Proceedings of the 1990 Winter Simulation Conference
Osman Balci, Randall P. Sadowski, Richard E. Nance (eds.)

SIMULATION MODELING AND ANALYSIS WITH INSIGHT: A TUTORIAL

Stephen D. Roberts
Mary Ann Flanigan

SysTech, Inc.
P.O. Box 509203
Indianapolis, IN 46250

ABSTRACT

The INSIGHT simulation language uses a network representation to describe systems in a quick, simple, and compact fashion. This description can be entered and simulated using novel interactive computing 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 mechanisms. 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 and continuous simulation language that allows simulation models to be described quickly and compactly. Its fundamental concepts are easy-to-learn and easy-to-use. The emphasis on nonprocedural facilities and high level concepts allows simulation models to be built rapidly. In addition, the time consuming activities of debugging and remodeling are minimized, allowing users to focus on **problem-solving** rather than simulation mechanics.

A simulation language is not only useful in describing systems, but its use 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.

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 algorithmic resource decision making, multiple and simultaneous resource requirements, activity abortion, multilevel preemption, reneging, free queues, 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. *Permitting 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.* 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. *Incorporating computer graphics for model-building and analysis.* INSIGHT models can be constructed using novel graphical input within Microsoft Windows. The graphical environment automatically checks the input and generates the simulation model. Graphical analysis through animation is possible using the Wolverine Proof animation system [see Brunner and Henriksen 1990].

7. *Supporting both discrete and continuous simulation.* The INSIGHT language provides for the modeling of continuous as well as discrete systems without any special programming or coding. A unique feature of the INSIGHT representation of continuous variables is that their inclusion is incorporated into the network model.

8. *Being fully supported and extensively tested, by being used in practice and in the 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. 1988] and the interactive INSIGHT version [SysTech, Inc. 1990]. These documents provide specific information on implementation details, error recovery, time and space use, and statistical features.

1.2 Background

The INSIGHT language and its simulation environment have evolved from extensive actual experience over the past fourteen years. Since 1978 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.

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.

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.

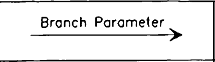
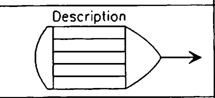
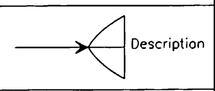
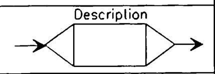
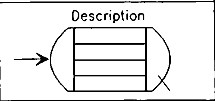
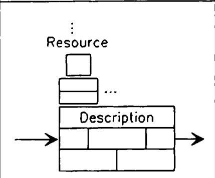
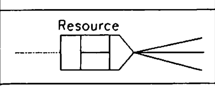
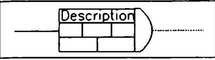
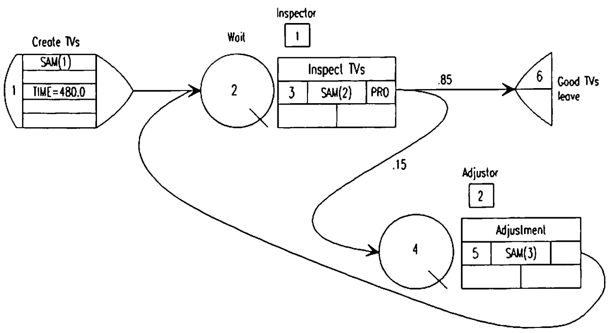
Node Type	Symbol	Processing Function
BRANCH		Connects two nodes in a network according to their precedence and specifies how transactions branch
SOURCE		Creates transactions and schedules their arrival to the network
SINK		Removes transactions from the network
ASSIGNMENT		Assigns values to attributes
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
DELAY		Represents a delay of the resource between activities

Figure 1. INSIGHT Modeling Symbols in Silhouette

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

2.1.1 *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. TVs that cannot find an idle resource at an activity must wait for service in queue nodes. 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.

2.1.2 *Getting help*

A very important dimension of the INSIGHT modeling environment is the on-line help. 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 the VISIFIT [DeBrota et al. 1989a] and FITTR1 [Debrota et al. 1989b] are available to the fitting of Johnson distributions.

2.1.3 *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

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. 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*.

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.

2.1.4 *The INSIGHT Statement Model*

The statement model is the intermediate, computer readable form of the simulation model, which is portable across a wide variety of machines. From the statement model, INSIGHT automatically compiles and executes the simulation and provides output.

2.1.5 *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 analyzed any time during its execution by scheduling an **interrupt** or striking a carriage return.

