N88-16403

LISP Based Simulation Generators for Modeling Complex Space Processes

Fan T. Tseng
Bernard J. Schroer
Wen-Shing Dwan
University of Alabama in Huntsville
Huntsville, AL 35899
(205) 895-6510

ABSTRACT

This paper presents the development of a simulation assistant for modeling discrete event processes. Included in this paper are an overview of the system, a description of the simulation generators, and a sample process generated using the simulation assistant.

INTRODUCTION

Numerous simulation languages exist for modeling discrete event processes and have now been ported to microcomputers (Pegden 1985 and Minuteman 1986). Graphic and animation capabilities have been added to many of these languages to assist the users build models and evaluate the simulation results.

However, with all these languages and added features, the user is still plagued with learning the simulation language which can be rather time consuming, especially if a complex system is being modeled. Furthermore, the time to construct and then to validate the simulation model is always greater than originally anticipated.

One approach to minimize the time requirement is to use pre-defined macros that describe various common processes or operations in a system. The literature recently has contained several approaches to developing and using these macros or simulation generators. For example, a simulation generator has been developed that translates the characteristics of a Flexible Manufacturing System (FMS) into a simulation model using SIMAN (Haddock 1987).

Another example is a ruled based expert system that assists the modeler in constructing simulation models (Khoshnevis 1986). The expert system automatically generates the corresponding SLAM simulation code. The icons used in this system are very similar to those in GPSS and SIMAN.

A Natural Language Interface (NLI) for an electronics assembly domain has also been developed that automatically generates SIMAN code from user defined text (Ford 1986). This NLI uses a Symbolics 3670 processor and presently has a limited dictionary.

A set of simulation generators has also been developed for simulating manufacturing systems (Schroer 1987). These generators are written in GPSS/PC and can be linked together through a GPSS main program.

SYSTEM OVERVIEW

Several simulation generators have been developed for modeling manufacturing processes (Tseng 1987). These simulators have greatly expedited the modeling process. The next step is to develop a more user friendly interface, or simulation assistant, between the modeler and the actual simulation language. Such a simulation assistant should reduce the user's need to know the details of the simulation language and reduce the time to construct and validate the model. The end result should be increased user productivity.

Figure 1 outlines a system schematic for the simulation assistant currently under development. The user sits at a Symbolics 3620 processor and, based on the prompts from the simulation assistant, defines the process or the model. The simulation assistant is written in Symbolics' Common Lisp.

Once the user has defined the process to be simulated, the simulation assistant automatically generates the corresponding GPSS simulation code of the process. The simulation assistant also calls on the library of simulation generators in generating the GPSS code.

The Symbolics 3620 is interfaced to a 3670 file server which is interfaced, via Ethernet, to a VAX 785. Resident of the VAX is the GPSS simulation system. The user then inputs the necessary run statements, the model is executed and the results printed. Currently, the VAX interface is not operational. Instead, the GPSS code that is generated on the Symbolics 3620 is re-entered into an IBM AT which has resident the GPSS system. The user then inputs the run commands on the AT to execute the model.

