Proceedings of the 1989 Winter Simulation Conference
E.A. MacNair, K.J. Musselman, P. Heidelberger (eds.)

STATISTICAL ISSUES IN A GENERAL-PURPOSE SIMULATION MODELING LANGUAGE

Stephen D. Roberts
Robert W. Klein
Regenstrief Institute for Health Care
1001 West Tenth Street
Indianapolis, IN 46202, U.S.A.

ABSTRACT

INSIGHT, a general purpose discrete event simulation language, has been designed to be easy to learn and use, making simulation accessible to a community of users having little or no previous simulation or programming experience. Because the language also presumes that its users have minimal statistical expertise, considerable attention has been given to statistical issues within the language, so users can enjoy the benefits of sound statistical analysis without the burden of detailed specifications. A wide range of statistical input distribution types are provided, including a time-varying arrival process and the univariate and multivariate Johnson System. INSIGHT simulations automatically produce a summary report, designed to represent the behavior of the simulation from several perspectives. The experimental framework of INSIGHT promotes proper computation of the variance of means. Variance reduction using correlation induction can be easily invoked. A non-procedural method of extending summary statistics and producing statistics on arbitrary values is available. INSIGHT's attention to statistical issues makes sound statistical analysis a natural partner of modeling. The convenience of such statistics facilities promotes the routine use of statistical concepts in examining and constructing simulations.

1. INTRODUCTION

Software design for scientific applications involves trade-offs among several competing objectives including numerical accuracy, size of code, and speed of execution. Design of a simulation language mandates consideration of modeling flexibility, ease of use, and portability. Statistical issues in simulation such as random variate generation, statistics collection, statistical display, and statistical analysis compound language design because they also impact modeling flexibility, ease of use, and portability. Various simulation languages exhibit different perspectives on how to incorporate statistical aspects of simulation.

Simulation languages such as SIMSCRIPT (Kiviat, et al. 1968) and SIMULA (Birtwistle, et al. 1975), offer statistics collection and display primitives, but require the modeler to incorporate them as a part of the modeling process within the execution of the simulation. This approach has the advantage of "forcing" the modeler to understand and specify detailed statistics collection and display within each model, but it also burdens the modeler with considerable responsibility for designing and writing statistics collection procedures, which often detract from the central modeling goals. To provide for details, these languages contain general purpose programming facilities that permit extensive manipulation of data. A major drawback is that they require fairly sophisticated programming skills in addition to a detailed understanding of statistical methods. As an example, Law (1979) presents routines in SIMSCRIPT for variance estimation using replications.

Some languages, such as GPSS (Schriber 1974) and SLAM (Pritsker 1986), routinely collect statistics and display results without any direction from the modeler. This approach relieves the modeler of the "burden" of statistics collection and display. However, because the modeler does not direct the collection and display, there is greater chance of misinterpretation of results and a loss of understanding. Such languages usually offer a more detailed statistics collection facility and report writer, but require procedural instructions within the model to execute the facilities, thereby diminishing their high level approach. Beyond these facilities, modelers must resort to another programming language and design an interface, for instance, using ASSEMBLER, PL/I, C, or FORTRAN. Furthermore, the inadequate variate generation in some languages mitigates against accurate modeling as well as statistics collection. Finally, in most simulation languages, the user must assume total responsibility for variance estimation and variance reduction.

SIMAN (Pegden 1987) requires a formal "experimental frame" as a part of each simulation. The experimental frame specifies most dimensions of the sampling experiment including the input distributions and the output. Each experimental frame must be tailored to the specific model and there is no provision for automatic statistics collection and display. Additional analysis of output is obtained by writing the individual observations to a file and processing that file by an output processor.

It is interesting to note that although the burden of statistics collection and analysis is extensive, provisions within simulation languages for easily accomplishing statistical tasks are rarely completely satisfactory. Law (1983) even states that the statistics which are available from some commercial languages can be biased and misleading. Three of the more important deficiencies of statistics in simulation languages are: (1) there remains considerable controversy about which statistics should be collected and how they should be displayed, (2) few simulation languages have attempted to include specific statistical procedures, choosing to leave their design and development to the user (who usually ignores them), and (3) no simulation language seems to offer a convenient, non-procedural approach to statistics collection and display of arbitrary simulation values (Henriksen 1983).

INSIGHT (Roberts 1983) addresses these concerns by providing three features: (1) an extensive, automatically produced report which summarizes the behavior of the simulation from several important perspectives; (2) a built-in, well defined set of statistical methods for variate generation, output analysis, and variance reduction, (3) a generally non-procedural method for easily extending the summary statistics and producing statistics on arbitrary simulation values. These facilities not only simplify incorporation of sound statistics, but enhance statistics use within simulation modeling. The purpose of this paper is to state how statistical aspects have been integrated into the simulation language and its utilities and to suggest why certain design decisions were made.

2. PURPOSE OF THE INSIGHT SIMULATION LANGUAGE

INSIGHT is a high level, general purpose, discrete event simulation language. It was designed to provide an easy-to-use simulation language that would extend the scope of simulation applications and broaden the community of simulation practitioners. Use of the language does not require prior simulation or programming experience and presumes limited statistical expertise.

The language is based on a visual or graphical representation of the system being modeled and is oriented, but not limited, to the modeling of complex queuing networks. Permanent fixed elements in the networks are nodes which represent fundamental processes like queues and activities. Temporary entities, called transactions, flow through the network causing the processes to be executed. The precedence among processes (nodes) in the network is formed by branches which connect the nodes through which transactions move. Resources are permanent but mobile network entities which may be needed to process a transaction at one or more activities. Resources may be active in deciding among competing resource requirements. Activities may have multiple resource requirements. Queuing occurs primarily for resource availability, but transactions also may queue for arbitrary system conditions or synchronization. Attributes of transactions, resources, and the network may be defined to establish model-specific characteristics. Complications such as balking, reneging, jockeying, preemption, etc. are all included as modeling concepts. Figure 1 shows a simple example of an INSIGHT network model.

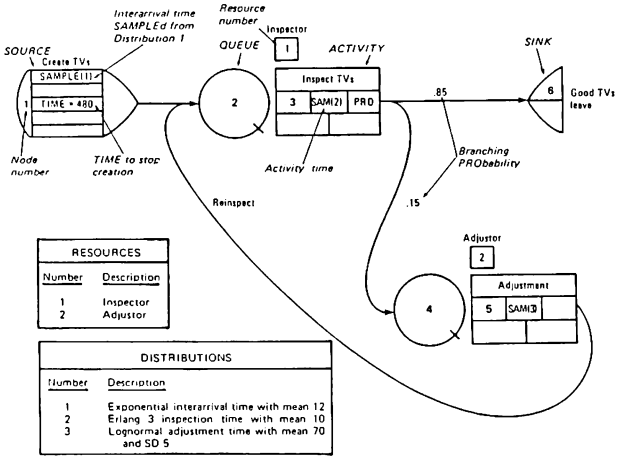


Figure 1: An INSIGHT Network Model

INSIGHT distinguishes itself from other similarly oriented languages by containing only seven node types with very flexible specifications. The network models generally correspond closely to the actual systems. Sophistication in model behavior is incorporated by unusual flexibility in describing the behavior of the nodes, so they can incorporate information about the system status, entity status, statistical information, etc. without resorting to general purpose programming. Thus, simple models are embellished by additional specifications rather than expansion of the network through additional nodes. INSIGHT graphical models, even complex ones, retain their visual appeal and concise description. Specification expressions add modeling flexibility not representable by a graphical analogue.

