Proceedings of the 1993 Winter Simulation Conference
G.W. Evans, M. Mollaghasemi, E.C. Russell, W.E. Biles (eds.)

FUNDAMENTALS OF SIMULATION USING MICRO SAINT

Lori Hood
Micro Analysis and Design, Inc.
4900 Pearl E. Circle #201 E.
Boulder, Colorado 80301, U.S.A

Dr. K. Ronald Laughery
Micro Analysis and Design, Inc.
4900 Pearl E. Circle #201 E.
Boulder, Colorado 80301, U.S.A

Susan Dahl
Micro Analysis and Design, Inc.
4900 Pearl E. Circle #201 E.
Boulder, Colorado 80301, U.S.A

ABSTRACT

Computer simulation is a technology which, while it has been available for almost thirty years, is just beginning to become a commonplace tool in the engineering tool kit. There are many reasons for this including more available computational power on the desktop, better user interfaces, and a greater need for finely tuned systems to maintain competitiveness in today's marketplace. Most engineers have had some exposure to the tools and technology of computer modeling and simulation, however, in this era of rapidly changing technology, the understanding of when simulation might be a cost-effective solution and how to approach the use of simulation can quickly become dated.

1 INTRODUCTION

Discrete event simulation has been a standard technique in the analysis of manufacturing systems for years. However, today's rising costs and competitive pressure has led to an increased awareness of the value of simulation in evaluating alternative strategies to reduce costs in other areas of industry as well. The adoption of total quality management (TQM) and process management techniques has created new challenges for managers and engineers. Process definition, quality measurement and control, process re-design, employee workload, safety and productivity are all areas where simulation is now being used to evaluate and improve efficiency. Decision makers within an organization are applying simulation technology to a wider variety of problems, consequently, the need for general purpose simulation tools that are capable of addressing all of these needs has increased. Micro Saint, a task network based modeling tool, has been found to be an efficient and cost-effective tool for simulating the complexities of systems within manufacturing, health care, retail, government, human factors and others. The problems being analyzed range from process control and resource utilization to military maintenance procedures and human performance.

The purpose of this paper is to provide a basic understanding of the principles of modeling with Micro Saint.

2 METHODOLOGY

Micro Saint was developed in 1985 specifically for modeling human performance in complex systems. One of the biggest challenges in the original development project was to provide a powerful and flexible tool that could be used by psychologists and human factors engineers. The target users for the product were not simulation experts or computer programmers. Their limited exposure to both simulation and computer science required a modeling approach that was much different from other simulation products that were available at the time. The engineers at Micro Analysis and Design, Inc. chose to develop Micro Saint implementing a methodology known as task network modeling, in which activities are represented in a diagram as nodes, and the arrows between the nodes represent the sequence in which the activities are performed. A simple task network is shown in Figure 1.

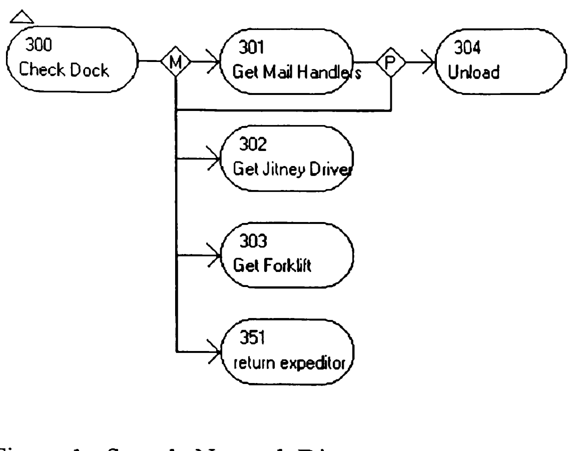


Figure 1. Sample Network Diagram

This approach allowed users to develop models using the same techniques that they would use to define a flow diagram of the activity. Each activity, whether it be a human activity or a system activity, is defined using the same method. This minimizes the complexity of the user interface and eliminates the need for programming blocks specific to an application.

Over the years, the size and scope of the problems to which users wanted to apply Micro Saint increased. This wider acceptance and use of Micro Saint in other application areas, including manufacturing and service industries, greatly influenced subsequent product development and enhancements. For example, there is rarely a need for defining queues when developing models of human performance, however, in order to remain competitive in the manufacturing market this capability to specify a queue and define parameters such as the exit order was critical.

Micro Saint has continued to evolve over the years; however, it's origin in human performance modeling is still evident in the terminology that is associated with the product. A Micro Saint model is composed of "networks" which may be a sequence of tasks to be performed by a human, a series of processes that define an organization, or a machine in a manufacturing plant. Networks are composed of either additional lower-level networks or "tasks." Although the identifier "task" has connotations of human activity, it is not restricted to such. Tasks represent the lowest level in the model and have specific parameters (timing information, conditions for execution, beginning and ending effects). The following section explains the task parameters in more detail.