In the simulator, you have the freedom to configure the trace that 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.

2.1.6 *Analyzing the Simulation*

From the simulator, the user may enter the analyzer to obtain more detailed information about the progress of the simulation. In addition, you may also edit the model *while* it is executing. 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, during the simulation you can construct confidence intervals, display specific statistics and view attributes and their values.

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. Without any specific instruction from the user, the *Summary Report* presents the entire set of statistics that could be examined at any time.

2.1.7 *Networker*

To facilitate model building a new graphical network drawing facility, called Networker, has been developed. Networker runs under Microsoft Windows on IBM and compatible PC's. Networker allows the users to draw the network on the screen and edit it graphically. Networker is a mouse-driven "point-and-click" program that can be used to quickly and efficiently construct INSIGHT simulation models. Figure 3 shows a typical networker interaction.

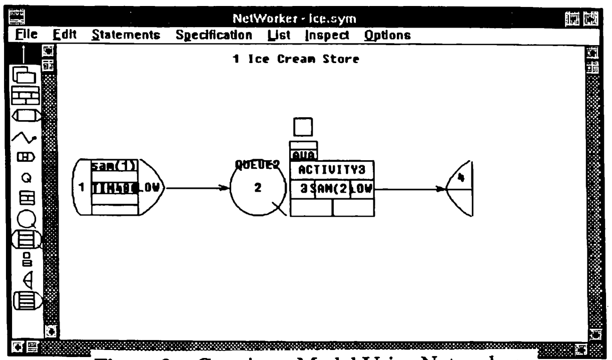


Figure 3. Creating a Model Using Networker

There are several features which make the Networker especially useful. First, all the INSIGHT symbols are pre-drawn and provided for easy use in the "toolbox." The toolbox, containing the INSIGHT symbols, is conveniently located down the left side of the window. Second, the specification of each symbol is known and as the user fills in the specifications, all the error checking is performed and problems noted. Third the graphical tools provide for the easy design and re-arrangement of the network. Other models may be employed or copied to the filing system. Finally, the automatic generation of the statement model means that the user does not need to translate the graphical representation into a textual representation, thereby eliminating errors and easing the burden of modeling.

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 user named and assigned values anywhere within the model. Attributes apply to transactions and resources. Transaction attributes may refer to a **single** transaction, to an entire **run**, the complete **simulation**, or to a **family** of transactions. Attributes for a family of transactions have an inheritance property in that all members of a 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. 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.

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 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.

