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

SIMULATION WITH MIC-SIM AND MIC-SIM VIEW

David E. Strack
Integrated Systems Technologies, Inc.
350 South Lowe, Suite C
Cookeville, TN 38501

ABSTRACT

MIC-SIM is a general purpose, discrete-event simulation system for use on IBM compatible microcomputers. It was developed as part of a doctoral dissertation in 1980 using Radio Shack computers. Microcomputers were new then and the only simulation languages available required either a mainframe or mini-computer. The MIC-SIM system was changed to work on IBM micro-computers soon after their introduction. The first version used a node and branch basis for representing simulation models. MIC-SIM was later changed to an activity based simulation system in order to improve ease of learning and to incorporate more powerful modeling constructs. This paper discusses the basic features of MIC-SIM and presents an overview on some of the modeling features of MIC-SIM.

1. INTRODUCTION

MIC-SIM is a simulation system designed specifically for microcomputers. It was created to make simulation modeling easy to do using inexpensive microcomputer hardware. All modeling features are built in to the system. No external computer languages are needed or supported.

MIC-SIM is a menu-driven system. A main menu is used for selecting the different features available in the package. Sub-menus and pop-up menus are used for all data input requirements.

Activities, queues and non-queues, are used by MIC-SIM to represent any type of "activity" that takes place within a simulation model. A network diagram of a model showing the activities and their relationships is all that is required to build a model. Entities flowing through a model determine the sequence of activities actually accomplished during a simulation run. Entities can have attributes attached and/or removed during a simulation, thereby also affecting the actual system flow. Resources, such as cash, raw materials, supplies, etc., can be consumed and created during a simulation. Resource levels can be used to control entity flow.

Once a model's general characteristics, such as the number of activities, queues, resources, entities, attributes, etc., have been entered, the data needed to complete the model definition is automatically requested by MIC-SIM. Default values are assigned to all data items prior to requesting the actual data.

An extensive amount of error checking is used during the model building process to insure that syntax errors are not committed. All non-numeric data entries are accomplished through the use of pop-up menus. The pop-up menus contain valid entries from which the user can make a selection. The entries in the pop-up menus are contingent upon earlier selections.

Ten different model validation areas are also checked prior to a simulation run to insure all model definition requirements have been met. If any errors are found then they must be corrected before MIC-SIM allows the simulation process to begin. Probable causes are reported along with suggested fixes.

A simulation can be run using either the animation or non-animation option. The animation option shows the entities flowing through the model using icons to represent activities and entities. The non-animation feature shows activities by name on the computer screen along with other information such as utilization percent, busy/idle data, queue size, resource quantities, and entity levels.

After a model has been simulated, simulation results can be viewed on screen, sent to a printer, or written to a disk file in ASCII format for use in spreadsheet or database programs. Activity and entity results can be viewed in MIC-SIM in either tabular format or chart format. Line and bar charts and histograms are provided by MIC-SIM.

2. ACTIVITIES

Queue and non-queue activity types are used to model all tasks and storage functions. Statistical data are automatically maintained for activities during a simulation run. The time for beginning the collection of statistical data is set by the user prior to beginning the simulation. Collection of additional data during a simulation for graphing purposes is optional for both types of activities. The frequency of data collection for graphing is also controlled by the user.

2.1. Non-Queue Activities

A non-queue activity, referred to as simply an "activity", definition is composed of 10 different data options - animation, graphing, time distribution, realization, entity creation, attributes, server requirements, system rules, resource rules, and exit rules. The animation option provides for icon selection if the activity is to be animated. Selecting the graphing option causes data to be collected for graphing. Each activity can use a probability function for representing the amount of time required to complete the activity.

The realization option allows the number of completions required to realize the activity to be set initially, as well as changing it during a simulation. The type of entity created by an activity can be specified. Attribute attachment and removal can also be set. When an activity needs more than one entity type and/or more than one unit of an entity the server rules option allows for specification of up to 10 multiple entity requirements.

System requirement rules can be used with each activity. These rules allow complex conditions such as pauses, waits, breakdowns, repairs, redirections, model modifications, and changes to be modeled quite easily. They follow the form of an "IF this condition exists THEN take this action". Activities can also check conditions such as queue sizes/balks/blocks, server status, simulation time, activity time, entity/attribute/resource amounts created or used, etc., to determine what action should be taken. Any of these options can be implemented before beginning the activity, during the process, or after completing the activity. Six of these rules can be used for each activity.

Four resource creation/usage rules can be assigned to each activity. These rules can be made to occur before, during, or after an activity is completed. Resource balances can be added to, subtracted from, and set equal to a quantity to using random values, other resource balances, activity times, entity levels, etc. Operations can be done using addition, subtraction, multiplication, division, and raising to a power. The results from one rule can be used in following rules to develop complex mathematical functions.

Exiting rules are used after an activity is realized for determining which activity(s) will receive an entity. Twenty rules can be assigned to each activity. More than one entity can be sent, and more than one activity can be chosen. Selection can be made by probability value, by entity type, and/or by attribute type. Deterministic, probabilistic, and combination branching is supported. Selection rules can be pre-empted and/or supplemented by system requirement rules.