Task Timing Information

The task "mean time" is the average time that a task takes to complete once it has begun executing. For example, if the task represents a human activity such as "transfer patient to recovery room," then the mean time to execute is the average time that it takes to perform the task. If the task represents a machine in a manufacturing process, then the mean time to execute is the average processing time for the machine. In many cases, the execution time is not constant. Rather, the elapsed time falls within a range of values that can be represented by a time distribution. Micro Saint supports more than 14 distribution types including normal, rectangular, exponential, gamma, wiebull, poisson, and others. In addition, users may enter parameters that control the spread of the distribution.

Alternatively, the mean time may be determined by the current state of the system or by an attribute of the process itself. In human performance modeling, the mean time to perform a task may be influenced by such conditions as how long the human has been working, the skill level of the human, or the current workload. In an insurance claims processing model, the time it takes to process a claim may be determined by the type of claim or the location of the client.

Conditions for Execution

Often, there are situations where a task cannot begin executing until certain conditions are met. A customer cannot make a transaction, even though the queue is empty, until a bank teller is available. A task may have resource requirements or other constraints (i.e., time of day, part type) that dictate when the task may begin executing. In Micro Saint, users enter a Boolean (logical) expression in the "release condition" field to control the execution of tasks. The release condition expression may be as simple as "teller <> 0", or it may be a complicated expression where several conditions are evaluated such as, "(clock > 8 & clock < 16) & (clerk <> busy)." Entities moving through the network, whether they be patients, parts, or claim forms cannot be released into a task for processing until the release condition for the task evaluates to true.

Beginning/Ending Effects

The current state of the system may change when a task begins or ends. For example, when a machine begins processing a part it becomes "busy" and is not available to another part until it has finished. The user would define the following expressions in Micro Saint to define this condition:
Release Condition: busy == 0[1]; This keeps an entity (part, patient, etc.) from moving into the task when the task is "busy" processing another entity.
Beginning Effect: busy := 1[2]; This sets the busy flag to TRUE so that the next entity cannot enter the task. As long as the task is executing, the busy flag will remain equal to "1".
Ending Effect: busy := 0; When the task finishes executing, the ending effect is evaluated and the busy flag is set to 0. Now, when the release condition is evaluated the condition will be true and the next entity can enter the task.

The relationship between the release condition and the beginning and ending effects provides a general, yet powerful mechanism for users to define complex behaviors within the system they are modeling. Users may define variables that are specific to their system and manipulate the value of the variables as needed so that they can accurately represent their system. They do not have to compromise the accuracy of their model by relying on pre-defined "blocks" within the modeling tool nor do they have to learn a complex programming language in order to obtain the level of control required.

A feature that greatly increases Micro Saint's power is the "parser" that evaluates algebraic expressions. It provides the mathematical power of computer programming languages such as FORTRAN or C, but eliminates the need for compiling the model before running it. One of the biggest advantages of the parser is that it allows users to interactively change the value of model parameters while the model is executing. For example, the user could increase the number of resources available or change the execution time for a task while the model is executing to see what the overall effects on the system would be.

3 MODEL DEVELOPMENT

The process of building a Micro Saint model involves two separate but interrelated steps. First, the user must define the structure of the task network. This is done by selecting the appropriate tool from the tool palette and placing the object in the drawing space of the network diagram. The Micro Saint tool palette and drawing space is presented in Figure 2.

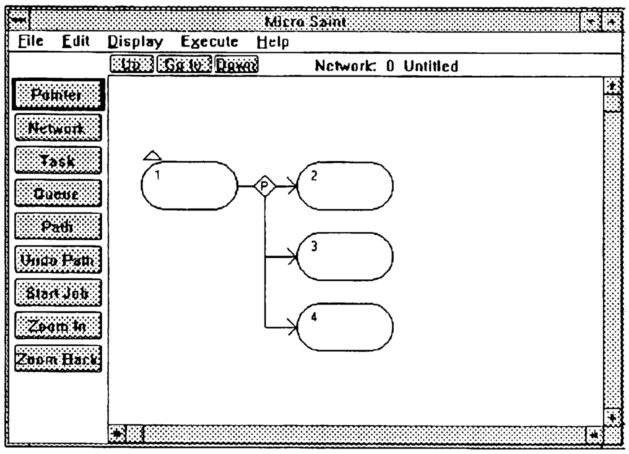
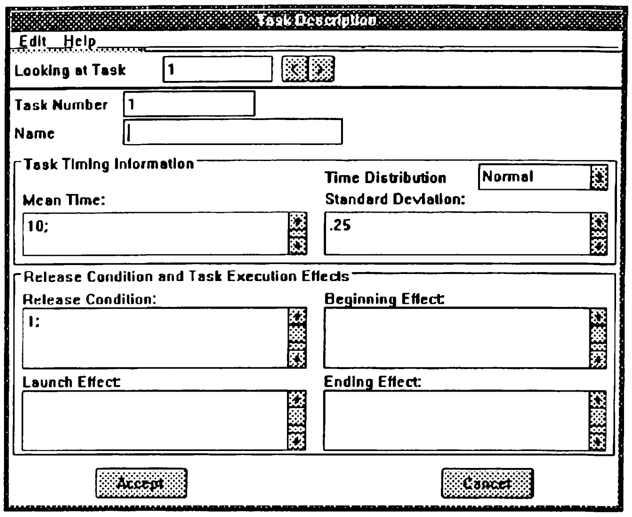


