model status and its statistics but also cause actions and edit the model without recompilation. The ANALYZER provides interactive access to all information provided by INSIGHT including that supplied by SDFs, attributes, and statistics. A variety of reports can be generated. Specific information about transactions are available by manipulation of the Internal Transaction Pointer (ITP). Statistics for mean values can include confidence intervals generated from direct estimates of the standard error (see Figure 12).

Figure 12: Obtaining a Confidence Interval

```

THE FOLLOWING STATISTICS TYPES ARE AVAILABLE:
1. NUMBER of transactions in a specific queue node
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 node
5. activity TIME in a specific activity node
6. total time in the NETWORK when leaving at a specific sink node
7. total time in QUEUES when leaving at a specific sink node
8. total time in ACTIVITIES when leaving at a specific sink node
9. statistical information on a specific TABLE
10. statistical information on a specific breakdown value
    or period of a specific table
11. UTILIZATION of specific resource(s)
CHOOSE ONE? 1
WHICH NODE NUMBER? 6
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
MEAN OF ALL RUN OR BATCH MEANS OF THE
NUMBER OF TRANSACTIONS IN QUEUE 6 = 1.00796
WITH A 95% CONFIDENCE INTERVAL HAVING 2 d.F. = [ .09222, 1.92370]
Please press <return> to continue.

```

An especially useful aspect of the ANALYZER is the extensive advisory facility based again on the joint knowledge base derived from the specific model and the general information drawn from INSIGHT. The key to such artificial intelligence arises from the well-defined model on which the INSIGHT simulation language is based.

FORTRAN INTERFACE

To allow conversation with other software, an interface with FORTRAN is incorporated which allows

two-way communication of information. Using the interface, INS can be used in trace-driven simulations. The interface can be used to store model-generated data in a file which is then available for further analysis. Complex or frequently used procedures may be written in FORTRAN to reduce execution time. SUFs and attributes may be used in the FORTRAN 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.

NEW ADDITIONS TO INSIGHT

Since August 1983, when INS was formalized (1), there have been several developments. An ENDRUN statement was added which is evaluated at the end of each run but before overrun statistics are collected. Two new decision modes, Conditional HIGH and LOW, are now available so that the resource's decision process can be based on the highest or lowest value of an expression but whose value must exceed zero. A new ITP (expression) is available to point the Internal Transaction Pointer to the transaction number given by the expression. Two new PRInt SUF options are available to print the contents of a specific node or just the event file. Printing of expressions during echo may now be suppressed.

AVAILABILITY

INSIGHT runs on any mainframe or mini computer that is a 32 bit machine and has a FORTRAN compiler. Core requirements depend on the operating system. Interactive INSIGHT is available for any desktop micro running MS-DOS or PC-DOS with a minimum of 256K memory. It can be configured for larger memory. The floating point processor (8087 co-processor) is highly recommended and a hard disk is helpful. A developmental version of Interactive INSIGHT is available for UNIX based systems. SysTech, Inc. at P.O. Box 509203, Indianapolis, IN 46250 is responsible for the distribution and maintenance of the simulation products.

CONCLUSIONS

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

INSIGHT is easy to learn because of its small symbol set and because INS networks closely correspond to the systems being simulated. Sophisticated INS models retain the simple modeling structure so that advanced concepts strengthen the fundamentals without expanding the network structure. Due to the ability to progress from simple to advanced concepts, INS has been taught at both the undergraduate and graduate level with equal success. The textbook details simulation modeling and analysis with INSIGHT. Many applications are described and it contains sections on simulation issues such as input modeling, random number generation, event processing, etc.

INSIGHT can be used on machines ranging from micros to mainframes. The interactive facilities available not only minimize the need for computer expertise but enhance the entire model and analysis process in simulation.

REFERENCES

1. Roberts SD. Simulation Modeling and Analysis with INSIGHT. Regenstrief Institute for Health Care, Indianapolis, Indiana, 1983.
2. INSIGHT User's Manual, Regenstrief Institute for Health Care, Indianapolis, Indiana, 1983.
3. Marse KJ, Roberts SD. Implementing a portable FORTRAN Uniform (0.1) generator. Simulation 41(4) October 1983.
4. Klein RW, Roberts SD. A time-varying Poisson process generator. Simulation 43(4) October 1984.
5. Roberts SD, Klein RW. Statistics collection, display, and analysis in INSIGHT. in review, 1984.