This overview is important because INSIGHT is directed at a user population composed chiefly of modelers for whom simulation is a tool for problem solving and "insight". Thus statistical, as well as modeling, facilities are required at a high level for routine use. Typical INSIGHT applications include production planning, bottleneck analysis, staffing, scheduling and dispatching, inventory control, material handling, facilities design, cost analysis, machine replacement, and productivity improvement. Generally, users are Industrial Engineers with undergraduate degrees, having only modest computer and statistical expertise. Few users are full-time simulation analysts.

The predominant mode of simulation with INSIGHT is interactive, usually on a personal computer or workstation. Modeling, execution, and analysis of simulation output can be done interactively. Extensive on-line help and support utilities are available. A textbook (Roberts 1983) and system documentation (SysTech 1988) describe the language and its computer facilities.

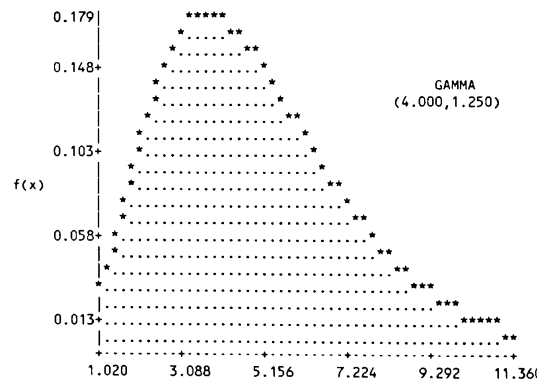
3. INPUT MODELING

Almost every simulation model requires some kind of statistical input. The broader the range of choices, the greater the variety of inputs that can be faithfully represented. Input models, such as distributions, should be easily specified from a wide range of alternatives.

The INSIGHT modeler can select from eleven standard statistical distributions for simulation inputs. These include the beta, binomial, Erlang, exponential, gamma, Poisson, lognormal, normal, Weibull, triangular, and uniform. Simple specifications transforming these distributions can be used to sample from other rarely selected ones like Chi-square, t, F, etc. Existing distributions also can be easily re-scaled for general use (see Appendices C and D in Roberts 1983). INSIGHT specifications can also accommodate various generation schemes such as inverse transform, composition, acceptance/rejection, etc. The parameterization of the standard distributions is aided by an interactive help facility, called InsHelp (SysTech 1988), that converts between the mean and standard deviation and the distribution parameters. The following interaction illustrates conversion between the mean and standard deviation and the Gamma parameters:

```
Enter the mean: 5

Enter the standard deviation: 2.5

For the Gamma with mean 5.000000 and standard deviation 2.500000

The shape alpha is 4.000000

The scale beta is 1.250000
```

Distributions also can be graphically displayed with the InsHelp facility, as follows:



In addition to the eleven standard distributions, a time-varying Poisson generator, the Johnson system of distributions and two arbitrary distributions are also available. One arbitrary distribution is for discrete (multinomial) input and the other for arbitrary continuous input. Either historical data or user supplied information can be used to specify these distributions. All distributions can be truncated to any minimum or maximum, and all random number seeds and sampling mechanisms may be controlled (for reasons to be described later).

The time-varying (non-homogeneous) Poisson arrival process generator models arrivals which, while satisfying a Poisson distribution, have a mean that varies linearly with time (see Klein and Roberts 1984). The distribution is specified by a piecewise, linear rate function and has extensive application in systems whose input changes throughout the day or are otherwise a function of time. Thinning can be used directly in other cases of time-varying processes whose rate function is a non-linear function of time.

The Johnson system of distributions (Johnson 1949) is available to provide even greater flexibility in the choice of inputs. This four parameter family provides for the modeling of distributions containing any skewness and kurtosis.

Parameterization of the distribution is greatly aided by the recent availability of software for the fitting of the distribution, both with and without data (Venkatraman and Wilson 1987, DeBrota, et al. 1989). Multivariate extensions of the family are also available within INSIGHT.

The specification of a statistical input is easy. Once declared, the samples may be obtained by using the System-Defined Function called SAMple (see the activity time specification in Figure 1). All statistical inputs are first described by a declaration. For example, the interaction with the INSIGHT Modeler to provide an Erlang distribution is given below:

```
:distribution
      distribution number = 1? 2
      distribution name = DIST2? Service Time
      distribution type = <unresolved>? Erlang
      mean = <unresolved>? 5
      order = <unresolved>? 4
      minimum = 0.0?
      maximum = [largest real number]?
```

The Interactive INSIGHT Modeler also makes numerous checks for statistical validity. For example, if a user requests a sample from a discrete distribution such as a Poisson for an interarrival time, the Modeler questions the specification (interarrival times would generally come from continuous distributions, such as an exponential). These warnings serve to remind users of the key role of the statistical input.

4. DEFAULT STATISTICAL OUTPUT

Because INSIGHT was targeted for an audience lacking programming skills and statistical expertise, a rather extensive automatic statistics collection and display was adopted to ease the statistics collection burden and promote a multi-dimensional view of the simulation behavior. Figure 2 contains an example of an automatically generated summary report. The decision to provide automatic statistics magnifies the importance of statistics collection and display considerations, since any standardization of statistics displayed cannot be highly model dependent. Determination of a set of statistics which is sufficient for most models, but not excessive for many, became a two part design decision. First, what entiti•s merit automatic statistics collection and second, what statistics should be collected? Related to the collection of statistics is the choice of numerical and statistical procedures.

4.1 Entities Meriting Statistics Collection

The orientation of INSIGHT restricts the entities which need to be described. Hence, a minimal set of statistics should depict the behavior of the "active" network entities. Transactions can reside only in queues or activities or they leave the network at a sink. Thus, node statistics which reflect the number of transactions and time spent by transactions in each activity and queue node, as well as time spent in the system for transactions exiting at sink nodes, were chosen. Resources can be idle, out of the network, delayed, or captured in a queue or activity. Resource statistics, therefore, were chosen to reflect the utilization of the resources. Four issues caused us to embellish this minimal set of statistics in designing the default summary report of output.