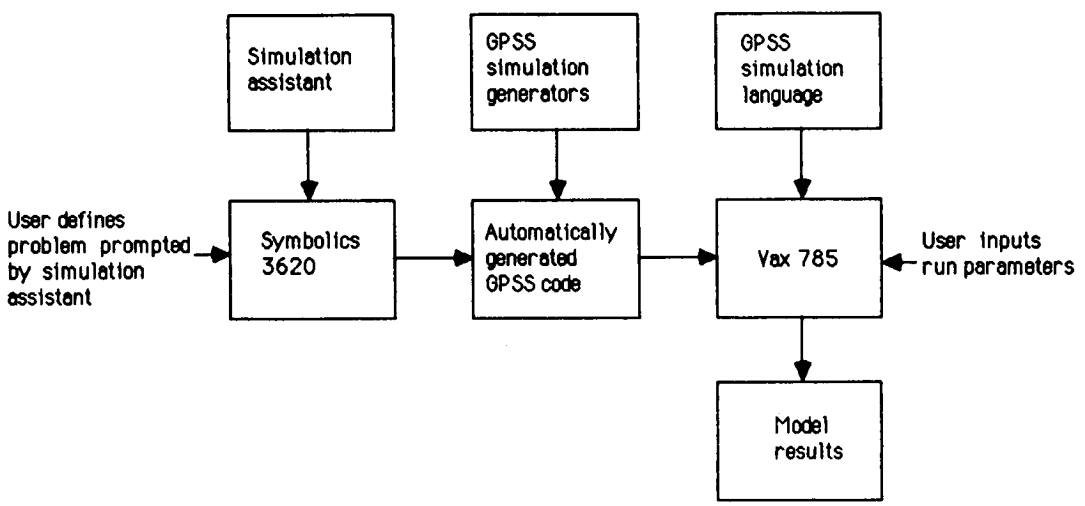

Figure 1. System overview

SIMULATION GENERATORS

Three simulation generators have been developed: an assembly station generator, a manufacturing cell generator, and an inventory transfer generator. Each generator consists of a GPSS macro or subroutine (Tseng 1987).

Figure 2 is a typical assembly station. The generator operates as follows. Items wait in a queue until the station becomes available. The item then seizes the station and waits until item A is available at stock point A. An amount of time is then simulated while item A is assembled to item B resulting in item C. If stock point A is empty, a signal is sent to the inventory transfer generator to move the empty cart to the manufacturing cell. The manufacturing cell then makes the required items for the cart. Another signal is sent to the inventory transfer generator to return the full cart back to the corresponding assembly station. This method of inventory control is commonly called the pull mode. If the assembly station is operated in the push inventory control mode, the items at stock point A are replenished at predefined intervals.

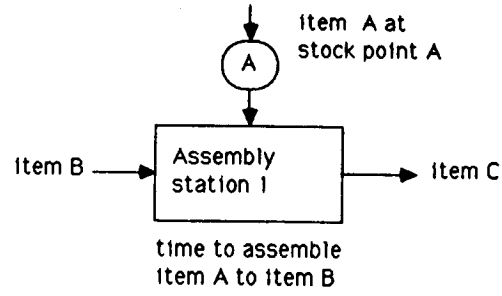

Figure 2. Typical assembly station

Figure 3 is a typical manufacturing cell. The generator operates as follows. Item D, or raw material D, is used to manufacture item E. Likewise, item F is used to make item G. An amount of time is then simulated to make an item. The finished items are stored in stock points E and G. Each stock point contains a defined inventory or carts with a fixed cart capacity. This output can go to either another manufacturing cell or an assembly station. The inventory at the stock points D and F can be raw material or items from another manufacturing cell.

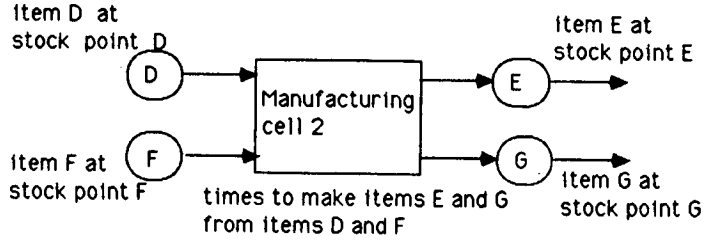

Figure 3. Typical manufacturing cell

SIMULATION ASSISTANT

An example of a typical process, which could translate to a space manufacturing task, is given in Figure 4 and consists of two assembly stations and one manufacturing cell. The following constraints are placed on the process: one inventory transfer generator for all item movement; a pull inventory control is imposed for items B and D; and a push inventory control is imposed for items A and C.

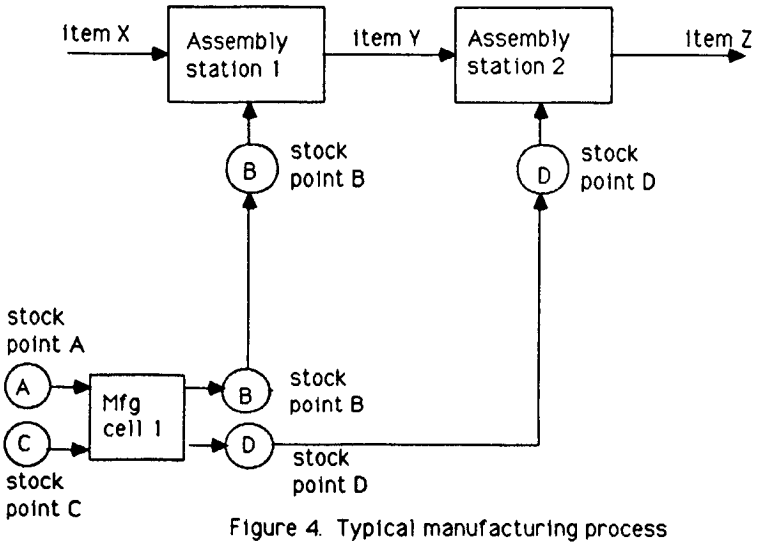

Figure 4. Typical manufacturing process

Figure 5 is a listing from the Symbolics 3620 screen showing a portion of the prompts from the simulation assistant and the corresponding user response. The screen consists of two windows. The right window contains various subwindows and options that the user selects via the mouse. Once the selection is made, the left window prompts the user for specific data that are used to construct a model of the process and to provide data for the simulation generators.