Figure 2. Micro Saint Tool Palette

Micro Saint uses a standard windows "point and click" approach to define the network objects. Using the mouse to "double-click" on an object will open it so that information specific to the object may be entered. Figure 3 is an example of the Task Description window that is opened when the user double-clicks on a task (i.e., node in the network.)



Looking at Task 1

Task Number 1

Name

Task Timing Information

Time Distribution: Normal

Mean Time: 10;

Standard Deviation: .25

Release Condition and Task Execution Effects

Release Condition: 1;

Beginning Effect:

Launch Effect:

Ending Effect:

Figure 3. Micro Saint Task Description Window

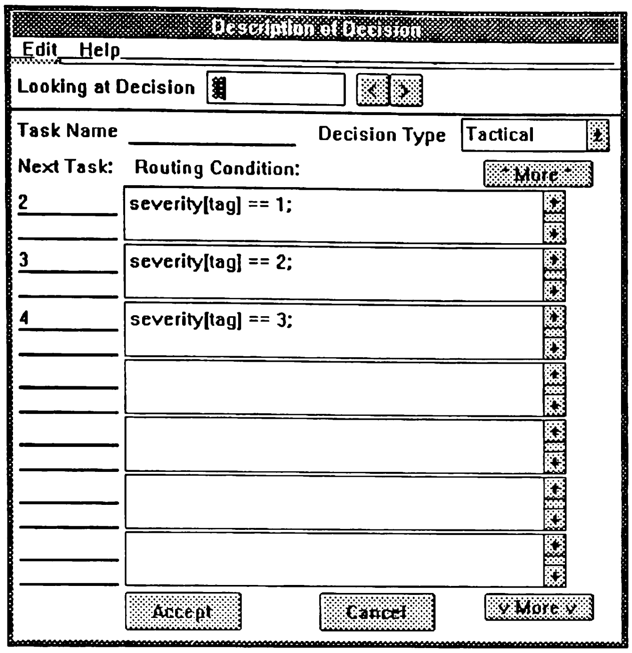
Task sequencing is defined by clicking and dragging with the mouse from the first task to the following task(s). A diamond shaped "decision icon" appears on the network diagram when more than one following task is defined. Users must enter the conditions that control the branching when there is more than one following task. Micro Saint provides the following decision types to ensure that all real-world situations may be represented in the model:

Probabilistic - The following task conditions are evaluated and the next task to execute is determined by the relative probabilities of all tasks listed. Probabilistic decisions allow only one of the following tasks to execute.

Multiple - The following task conditions are evaluated and all of the tasks whose conditions evaluate to non-zero will execute.

Tactical - The following task conditions are evaluated and the next task to execute is the task whose condition evaluates to the highest value.

Variables and algebraic expressions can be used in the branching logic and the value of the variables can be changed by conditions in the model. This gives the user complete control and manipulation of the network flow. Figure 4 is an example of a tactical decision that controls the flow of patients in an emergency room model through the network based on the seriousness of their injury.



Looking at Decision	
Task Name	
Decision Type	Tactical

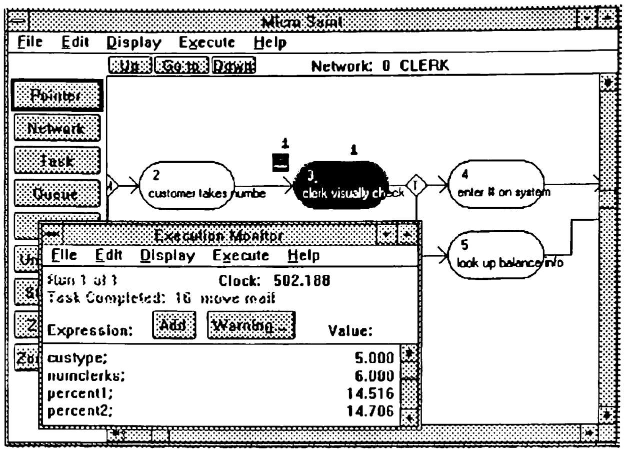
Next Task:	Routing Condition:
2	severity[tag] == 1;
3	severity[tag] == 2;
4	severity[tag] == 3;

Figure 4. Micro Saint Tactical Decision

All of these features work to provide an environment for the model developer that is easy to learn and easy to use. Once the basic concepts are understood, any system or process can be modeled using Micro Saint. In addition, users can build models at any level of complexity. Some applications may require a very low-level, detailed definition while for others, a high-level definition is sufficient.