First, for any arbitrary observation period of a simulation, Little's Theorem (line length = arrival rate * waiting time) cannot be used to relate the time-weighted number in the node or system to the unweighted time in that node or system. Although some simulation languages compute the unweighted time in a node or system by dividing the total residency time by the number of observations (an adaptation of Little's Theorem), the approach has two serious deficiencies. If there are transactions present when the observation period begins and/or ends (as characteristic of transient simulations), then the number of observations during the observation period is unknown and there will be bias in the time computed. Also, by computing one statistic directly from the other, there is no opportunity to measure the variation of the computed statistic. Therefore, INSIGHT collects and displays both the unweighted time in queue or activity and the time-weighted number in queue or activity. The number in queue or activity is computed for the entire observation period while time in queue or activity is collected only for transactions leaving the queue or activity during the observation period (the user can adapt Little's Theorem to either statistic to obtain corresponding results).

Second, time in the queue is often variously interpreted. For instance, should a transaction which does not have to wait in a queue be included as a zero wait or be excluded from the computation? INSIGHT provides statistics on both waiting time including and excluding zero waits (GPSS also provides two waiting time statistics) since either or both may be meaningful in certain applications.

The third addition to the set of automatic statistics involves time in the system. In most simulations, a transaction need not visit all nodes and, in fact, may encounter the same nodes several times. Therefore, the time in activities and queues cannot easily be used to obtain the transaction's time in the system. Furthermore, in more complicated models where transactions may have resources preempted from an activity, resuming the activity at a later time, their time in an activity may exceed their activity time. As a result, INSIGHT collects total time in activities and total time in queues for each transaction and presents these as well as the total time in the system for transactions exiting a sink node during the observation period. The percent of time a transaction "waits" during its time in the system can be used to measure the "efficiency" of the system in processing transactions.

Fourth, the resource statistics present resource utilization for each resource or resource pool (a pool is a collection of identical resources), and resource type (a type is an implicit identifier of several resources). Resource utilization was defined for the purposes of the default output to be "time busy" divided by "time present." Time busy includes time a resource is captured in an activity, in a queue, or delayed. The time present excludes any time the resource is out of the network.

In addition to the statistics which summarize the behavior of transactions (node statistics) and resources (resource statistics), additional information was added to depict the status of the network, called the Network Status report (see Figure 2). This report presents the current state of the system and delineates the period of statistical observation. It contains a number of counts including total number of transactions created and transactions currently in nodes as well as a count of all transaction encounters for each node. Current values of global attributes as well as current distribution random number seeds are also included.

4.2 Statistics Collected

Having established what entities are contained in a summary report, the specific statistics collected for each becomes important. For any node or resource statistic, the mean, minimum, and maximum are important summary measures of location and range. The applicability and interpretation of a standard deviation varies with the method of statistics collection. How to properly compute the standard deviation of the mean remains a research topic (Law 1983) and is rarely addressed by simulation languages.

Two broad types of simulations may be conducted: transient (or terminating) simulations where statistics depend explicitly on initial and final conditions, and steady-state simulations which are independent of initial and final conditions. Transient simulations are usually analyzed by replications or runs. When INSIGHT is directed to use runs as the method of statistics collection, the mean of every run is treated as an independent observation (each run is independently seeded). Typically, the number of runs is specified. Using run means, the mean, a standard deviation of these values, and a standard error (i.e., the standard deviation of the mean computed from the standard deviation of the run means divided by the square root of the number of runs) is provided. Because the runs can cause the

INSIGHT 5.2 JULY 1989 SysTech, Inc

****
* *
* INS SUMMARY REPORT *
* *
* PROJECT NUMBER 1 *
* *
* TV INSPECTION AND ADJUST *
* *
* 7/12/89 *
* *
* STEVE ROBERTS *
* *
****

****
NETWORK STATUS
****

CURRENT TIME =	680.247	CURRENT RUN =	10	
CURRENT RTC =	0	CURRENT TRANS =	0	
CURRENT NODE =	0	CURRENT ACT =	0	
CURRENT RES =	0	NUM IN NET =	0	
NUM CREATED =	40	NUM DERIVED =	0	

MAXIMUM SPACE UTILIZED = 767/ 50000 OR 1.53 PERCENT
NODE STATISTICS LAST STARTED AT TIME = .000
RESOURCE STATISTICS LAST STARTED AT TIME = .000
NODE STATISTICS COLLECTION TIME SINCE LAST CLEAR = 680.247
RESOURCE STATISTICS COLLECTION TIME SINCE LAST CLEAR = 680.247

	NODE NAME	NODE NUMBER	NODE COUNT	NUMBER IN NODE	ZERO WAIT COUNT
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	OBSERVATIONS OF THE MEANS: TIME OF OBSERVATION	STANDARD DEVIATION	STANDARD ERROR
2	INSPECT WAIT	1.42561	6829.15	1.64405	.519893
4	ADJUST WAIT	.674818	6829.15	.564294	.178445

NODE NUMBER	NAME	SMALLEST	LARGEST	MEAN	ALL OBSERVATIONS: TIME OF OBSERVATION	SUM
2	INSPECT WAIT	0	16	.487983	680.247	331.949
4	ADJUST WAIT	0	4	1.49339	680.247	1015.87

QUEUE NODES TIME IN QUEUE INCLUDING ZERO WAITING TIMES

NODE NUMBER	NAME	MEAN	OBSERVATIONS OF THE MEANS: NUMBER OF OBSERVATIONS	STANDARD DEVIATION	STANDARD ERROR
2	INSPECT WAIT	21.6831	10	22.1288	6.99773
4	ADJUST WAIT	68.7826	10	46.7156	14.7728

NODE NUMBER	NAME	SMALLEST	LARGEST	MEAN	ALL OBSERVATIONS: NUMBER OF OBSERVATIONS	SUM
2	INSPECT WAIT	.0000	197.3	6.91560	48	331.949
4	ADJUST WAIT	.0000	272.8	126.984	8	1015.87

QUEUE NODES TIME IN QUEUE EXCLUDING ZERO WAITING TIMES

NODE NUMBER	NAME	MEAN	OBSERVATIONS OF THE MEANS: NUMBER OF OBSERVATIONS	STANDARD DEVIATION	STANDARD ERROR
2	INSPECT WAIT	29.7728	10	24.9016	7.87458
4	ADJUST WAIT	104.737	10	46.9386	14.8433

NODE NUMBER	NAME	SMALLEST	LARGEST	MEAN	ALL OBSERVATIONS: NUMBER OF OBSERVATIONS	SUM
2	INSPECT WAIT	.5038	197.3	9.76319	34	331.948
4	ADJUST WAIT	5.514	272.8	145.125	7	1015.87

ACTIVITY NODES NUMBER OF TRANSACTIONS IN ACTIVITY

NODE NUMBER	NAME	MEAN	OBSERVATIONS OF THE MEANS: TIME OF OBSERVATION	STANDARD DEVIATION	STANDARD ERROR
3	INSPECT TVS	.663770	6829.15	.117624	.371960E-01
5	ADJUST TVS	.683530	6829.15	.163073	.515682E-01