Several of the various subwindows are given in Figure 6. The first subwindow allows the user to select the type of distribution for the assembly time at a station. The current options are either exponential or normal. After the user selects the appropriate distribution, the user enters the mean time and standard deviation in the left window. The second subwindow allows the user to select the method of inventory, either push or pull. Once the method is selected, the user enters the amount of inventory at the various stock points.

CONCLUSIONS

In summary, a simulation assistant is an effective method of rapidly developing discrete event simulation models with minimum knowledge of the simulation language. An expanded set of simulation generators are necessary for modeling more complex processes. However, it is difficult to define these general purpose generators and to then link these generators to the simulation assistant. Considerable more work is required to make the simulation assistant more robust.

ORIGINAL PAGE IS
OF POOR QUALITY

```
Command: (sample-process)
Name of line: MAIN
Number of stations: 2
Mean arrival time: 100
Name of inventory part required at station 1: B
Mean assembly time: 100
Name of inventory part required at station 2: D
Mean assembly time: 100
Any more lines to create? (Y or N) No.
Maximum part B at assembly line [default 100]: 2
Maximum cart B at assembly line [default 100]: 2
Maximum part B at manufacturing cell [default 100]: 2
Maximum cart B at manufacturing cell [default 100]: 2
Mean manufacturing time: 10
Name of inventory transfer: FGIG1
Mean inventory transfer time: 2
Name of manufacturing cell: CELL1
Number of types of input items: 2
Name of input item: A
Name of input item: C
Maximum part D at assembly line [default 100]: 2
Maximum cart D at assembly line [default 100]: 2
Maximum part D at manufacturing cell [default 100]: 2
Maximum cart D at manufacturing cell [default 100]: 2
Mean manufacturing time: 10
Name of inventory transfer: FGIG1
Mean inventory transfer time: 2
Name of manufacturing cell: CELL1
Number of types of input items: 2
Name of input item: A
Name of input item: C
Maximum part A capacity at assembly line: [default 5000]: 100
Maximum part C capacity at assembly line: [default 5000]: 100
```

Figure 5. Simulation assistant display

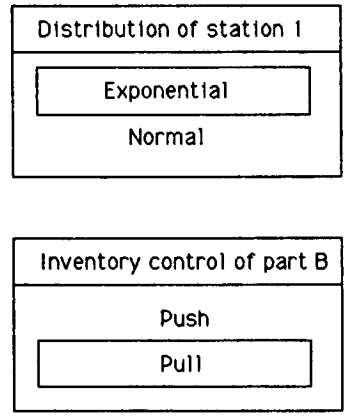

Figure 6. Various subwindows

REFERENCES

Ford, D. R. and B. J. Schroer. 1987. "An Expert Manufacturing Simulation System." Simulation, vol 48, no. 5.

Haddock, J. and Robert P. Davis. 1985. "Building a Simulation Generator for Manufacturing Cell Design and Control." Proceedings 1985 IIE Spring Conference, Los Angeles, CA.

Khoshnevis, B. and A. P. Chen. 1986. "An Expert Simulation Model Builder." Intelligent Simulation Environments, Society for Computer Simulation, vol 17, no. 1.

Mathewson, S. C. 1984. "The Application of Program Generator Software and Its extensions to Discrete Event Simulation Modeling." IIE Transactions, vol 16, no. 1.

Pegden, C. Dennis. 1985. Introduction to SIMAN. Systems Modeling Corp.

Schroer, B. J. and F. T. Tseng. 1987. "Modeling Complex Manufacturing Systems Using Simulation." Proceedings 1987 Winter Simulation Conference, December 1987, Atlanta, GA.

Tseng, F. T. and B. J. Schroer. 1987. "Use of Simulation Generators in Modeling Manufacturing Systems." Proceedings Southeastern Simulation Conference SESC 87, Society for Computer Simulation, October 1987, Huntsville, AL.