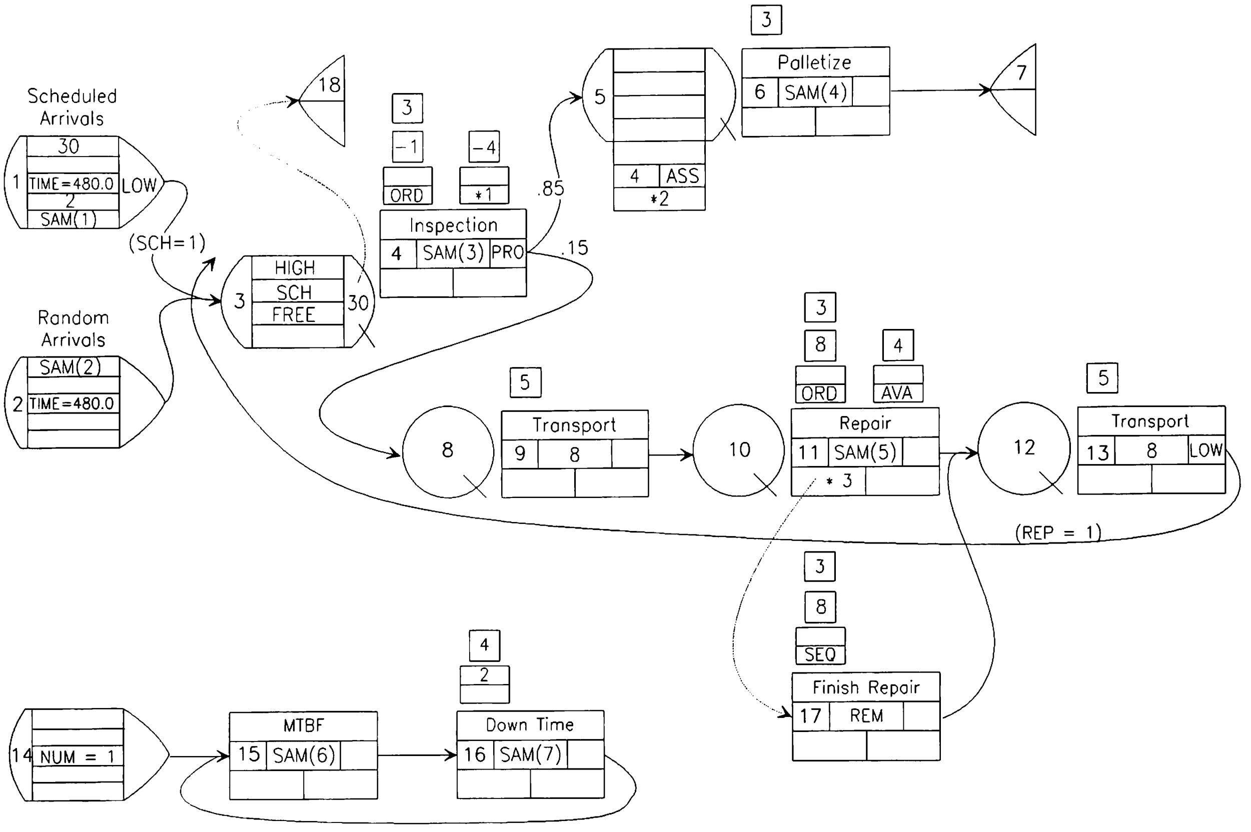
3. MODELING SOME PRACTICAL PROBLEMS

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 4, 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

3.1.1 Problem

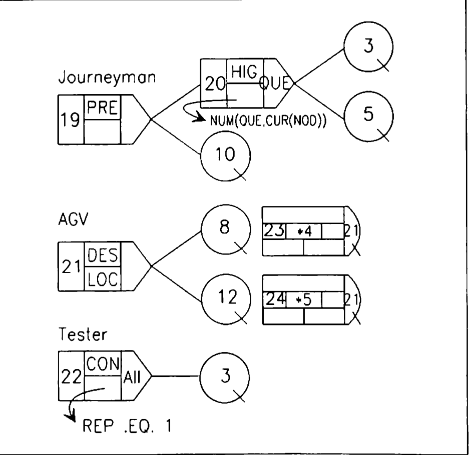
The TVs arrive in two broad patterns. Two TVs are regularly brought to the department by material handlers who are scheduled to arrive every 30 minutes. The actual arrival varies around the half hour schedule. The second pattern of arrivals is much less predictable and varies by the time of day. The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