NODE NUMBER	NAME	SMALLEST	LARGEST	MEAN	ALL OBSERVATIONS: TIME OF OBSERVATION	SUM
3	INSPECT TVS	0	1	.666321	680.247	453.263
5	ADJUST TVS	0	1	.848396	680.247	577.119

ACTIVITY NODES ACTIVITY TIME

NODE NUMBER	NAME	MEAN	OBSERVATIONS OF THE MEANS: NUMBER OF OBSERVATIONS	STANDARD DEVIATION	STANDARD ERROR
3	INSPECT TVS	10.0957	10	.868443	.274626
5	ADJUST TVS	69.6705	10	1.97769	.625402

NODE NUMBER	NAME	SMALLEST	LARGEST	MEAN	ALL OBSERVATIONS: NUMBER OF OBSERVATIONS	SUM
3	INSPECT TVS	.7268	33.72	9.44297	48	453.263
5	ADJUST TVS	59.25	82.07	72.1398	8	577.119

SINK NODES TIME IN SYSTEM

NODE NUMBER	NAME	MEAN	OBSERVATIONS OF THE MEANS: NUMBER OF OBSERVATIONS	STANDARD DEVIATION	STANDARD ERROR
6	GOOD TVS LEA	61.6362	10	28.9446	9.15309

NODE NUMBER	NAME	SMALLEST	LARGEST	MEAN	ALL OBSERVATIONS: NUMBER OF OBSERVATIONS	SUM
6	GOOD TVS LEA	1.550	527.0	59.4551	40	2378.20

SINK NODES TOTAL TIME IN QUEUES

NODE NUMBER	NAME	MEAN	OBSERVATIONS OF THE MEANS: NUMBER OF OBSERVATIONS	STANDARD DEVIATION	STANDARD ERROR
6	GOOD TVS LEA	37.5501	10	26.2372	8.29692

NODE NUMBER	NAME	SMALLEST	LARGEST	MEAN	ALL OBSERVATIONS: NUMBER OF OBSERVATIONS	SUM
6	GOOD TVS LEA	.0000	322.0	33.6955	40	1347.82

SINK NODES TOTAL TIME IN ACTIVITIES

NODE NUMBER	NAME	MEAN	OBSERVATIONS OF THE MEANS: NUMBER OF OBSERVATIONS	STANDARD DEVIATION	STANDARD ERROR
6	GOOD TVS LEA	24.0861	10	4.81305	1.52202

NODE NUMBER	NAME	SMALLEST	LARGEST	MEAN	ALL OBSERVATIONS: NUMBER OF OBSERVATIONS	SUM
6	GOOD TVS LEA	.7268	275.6	25.7595	40	1030.38

****
RESOURCE STATISTICS
****

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

NUMBER(S) OR TYPE	NAME	MEAN UTILIZATION	NUMBER OF OBSERVATIONS	STANDARD ERROR	SMALLEST	LARGEST
1	INSPECTOR	.66377	10	.03720	.46288	.83045
2	ADJUSTOR	.68353	10	.05157	.35170	.84840

FOR THE CURRENT RUN OR BATCH
TIME EACH RESOURCE WAS OBSERVED = 680.247

NUMBER(S) OR TYPE	NAME	STATE NAME	TIME IN STATE	PERCENT OF TIME	FRACTIONAL UTILIZATION
1	INSPECTOR				
		IDLE	226.984	33.368	
		AT AN ACTIVITY	453.263	66.632	
		TOTAL BUSY	453.263	66.632	
		TOTAL PRESENT	680.247	100.000	.666321
2	ADJUSTOR				
		IDLE	103.128	15.160	
		AT AN ACTIVITY	577.119	84.840	
		TOTAL BUSY	577.119	84.840	
		TOTAL PRESENT	680.247	100.000	.848396

Figure 2: Automatically Generated Summary Report from INSIGHT

number of observations for unweighted statistics and/or the time of observation for time-weighted statistics to vary from run to run, each mean and standard deviation is computed by weighting the time-weighted values by their respective observation time period and the unweighted values by their respective number of observations. This latter step also tends to produce statistics which are more identically distributed among runs. The summary report in Figure 2 was produced by multiple runs.

Steady-state simulations are usually analyzed by batches within a single simulation run (although replications could be used). In directing INSIGHT to use batches as the statistics collection method, the user typically specifies a minimum number of batches for all statistics and stipulates a batch size (the number of observations of unweighted statistics composing a batch) and a batch interval (the length of time in which time-weighted statistics are observed to comprise their batch). If the resulting batches are large enough, they can be treated as independent observations, just like the means of runs in a transient simulation (no weighting is needed because the batch sizes and intervals are identical). With batch analysis, the number of batches is likely to vary among entities, so the number of observations or time of observation is also reported. The following interaction shows how easily a batch means analysis is produced:

```
:run
      collection method = RUN? Batches
      number of batches = 1? 10
      batch interval = <unresolved>? 500
      batch size = <unresolved>? 15
```

When statistics are collected into batches, output analysis follows the form of replications because the batch means are treated as independent observations. For example the following illustrates statistical analysis within interactive INSIGHT execution:

```
THE FOLLOWING OPTIONS ARE AVAILABLE TO OBTAIN STATISTICS:
    1. STATISTICS FOR A NODE
    2. RESOURCE UTILIZATION STATISTICS
    3. TIMES IN STATES FOR A RESOURCE
    4. STATISTICS IN A TABLE
    5. STATISTICS FOR A BREAKDOWN ROW IN A TABLE
    6. HISTOGRAM FREQUENCIES
    7. RETURN TO MAIN ANALYZER MENU

CHOOSE ONE? 1

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

CHOOSE ONE? 2

WHICH NODE NUMBER? 2

WAITING TIME INCLUDING ZERO WAITS IN QUEUE      2
MEAN OF ALL BATCH MEANS =      71.2405
STANDARD DEVIATION OF ALL BATCH MEANS =      61.4704
SMALLEST OBSERVATION WITHIN ALL BATCHES =       .0000
LARGEST OBSERVATION WITHIN ALL BATCHES =    297.7351
NUMBER OF OBSERVATIONS (AN UNWEIGHTED STATISTIC) =    64.0000
STANDARD ERROR OF ALL BATCH MEANS =      7.6838
MEAN OF CURRENT BATCH =       .0000
NUMBER OF OBSERVATIONS IN CURRENT BATCH =     13.0000
SUM (MEAN*OBSERVATIONS) OF CURRENT BATCH =       .0000

DO YOU WANT A CONFIDENCE INTERVAL FOR YOUR MEAN? y

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

CHOOSE ONE? 3

A 90% CONFIDENCE INTERVAL HAVING   63 d.f. = [ 58.4131, 84.0679]
```

An important difference in the execution of the simulation when batch means are used for statistics collection is that the length of the simulation run is no longer deterministic. The simulation run will last long enough for all the batch means to be collected. In the previous example, every statistic being collected must have at least 10 batches (more will be said about run length control later).