2.2. Queue Activities

A queue activity, referred to as a queue, is used to represent a storage function. Each queue definition is made up of 8 data options - animation, graphing, initial number assigned, maximum number allowed, disposition technique, management rules, queue/server disciplines, and exiting rules.

Animation and graphing options are identical to activity options. The number and entity type of units assigned to a queue when the simulation begins can be specified. The maximum number of units that can be in a queue at any given time can also be specified. The disposition technique to be used when a unit arrives at a queue that is at its maximum capacity can be set to either balking or blocking.

Six queue management rules can be used to delete units from a queue based on some condition being true. Resource usage can also be changed while an entity is waiting in a queue. Attributes can be attached and removed based on the length of time a unit is in a queue.

Queue and server disciplines can also be represented without having to write any code. Entities can be selected from a queue by name or from all units waiting based on LIFO; FIFO; priority level; minimum/maximum waittime, service time, or resource level; randomly; or to jockey to another queue. Server selection can be based on idle time, busy time, randomness, or priority level. Combinations of up to six of these rules can be used for each queue.

Exiting rules for queues are identical to those discussed earlier for activities. Queues can exit to other queues or to activities, or both. Zero length queues are also permitted.

3. ENTITIES AND ATTRIBUTES

Entities are used to represent the items flowing through the model. Five hundred different entity types can be used in each model. Attributes can be assigned and/or removed to an entity as it travels through the system. Five hundred different attribute type are also available in each model definition.

Entities flowing through the system can contain a maximum of 10 attributes at any given time. Entity type and/or attribute types assigned to an entity can be used to determine the flow of the system. The number of entities created, used, and/or remaining can be used in system requirement rules and resource rules for decision making purposes and for mathematical functions.

4. RESOURCES

One hundred different resource types are available in each model definition. The number of units available at the beginning of a simulation can be specified for each resource type. Resource levels can be positive or negative during a simulation.

Each resource type can also be classified as either renewable or non-renewable. Renewable resources are returned to the resource balance as soon as an activity which uses them is complete. Non-renewable resources are no longer available to a model once an activity uses them. Resource creation and usage can be specified to take place before an activity starts, while the activity is in progress, or after the activity is complete.

The quantity of resources created, used, and/or remaining can be used in system requirement rules and resource rules for decision making purposes and for mathematical functions. They can also be used by queue discipline rules for selecting entities from a queue for service.

5. PROBABILITY FUNCTIONS

MIC-SIM has nineteen "built-in" probability distributions. They include the Beta (3 forms), Binomial, Chi-square, Constant, Erlang, Exponential, F, Gamma (2 forms), Geometric, Log-normal, Poisson, t, Triangular, Uniform (discrete and continuous), and the Weibull. All distributions can have a minimum and maximum allowed value assigned. Distribution parameters are checked during input for validity.

Fifty user defined discrete probability distributions can be defined for a model. Twenty intervals are provided for each distribution. Fifty pre-defined distributions created from the built-in distributions are available for each model. These distributions can be used by merely specifying their name without having to enter their parameters.

Probability distributions can be used to represent the time required to complete an activity, for calculating resource usage and creation, and for assigning probability values for such things as breakdowns, pauses, etc. Distribution values used for time representation are restricted to positive values.

6. RESULTS

Statistical time, resource, and service data in the form of mean, standard deviation, minimum value and maximum value are automatically maintained for all activities. Queue statistics for average queue size, average wait time, number of balks, number of times blocked and time blocked, minimum and maximum queue size, etc., are also automatically collected. Additional data can be collected for any activity or queue for use in line and bar charts. Statistical data can also be saved at the end of a simulation run for use in spreadsheet and database software packages.

7. MIC-SIM VIEW

MIC-SIM VIEW provides the ability to animate a simulation model during a simulation run. Icons are used to represent activities, queues, and entities on a user-created layout. An icon editor is provided for modifying/creating icon libraries. A layout editor is provided for constructing model layouts.

AUTHOR'S BIOGRAPHIES

DAVID E. STRACK is president of IST, Inc. He received a B.S.B.A. in Industrial Management from the University of Tennessee, an M.B.A., an M.S.I.E./O.R., and a Doctor of Engineering degree from Louisiana Tech University, in 1972, 1979, 1981, and 1986 respectively. He was an officer in the U.S. Air Force for 15 years where he served as a B-52 pilot, a management engineer, and an operations research analyst. From 1984 to 1987 he was an assistant professor in the Industrial Engineering Department of Tennessee Tech University. Dr. Strack has conducted numerous simulation studies during his military and civilian careers. The simulation seminars he has given have been attended by several hundred people. He is a member of Tau Beta Pi, Alpha Pi Omega, ACM, IIE, and SCS.

David E. Strack
IST, Inc.
350 South Lowe, Suite C
Cookeville, TN 38501
(615) 528-5280