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)
```

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 x = 0, lambda = 20, gamma = .647, and delta = 1.135

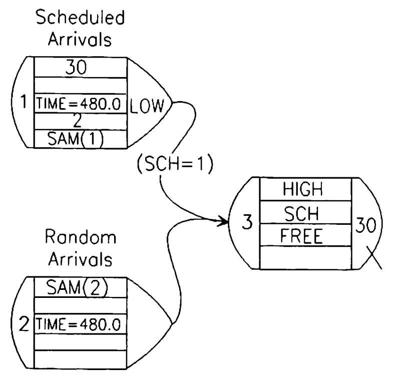
ATTRIBUTES

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 4. Fully Embellished TV Inspection and Adjustment Problem

3.1.2 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.



3.1.3 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.

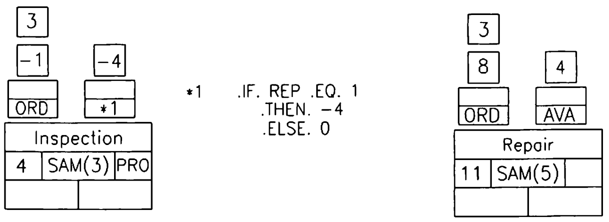
3.2 People and Machines – Simultaneous Resource Usage

3.2.1 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.

3.2.2 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 is not always used at the inspection activity, its selection is given by a conditional expression that yields -4 (a tester) when the TV has been repaired (REP .EQ. 1).



3.2.3 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.

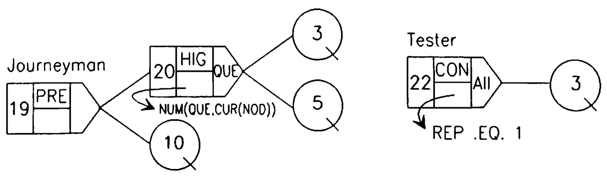
3.3 Resource Decision Making – What to do next?

3.3.1 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.

3.3.2 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, selecting the QUEue with the HIGhest Number 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).



3.3.3 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.

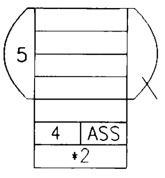
3.4 Gathering/Gating

3.4.1 Problem

TVs are palletized in groups of four. However scheduled TVs are palletized together separately from the randomly arriving TVs.

3.4.2 *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.



*2 SCH .EQ. ITP(CLA) + SCH

3.4.3 *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.

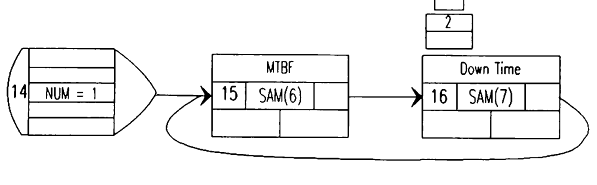
3.5 Breakdowns and Preemptions

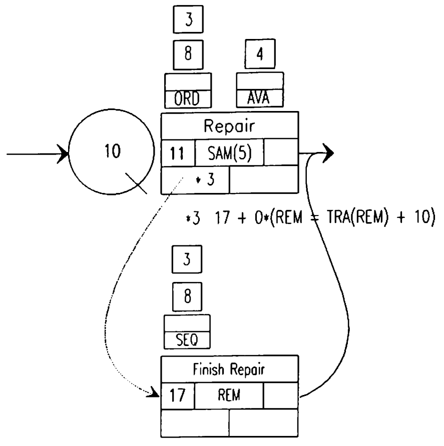
3.5.1 *Problem*

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 to compensate for the work being finished manually until the machine is fixed.

3.5.2 *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 TV in repair can either be suspended temporarily or abort the activity. In this case, the TV aborts the activity, and goes to Activity 17 to complete the repair. The same resource is taken SEQuentially from Activity 11 and is used to manually finish repairing the TV. The time remaining to repair the TV, before the preemption occurred, is retained, increased by 10 minutes (it takes 10 additional minutes without the machine), and is used in Activity 17 as the activity time.





3.5.3 *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 resource to return or it can be aborted to be completed in a different way.

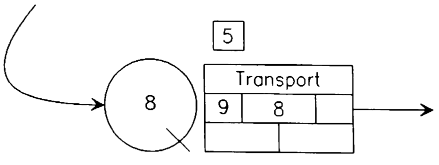
3.6 Resource Delays and Materials Handling

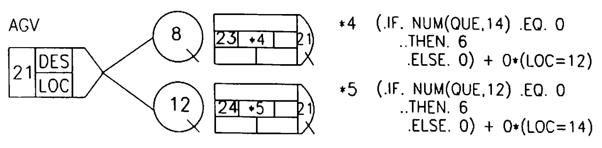
3.6.1 *Problem*

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

3.6.2 *Approach*

By defining the AGV as Resource 5, it can simply serve the transport activities (Activities 9 and 13) like other resources. When a resource must be active, but not in servicing 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 delay times specified by an expression conditioned on the availability of transactions in the associated queues. When the queue is empty, then a delay of six minutes occurs representing the travel time for the AGV to reach the next stop. If there is a transaction in the queue, it takes the AGV eight minutes to reach the next stop.





3.6.3 *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.

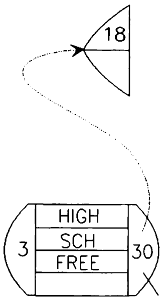
3.7 Reneging

3.7.1 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.

3.7.2 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.



3.7.3 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 or by batches. INSIGHT does not automatically provide an estimate of the standard error from observations known to lack independence. Similar computations could be obtained using batches when dealing with a steady-state simulation. INSIGHT will construct confidence intervals using the standard error, estimated as above, in either replication or batch mode.

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.

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 including these features in INSIGHT, statistical analysis can become an integral part of simulation modeling.

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

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. Data stored separately from the model may be changed without affecting the structure of the model.

4.3 Stations and Macros

File i/o and INSIGHT's flexible specifications allow the user to easily model workstations, and construct station macros. For example, a job shop manufacturing a variety of products, often follows the same process and/or procedure for all parts. The ability to construct stations can significantly reduce the size of the model and increase its extensibility and flexibility.

As a simple example, consider a manufacturing company that produces four different part types. Each part type goes through a series of processes. The order of processes, number of processes and the length of time at each process, are all dependant upon the individual part type.