Whether statistics are collected by runs or batches, an addition to the summary statistics is made automatically to include the statistics for the last (current) run or batch. These values (refer to Figure 2) enable the analyst to examine the statistical characteristics of the simulation just prior to its termination. Resource statistics for the last run or batch contain the time each resource occupies each resource state.

When only one run is specified without having statistics collected into batches, the standard deviation is obtained from individual observations. Such observations are generally not independent and the standard error is not computed, to discourage erroneous confidence interval computations. No overrun or overbatch statistics are possible, so all statistics are relevant to the current (only) run. Furthermore, when the Interactive INSIGHT Analyzer is being used and there are no overrun or overbatch statistics, a warning is issued and no confidence interval will be computed. These actions remind users of the importance of understanding the statistics being collected.

4.3 Numerical Accuracy

An implementation decision essential in statistics collection involves the specific numerical procedure used to maintain the statistical records. Here, the trade-off among precision, space, and execution is particularly critical. Traditional calculator algorithms which require a sum and sum of squares are notoriously imprecise on computers, but used in many simulation languages. The large number of observations in a simulation magnifies the inaccuracy by producing large sums which overflow the floating point mantissa of most computer words resulting in a loss of precision and spurious values. INSIGHT uses the algorithms by West (1979) and Chan and Lewis (1979) to collect means and deviations (for unweighted and weighted statistics). These maintain a running mean and a measure of deviation which is a simple, quick calculation from a standard deviation. Therefore, the ease of obtaining intermediate results and the increased precision more than compensate for the extra space and slower updating of statistics.

5. EXTENDING STATISTICS COLLECTION AND DISPLAY

A modeler can request statistics on arbitrary values of the simulation by requesting a table on the identified values (see Chapter 5 of Roberts 1983). Tables can be specified as direct enhancements of automatically collected statistics or as new statistical entities. Enhancements of the automatically collected statistics might involve aggregation, such as the time in several queues, or breaking down the resource utilization by the locations where the resource serves. Tables can be specified as time-weighted or unweighted collections of arbitrary entities. For example, the time-weighted number in inventory or the unweighted cost of production may be collected automatically and displayed by declaring a table. Figure 3 contains two examples of INSIGHT tables, an unweighted time to obtain service and a time-weighted number waiting at a restaurant.

Tables provide more flexibility than the default statistics because they are individual statistical entities. A table may be compiled into runs or batches or simply collect individual observations. A table may specify a histogram which produces a distribution of values. Furthermore, a table can be broken down by the value or period of any arbitrary expression which is evaluated when an observation is collected into the table.

5.1 Table Declaration

The table declaration is a non-procedural specification of a desired statistic. For example, the following statement

```
TABLE, 2 TIME IN NETWORK, (7), NET, 11, 15, 21
```

** TABLE NUMBER 5 **
WAITING TIME FOR SERVICE

BREAKDOWN VALUE	MEAN	OBSERVATIONS OF THE MEANS: NUMBER OF OBSERVATIONS	STANDARD DEVIATION	STANDARD ERROR
PARTY SIZE: 4.000	16.6213	10	1.00071	.316451
PARTY SIZE: 2.000	12.6307	10	.835732	.264282
PARTY SIZE: 5.000	17.3542	10	.510459	.161421
PARTY SIZE: 1.000	11.4306	10	.443607	.140281
PARTY SIZE: 8.000	22.6605	8	2.20769	.780535
PARTY SIZE: 3.000	14.0575	10	.283149	.895396E-01
PARTY SIZE: 6.000	18.7222	10	1.01635	.321399
PARTY SIZE: 7.000	21.5801	10	1.34647	.425791
PARTY SIZE: 10.00	27.4532	4	5.06748	2.53374
PARTY SIZE: 9.000	24.0987	7	2.91454	1.10159
AGGREGATE	16.1626	10	.360968	.114148

BREAKDOWN VALUE	ALL OBSERVATIONS: SMALLEST	LARGEST	CURRENT RUN OR BATCH ONLY: MEAN	NUMBER OF OBSERVATIONS
PARTY SIZE: 4.000	12.1520	33.6300	16.9971	14
PARTY SIZE: 2.000	10.3918	18.4230	14.4837	8
PARTY SIZE: 5.000	13.2060	26.8368	17.5757	8
PARTY SIZE: 1.000	9.27554	15.8376	11.9552	6
PARTY SIZE: 8.000	18.6143	27.4368	20.2966	1
PARTY SIZE: 3.000	11.1698	20.8693	13.8004	18
PARTY SIZE: 6.000	14.8677	26.4948	19.0900	11
PARTY SIZE: 7.000	16.9282	29.3869	22.2457	3
PARTY SIZE: 10.00	20.8683	35.4519	30.8985	2
PARTY SIZE: 9.000	17.4410	30.2863	22.2422	2
AGGREGATE	9.27554	35.4519	16.6833	73

** TABLE NUMBER 6 **
NUMBER WAITING BY TIME

BREAKDOWN PERIOD	MEAN	OBSERVATIONS OF THE MEANS: TIME OF OBSERVATION	STANDARD DEVIATION	STANDARD ERROR
5 TO 6 PM 1.000	.185674	600.000	.325979	.103084
6 TO 7 PM 2.000	.995795	600.000	1.95874	.619407
7 TO 8 PM 3.000	3.16577	600.000	3.56404	1.12705
8 TO 9 PM 4.000	5.50680	600.000	4.62230	1.46170
9 TO 10 PM 5.000	4.17871	600.000	4.40021	1.39147
AFTER 10 PM 6.000	.967956	638.867	1.02648	.324601
AGGREGATE	2.48375	3638.87	2.19095	.692841

BREAKDOWN PERIOD	ALL OBSERVATIONS: SMALLEST	LARGEST	CURRENT RUN OR BATCH ONLY: MEAN	TIME OF OBSERVATION
5 TO 6 PM 1.000	.000000	5.00000	1.06918	60.0000
6 TO 7 PM 2.000	.000000	9.00000	6.46208	60.0000
7 TO 8 PM 3.000	.000000	14.0000	11.9160	60.0000
8 TO 9 PM 4.000	.000000	19.0000	14.2983	60.0000
9 TO 10 PM 5.000	.000000	17.0000	8.59685	60.0000
AFTER 10 PM 6.000	.000000	9.00000	.690494	64.6297
AGGREGATE	.000000	19.0000	7.08986	364.630

Figure 3: Tables Broken Down by Value and by Period

causes INSIGHT to create a table (Table 2) which includes the time in the NETwork for all transactions exiting at sink nodes 11, 15, and 21. Nothing else is needed as the statistics are collected automatically. To obtain resource use of resources 6 and 7, broken down by their five possible service locations, only the following statements are needed:

```
TABLE, 3 UTILIZATION, (7)BUSY, 6, 7
   BREAKDOWN, 3,, RES(NODE, CUR(RES)), 5
```

The convenience of such statements is that the modeler simply declares what statistics are desired without telling INSIGHT how they are to be collected. Even when a new statistical entity is being identified, only the table declaration is needed. For example,

```
TABLE, 4 INVENTORY, (7)WEIGHTED, INV(1.TO.10)
   BREAKDOWN, 4,, IDNUM, 10
```

requests a table on the time-weighted number in INVentory (not a native network entity) broken down by the model identification number (IDNUM), given here as an attribute. The modeler need only model the changes in inventory status in the network model, leaving INSIGHT to collect and "break down" the statistics automatically.