4 ANALYSIS AND RESULTS

People build models to provide insight to, or answer specific questions about, a system or process. Some information can be gained by watching the Micro Saint model run. Micro Saint's symbolic animation capability provides an animated view of the network diagram as the model is running. Users can watch as entities flow through the network or wait in queues before being processed. This type of animation is particularly useful in debugging the model. An example of an animated network is shown in Figure 5. Micro Saint also provides an iconic animation capability called ActionView (see Figure 6) that allows users to build a realistic "picture" of their model.



Run 1 of 1 Clock: 502.188
Task Completed: 16 move mail

custype;	5.000
numclerks;	6.000
percent1;	14.516
percent2;	14.706

Figure 5. Micro Saint Symbolic Animation

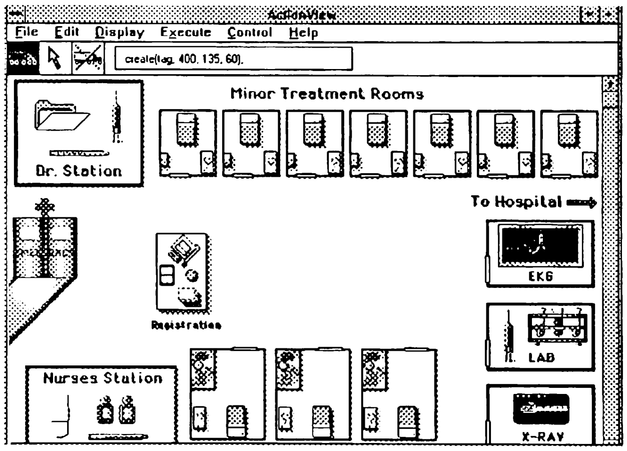


Figure 6. Micro Saint ActionView Animation

In addition, data can be collected at any time during the model run. Sometimes it is sufficient to save the state of the system at the end of the run. However, in order to gain insight into the dynamic aspects of the system users can "take snapshots" of the model variables any time during run. These "snapshots" of data can be analyzed using the graphing capabilities within Micro Saint (see Figure 7) or imported into another statistical analysis package. Through the results of the simulation analysis and the insights gained, users can assess the relative merits of alternative solutions as well as predict their impact which leads to a better understanding of the costs and benefits.

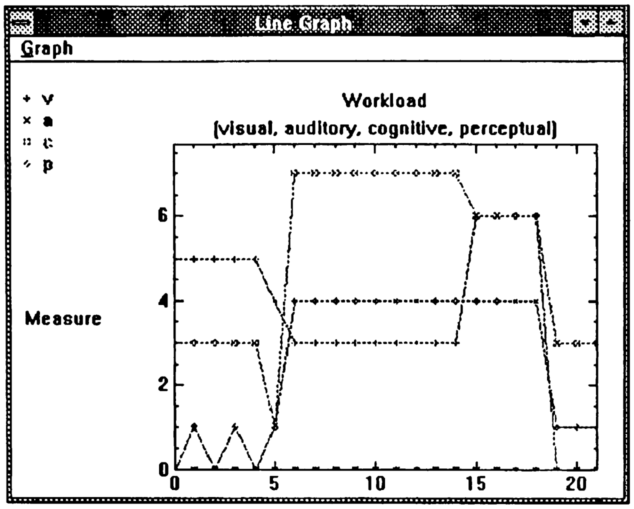


Figure 7. Micro Saint Line Graph

5 SUMMARY

In this paper, we have focused on the Micro Saint methodology and the underlying principles of modeling with Micro Saint; we have not attempted to cover all of the software features. Micro Saint is a powerful tool for evaluating the dynamic issues of systems within a wide variety of application areas. It's primary strength is that it's intuitive, graphical interface allow users to quickly develop models that accurately represent their system. Users are then able to play "what if" with a variety of inputs to evaluate alternative solutions. Simulation technology has been used by Industrial engineers in a manufacturing context since the 1960's but only recently has been applied to more "white collar" applications. Regardless of the application area, the results are the same: better decisions can be made, money can be saved, productivity can be increased, and customers can be better served.

---

[1] A test for logical equality is represented as: "==" in Micro Saint

[2]Assignment statements are represented as: ":=" in Micro Saint