This manufacturing company requires three different stations to produce parts A - D. Below is a possible sequence of processes for each part type.

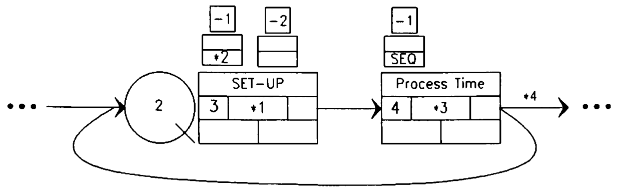
Part Type	Process Sequence
A	die-cutting, drilling, grinding
B	grinding
C	grinding, die-cutting, die-cutting
D	drilling, die-cutting

Note, each part type goes through a different series of processes, and a part does not have to visit all of the stations.

In INSIGHT, this system can be reduced to a simple network, as shown in Figure 45. This network is generic, easily altered, and convenient. Transactions represent the different part types, the workers and the different machines are resources, and the process sequences are maintained in one global two-dimensional array, called OPEration(*,*).

The process sequences for each of the different part types, are entered interactively with the FORMAT statement. They may also be entered into and read from a file. These sequences may be easily changed at any time during the simulation, thus allowing for a variety of sequence combinations, without modifications to the network.

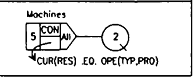
Each transaction (part) maintains its current process location, starting at the first process, in the individual attribute PROcess. This attribute is updated each time the part finishes at a particular process (station), until the part completes all of its processes, at which time the part advances out of the station macro and continues in the network.



RESOURCES

Number	Description	Type	Primary Service Node
1.TO.3	Machines	-1	5
4.TO.5	Workers	-2	2

DECISION TREES



EXPRESSIONS

•1	SAM(OPE(TYP,PRO))
•2	OPE(TYP,PRO)
•3	SAM(OPE(TYP,PRO) + 3)
•4	OPE(TYP,(PRO=PRO+1)) .EQ. 0

DISTRIBUTIONS

Number	Description
1	Set-up time for drilling
2	Set-up time for die-cutting
3	Set-up time for grinding
4	Processing time for drilling
5	Processing time for die-cutting
6	Processing time for grinding

ATTRIBUTES

Label	Description	Scope	Initial Value
TYP	Type of part	IND	0
PRO	Current process	IND	0
OPE(4,4)	Operation sequence	RUN	0

FORMAT STATEMENTS

1	'Enter the operation sequence for part A'
2	'Enter the operation sequence for part A'
3	'Enter the operation sequence for part A'
4	'Enter the operation sequence for part A'

Figure 5. Network Model Using Stations

Activity times are expressions determined from samples from distributions, which in turn are dependent upon the current process being performed. Instead of sampling from distributions, the activity times could have been read from empirical data contained in files, thus resulting in a data-driven simulation.

Using this station approach, this model can be used to represent a much larger system, one with additional part types, more processes and stations, and more machines. This can be achieved without adding complexity to the model. Hence, a model with 50 different part types, and 30 different processes will be virtually the same as a model with 1 part type and 1 process.

Stations are extremely versatile. Even the above small example can illustrate their powerful capabilities. Using the station concept and the FORMAT statement, the operation sequences can be edited, the number of part types can be altered, and the process times can be changed. All of these changes can be made without changing the statement model, re-compiling, or resorting to FORTRAN. In addition, the simulation model may be hidden from the user and the user will still have the ability to edit the model. This feature enables the modeler to create a generic model, and place it on the shop floor where a supervisor can enter the part type, or desired times, and then receive the desired output from the simulation. Most importantly, the supervisor does not have to be familiar with the simulation, and the modeler does not have to generate multiple models to consider all of the possibilities the supervisor may want or need.

4.4 Programming Interface

To allow conversation with other software, an interface is available which permits 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.

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 (1990), SysTech, Inc., Indianapolis, Indiana.

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 R.W. Klein (1989), "Statistical Analysis in a Simulation Language," In *Proceedings of the 1989 Winter Simulation Conference*, E.A. MacNair, K.J. Musselman, and P. Heidleberger, Eds. IEEE, Piscataway, NJ, 325-333.

DeBrota, D.J., Dittus, R.S., Roberts, S.D., Wilson, J.R. (1989), "Visual Interactive Fitting of Bounded Johnson Distributions," *Simulation* **52**.

DeBrota, D.J., Dittus, R.S., Roberts, S.D., Swain, J.J., Venkatraman, S., and Wilson, J.R. (1989), "Input Modeling with the Johnson System," In *Proceedings of the 1989 Winter Simulation Conference*, E.A. MacNair, K.J. Musselman, and P. Heidleberger, Eds. IEEE, Piscataway, NJ, 308-318.