The interpretation of a table can be requested to agree with the method of statistics collection (i.e., runs or batches) or it can collect observations individually. This latter specification is recommended only when table entries are known to be independent. The statistical quantities in a table correspond to those described for the node and resource statistics (see Figure 3).

A useful dimension of the table declaration is that the breakdown expression can be interpreted as a value or period, somewhat analogously to weighting of statistical entries. A value distinguishes among table entries, whether unweighted or time-weighted, by the value of the breakdown. A period distinguishes among table entries by interpreting the breakdown expression as a time period. Figure 3 presents both types of breakdown criteria. The first table in Figure 3 is broken down by the size of the party (a value) waiting for service; the second table illustrates the number waiting at a restaurant broken down by hourly periods. Breakdown periods are especially useful in examining time-dependent behavior of a model, such as the build-up of a queue during the day or the difference in utilization of resources between two shifts.

5.2 Histograms

A histogram is an additional optional display of a table used to reflect the distributional characteristics of the table's statistic (the histogram applies only to the aggregate value in a table). The histogram may be a display of the distribution of values from an unweighted statistic or the distribution of time from a time-weighted statistic. Figure 4 shows examples of each of these two types of display. Note that both the relative frequency (density) and the cumulative frequency (distribution) are enumerated.

A complete specification of a histogram includes the number of histogram cells, the lower bound of the first cell, and the cell width. However, INSIGHT requires only specification of the number of cells and will adaptively construct a histogram. The other specified values for histograms are rarely known before a simulation and this feature can be of considerable convenience to the modeler, especially during earlier stages. For instance, the following table declaration,

```
TABLE, 1 WAITING TIME, 15, (7) EXC, 2, 4, 6
```

causes a table of the waiting time excluding zero waits in three queues and produces a histogram of 15 cells. No other instructions are required to obtain the entries for the table or the histogram, making both displays readily available with one non-procedural declaration. For final presentation, the cell width or lower bound may be declared explicitly.

5.3 Generalizing Batches

As the batch statistics collection method is implemented in INSIGHT, all statistics are collected into the same batch size or batch interval depending on whether they are unweighted or time-weighted. For example, if the waiting time at one queue in the network requires 100 observations per batch to achieve independent and identically distributed batch means, but another queue requires only 50 observations per batch, you have no choice but to specify a general batch size of 100. If, however, the second queue accumulates its observations slowly, then the simulation may run inordinately long to accumulate sufficient batches.

When the global batch size and batch interval are inappropriate for statistics collection, then statistics collection may be managed by tables. Two general tactics are possible. The first uses a global attribute to maintain the batch mean during the collection of statistics within the batch. This mean can then be collected directly into a table to produce the over batch results. The second tactic declares one table to collect the within batch observations and uses a second table to collect observations on means of the first table (the first table must be cleared when the batch mean is collected). Either of these tactics frees the modeler of a fixed batch size and batch interval,

however, more is required of the user for the simulation to collect and display meaningful results.

The idea of varying batch sizes and intervals can be extended to allow these to vary within a statistical entity for an observation period. For instance, we can obtain regeneration cycles by allowing the batch size and interval to be defined by regeneration epoches, detectable within the simulation. Therefore, we can collect the numerator of the ratio estimate in one table and the denominator in another. Unfortunately, to compute the variance of the ratio estimate will require an estimate of the covariance between the numerator and denominator which must be collected separately. Likewise, if jackknifing of results is needed to adjust for estimator bias, the adjusted estimates must also be kept in separate tables. Hence, it may be more convenient to use an external file to store statistics and calculate results.

5.4 External Data Files

INSIGHT can read/write formatted external data files during simulation execution. When read as input during the simulation, the simulation can be data-driven (trace simulation). Also the simulation can be interrogated and respond to user directives. Simulation output can be written to any file (including a terminal). Of special interest is that an output file can store information for further analysis by other software. It may also be used to print specially tailored output. A file reference is easily declared by

```
FORMAT, 2, INPUT.DAT, F8.3, ARR, F8.3, SER
```

and can be manipulated by the REAd and WRIte System-Defined Functions.

6. SIMULATION RUN LENGTH

The length of a simulation should be controlled by the statistics required from the experiment. For a transient simulation, the number of replications is specified. For a steady-state simulation, the minimum number of batches must be met. However, there are circumstances when greater control over the statistics collection period is needed. For example, the simulation models periods of time when no statistics are needed. Also, adaptive simulation run lengths may be desired to meet certain statistical requirements.

6.1 Specifying Statistics Collection Periods

Under several circumstances, the statistics collection period may be a subset of the entire simulation (composed of one or more runs). To obtain steady-state statistics, the statistics from a transient or warm-up period of the simulation may need to be deleted. In other simulations, certain time intervals such as lunch hours or shift breaks, should not contribute to the statistics.

System-Defined Functions instigate either the clearing (deleting) of already collected statistics or the starting/stopping of statistics collection. Either can be implemented within the actions of the simulation model through a specification or at an independent point in time through a monitor event. To clear all statistics (namely nodes, resources, and tables) at time 100, the following statement could be included:

```
MONITOR, CLEAR (ALL STATISTICS), 100
```

Similarly, statistics collection may be started or stopped. Furthermore, clearing and/or stopping can be constrained to particular statistics (e.g. resource statistics, one table, or even to statistics in the current run). These facilities allow individual statistics to be collected and monitored until some stability threshold is reached, then cleared to re-initialize statistics collection. While clearing erases all statistical memory of the past (while not affecting the simulation entities), stopping is like closing your eyes, i.e., no statistics are observed while statistics are stopped but previous information is not forgotten.

** HISTOGRAM NUMBER 1 **
STOCK LEVEL

TIME OF OBSERVATION	RELATIVE FREQUENCY	CUMULATIVE FREQUENCY	---CELL LIMIT--- LOWER	UPPER
.000000	.000000	.000	-INFINITY	.000000
8.39874	.093	.093	.000000	25.0000
19.8017	.220	.313	25.0000	50.0000
21.3357	.237	.550	50.0000	75.0000
20.7181	.230	.781	75.0000	100.000
10.1410	.113	.893	100.000	125.000
4.29797	.048	.941	125.000	150.000
1.12832	.013	.954	150.000	175.000
1.02584	.011	.965	175.000	200.000
.600853	.007	.972	200.000	225.000
.794107	.009	.980	225.000	250.000
.828005	.009	.990	250.000	275.000
.862989	.010	.999	275.000	300.000
.000000	.001	1.000	300.000	+INFINITY
90.0000				

LOWER	UPPER	
-INFINITY	.000000	+
.000000	25.0000	+****C
25.0000	50.0000	+********** C
50.0000	75.0000	+********** C
75.0000	100.000	+********** C
100.000	125.000	+******* C
125.000	150.000	+** C
150.000	175.000	+* C
175.000	200.000	+* C
200.000	225.000	+ C
225.000	250.000	+ C
250.000	275.000	+ C
275.000	300.000	+ C
300.000	+INFINITY	+ C
LOWER LIMIT INCLUSIVE		0.0 0.2 0.4 0.6 0.8 1.

** HISTOGRAM NUMBER 4 **
QUANTITY ORDERED

NUMBER OF OBSERVATIONS	RELATIVE FREQUENCY	CUMULATIVE FREQUENCY	---CELL LIMIT--- LOWER	UPPER
0	.000	.000	-INFINITY	.000000
0	.000	.000	.000000	2.00000
2	.015	.015	2.00000	4.00000
0	.000	.015	4.00000	6.00000
9	.067	.081	6.00000	8.00000
13	.096	.178	8.00000	10.0000
19	.141	.319	10.0000	12.0000
17	.126	.444	12.0000	14.0000
13	.096	.541	14.0000	16.0000
13	.096	.637	16.0000	18.0000
15	.111	.748	18.0000	20.0000
21	.156	.904	20.0000	22.0000
2	.015	.919	22.0000	24.0000
2	.015	.933	24.0000	26.0000
9	.067	1.000	26.0000	28.0000
0	.000	1.000	28.0000	30.0000
135				

LOWER	UPPER	
-INFINITY	.000000	+
.000000	2.00000	+
2.00000	4.00000	+C
4.00000	6.00000	+C
6.00000	8.00000	+***C
8.00000	10.0000	+****** C
10.0000	12.0000	+******** C
12.0000	14.0000	+******* C
14.0000	16.0000	+****** C
16.0000	18.0000	+****** C
18.0000	20.0000	+******* C
20.0000	22.0000	+******** C
22.0000	24.0000	+** C
24.0000	26.0000	+* C
26.0000	28.0000	+*** C
28.0000	30.0000	+ C
LOWER LIMIT INCLUSIVE		0.0 0.2 0.4 0.6 0.8 1.

Figure 4: Histograms for Time-Weighted and Unweighted Statistics

With clearing, the time of observation for time-weighted statistics is the time since the last clear. With stopping, it is the total time statistics are not stopped. In either case, an observation of an unweighted statistic corresponds to the occurrence of an event when statistics collection is on. For instance, time in queue is collected if a transaction leaves a queue after statistics have been started, even if it entered while statistics were stopped. Departures while statistics are stopped would not result in statistics collection.

6.2 Adaptive Run Length Control

Simulation lengths are generally prespecified, i.e., the number of runs when statistics collection uses replications or the minimum number of batches when using batches. Such specifications have the advantage of avoiding lengthy simulations, but often stop prior to satisfactory statistical precision. An alternative to specifying the length of the simulation in advance is to specify the precision of the final statistics. All statistics, including the standard deviation and standard error, are maintained internally and automatically by INSIGHT and they are accessible through System-Defined Functions.

The entire simulation or the current run in the simulation may be terminated by executing other System-Defined Functions. Therefore, a stopping rule can be directly implemented in INSIGHT by first specifying some arbitrarily large number of runs or batches to fulfill these requirements. Conditions for terminating the simulation based on sequential estimation procedures can be inserted after a run or batch. When these conditions are satisfied (perhaps based on some computed confidence interval or half width), the simulation can be terminated and results interpreted accordingly. The writing of these conditions is facilitated by a built-in t distribution which can be referenced directly. For instance,

```
POSTRUN, .if. CUR(RUN) .gt. 1
           .then. .if. (STD(NET,14) / MEA(NET,14) ** 2 .lt.
                       CUR(RUN) * ((DEL / TDIS(.05,CUR(RUN-1))**2
                  .then. TER(SIMULATION,CUR(RUN))
                  .else. 0
           .else. 0
```

this POSTRUN statement implements a procedure cited by Law (1980) where time in the network at node 14 is the important statistic. Here DEL is the attribute defining relative half-width and the TDIS returns a value of Student's t distribution with a specified alpha and n degrees of freedom.

The implementation of any such rules can involve considerable risk (particularly in a batch processing environment), for if values for the stopping rule are misspecified or conditions are "tight," the simulation may be excessively long and expensive. Pilot (interactive) runs are strongly recommended before using these approaches.

7. VARIATE GENERATION AND VARIANCE REDUCTION

Several key decisions in variate generation within INSIGHT were influenced by the decision to incorporate variance reduction techniques into the language. Specifically, because common and antithetic variates require only modest prior information (i.e., the monotonicity condition) and can be automated directly, we wanted to make them readily available and routinely useful. Arguments for the centrality of variance reduction in simulation are given in Bratley, Fox, and Schrage(1987). A number of design decisions, affecting both the random number and random variate generators, resulted.

A prime modulus random number generator was chosen to provide uniform random numbers between zero and one (Marse and Roberts 1983). The generator has been widely tested and has excellent global statistical properties. We were able to implement it in a portable fashion using standard FORTRAN (for machines having a 32 bit or larger word size). Furthermore, we developed a simple means to generate well spaced seeds and tested their local statistical behavior. Such an approach permits INSIGHT to assign a different (and well spaced) seed to each source of variation in a simulation model. Thus, there is no practical limit to the number of random number streams in an INSIGHT model and each source of variation is automatically assigned its own stream (the user can of course interfere with this assignment and specify seeds directly). The primary consequence of these decisions is that no two sources of variation need to share the same random number stream.

Most importantly, every random variate generator in INSIGHT uses an exact or approximate inverse transform to generate exactly one variate from exactly one random number. This decision has many ramifications. Experiments using INSIGHT models tend to have automatically synchronized random variates because each source of randomness has its own stream and each random variate is generated from exactly one random number. Such synchronization was deemed necessary for the use of either common or antithetic variates.

The use of inverse generators has two additional benefits. First, antithetic variates with the greatest negative correlation can be easily generated using inverse generators by replacing the original random number before transformation with its complement. This feature maximizes the variance reduction produced by correlation induction variance reduction techniques, such as common and antithetic variates. Second, inverse generators tend to yield very portable results and portability (i.e., getting the same results at different installations) was a primary design goal.

The decision to use a portable random number generator and inverse variate generators has been substantiated by profiling a wide variety of simulation models. The results showed that random number and random variate generation rarely exceeded three percent of the execution time of an INSIGHT simulation. Therefore, faster generators offer little as sources of computer time savings.

Different experiments with an INSIGHT simulation model automatically use common sources of variation. The modeler has the option of designating one or more sources of variation to be common among the runs of the same experiment. Antithetic variates can be specified for all the runs of one experiment. However, it is most useful to specify thetic-antithetic pairs of runs. Using stream control of the interactive Modeler,

```
:stream
        system seed = 19732729127
        system stream control = THE? ;help

        Alphanumeric beginning with:
          THE:  generate the THEtic stream without
                resetting the seed
          ANT:  generate the ANTithetic stream without
                resetting the seed
          COM:  generate a COMmon stream by resetting the
                seed to the original seed every run
          PAI:  generate PAIrs of runs, one thetic and one
                antithetic starting with a common seed;
                reset to a new seed for the next pair of runs

        system stream control = THE? pairs
        distribution control = NO? yes
```

the interaction provides a paired simulation experiment. In this case, one thetic run is made and then a second run replicates the first but employs the antithetic streams to form a thetic-antithetic pair. The second pair of runs has each seed set to the longest use of each stream. In this way synchronization is strictly maintained from pair to pair, further enhancing the opportunity for variance reduction. All of these procedures are instigated by a simple specification in the declaration of the distributions.

A by-product of the method used to pair thetic and antithetic runs is that streams for two variate generators may be coupled. Thus on one run, one generator uses the first stream and the second generator uses the second stream while on the second run the streams are switched. In this way, the stream that initially produced arrivals may be used to generate service times. As with paired runs, coupling also uses the longest use of each stream to seed the next pair of runs. The "help" illustrated

above, describes other options available for managing streams.

These variance reduction techniques are believed to have general utility (see Bratley, Fox, and Schrage (1987) for extensive discussion). They were made an integral part of INSIGHT to encourage their use (Chapter 7, Roberts 1983).

8. CONCLUSIONS

Because INSIGHT is targeted for an audience with minimal statistical expertise, considerable attention has been given to statistical issues within the language. A variety of input distributions, including a time-varying Poisson and a flexible family of univariate and multivariate distributions, are available for model building. Great care has been taken to provide automatic reporting on statistics which are meaningful and complete. Special attention has been given to their accuracy and possible interpretations. The summary report, whether for a transient (runs) or steady-state (batches) simulation, provides the modeler with relevant information not only about the means of important values, but also their standard deviations and ranges. These displays encourage the use of statistical information to insure that judgments have a sound statistical foundation.

The mechanisms to instigate these and other statistical methods are non-procedural. Thus, the modeler describes only what is desired rather than describing how it is obtained. The table collection facility illustrates how a variety of statistics can be collected and displayed within the language. Simple mechanisms extend to affect the statistical observation period and run length controls.

Variance reduction is available employing common, antithetic, and paired random variates. The random number and random variate generators were explicitly designed to accommodate the variance reduction techniques. Such convenient facilities should make variance reduction more routine and encourage greater attention to these important components of the simulation.

Experience with the language has been very positive. Its use has brought statistical issues from an incidental topic in a simulation study to a place of prominence and use. In the classroom, statistical concerns can be described jointly with the modeling. Such interaction can result in better simulation applications.

REFERENCES

Birtwistle, G. M., Dahl, O., Myhrhaug, B. and Nygaard, K. (1975). *Simula Begin*. Petrocelli/Charter, New York.

Bratley, P., Fox, B. L. and Schrage, L. E. (1987). *A guide to simulation*, second edition. Springer-Verlag, New York.

Chan, T. F. and Lewis, J. G. (1979). Computing standard deviations-accuracy. *Communications of the ACM* **22**, 526-531.

DeBrota, D. J., Dittus, R. S., Roberts, S. D. and Wilson, J. R. (1989). Visual interactive fitting of bounded Johnson distributions. *Simulation* **52**, 199-205.

Henriksen, J. O. (1983). The integrated simulation environment (simulation software of the 1990s). *Operations Research* **31**, 1053-1073.

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

Johnson, N. L. (1949). Systems of frequency curves generated by methods of translation. *Biometrika* **36**, 149-176.

Kiviat, P. J., Villanueva, R. and Markowitz, H. M. (1968). *Simscript II.5 Programming Language*. RAND Corporation, Santa Monica, CA.

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

Law, A. M. (1979). *Statistical analysis of simulation output data with SIMSCRIPT II.5*. CACI, Inc., Los Angeles.

Law, A. M. (1980). Statistical analysis of the output data from terminating simulations. *Naval Research Logistics Quarterly* **27**, 131-143.

Law, A. M. (1983). Statistical analysis of simulation output data. *Operations Research* **31**, 983-1029.

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

Pegden, C. D. (1987). *Introduction to SIMAN*. Systems Modeling Corporation, State College, Pennsylvania.

Pritsker, A. A. B. (1986). *Introduction to Simulation and SLAM II*, third edition. Systems Publishing Co., West Lafayette, Indiana.

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

Schriber, T. J. (1974). *Simulation with GPSS*. John Wiley and Sons, New York.

Venkatraman, S. and Wilson, J. R. (1988). Modeling univariate populations with Johnson's translation system - Description of the FITTR1 software. Research Memorandum 87-21, School of Industrial Engineering, Purdue University, West Lafayette, Indiana.

West, D. H. D. (1979). Updating mean and variance estimates -- an improved method. *Communications of the ACM* **22**, 532-535.

AUTHORS' BIOGRAPHIES

STEPHEN D. ROBERTS is Professor of Industrial Engineering at Purdue University and at the Indiana University School of Medicine. His academic and teaching responsibilities are in simulation modeling. His methodological research is in simulation language design and includes INSIGHT a general purpose, discrete event language, and SLN for the Simulation of Logical Networks. He is also a principal in SysTech, Inc. which distributes the simulation languages and consults on their application. He received his BSIE, MSIE, and PhD in Industrial Engineering from Purdue University and has held research and faculty positions at the University of Florida. He was Proceedings Editor for WSC '83, Associate Program Chairman for WSC '85, and Program Chairman for WSC '86. Presently he is Chair of ACM/SIGSIM, Associate Editor for Management Science, an Area Editor for Simulation, and WSC Board Representative for TIMS/CS.

Regenstrief Institute for Health Care
1001 West Tenth Street
Indianapolis, Indiana 46202
(317)-630-7447
steverob@gb.ecn.purdue.edu

ROBERT W. KLEIN has been employed for over ten years by the Regenstrief Institute for Health Care. He has held various positions including simulation analyst and health systems program coordinator. He helped design, develop and code substantial parts of the INSIGHT and SLN simulation languages, encompassing most of the statistical and interactive features. He has contributed to other software projects such as the TreeModeler decision analysis package. He has also developed simulation models applicable to health care issues such as treatment protocols for renal and heart disease and intern staffing schedules. He received his BS and MS degrees in statistics and a BSIE in Industrial Engineering from Purdue University. He maintains membership in SCS, SHS and IIE.

Regenstrief Institute for Health Care
1001 West Tenth Street
Indianapolis, Indiana 46202
(317)-630-7433