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

DOMAIN-BASED ON-LINE SIMULATION FOR REAL-TIME DECISION SUPPORT

Murali Krishnamurthi
Suresh Vasudevan

Department of Industrial Engineering
Northern Illinois University
DeKalb, Illinois 60115, U.S.A.

ABSTRACT

In this research the applicability of on-line simulation systems for real time decision support is explored and the concept of domain based on-line simulation systems is introduced. For the purpose of demonstrating the feasibility of this concept, a prototype domain based on-line simulation has been designed, developed, and implemented. The details of the prototype system and how it could be used to make real-time decisions for various problem situations in a chosen domain are discussed. The implemented on-line simulation system has been validated using an off-line simulation model and the results have been analyzed to evaluate the feasibility and cost effectiveness of developing domain based on-line simulation systems.

1 INTRODUCTION

1.1 Background and Motivation

Traditionally simulation has been used as an effective tool for off-line decision making. However, one of the limitations of the traditional simulation approach is the considerable amount of time spent in collecting and analyzing the data input to the simulation or output from the simulation experiment. This has often forced decision makers to rely on simulation primarily for off-line decision making and not for critical decision making situations that may arise on-line or in real-time.

The application horizon of simulation could be expanded by focusing on the development of on-line simulation systems that acquire data, run simulation experiments and deliver the decision in a specified time window without direct human input. This will lead to the use of simulation in real-time decisions such as flight and crew scheduling in airline industries where various delays force the airline industries to reschedule their flights and crew.

The primary issues concerning the development of on-line simulation systems are real-time data acquisition, and on-line input and output analysis. Real-time data could be acquired by developing simulation models to monitor the real-life system status continuously so that the data is available at any time when a simulation run is to be made or by developing data interfaces to acquire the data continuously from the real-life system thereby eliminating the need for separate data collection. The input analysis and the output analysis could be performed by the use of expert systems and knowledge bases developed for a particular domain. In this research effort, these aspects of on-line simulation are explored in order to develop on-line simulation as a tool for real-time decision support.

1.2 Research Objectives

The concept of domain based on-line simulation systems is explored in this research for the purpose of using them as general purpose decision support systems. The feasibility of this concept is demonstrated by designing, developing, and implementing a prototype on-line simulation system for a selected domain and showing that it can be customized to make real-time decisions in that domain. The prototype system has been developed using Turbo C, CLIPS, and SIMAN and has been implemented on microcomputers. The implemented prototype system has been validated with off-line simulation models to compare and analyze the results.

The objectives of this research are as follows:

1. To explore the concept of domain based on-line simulation system as a tool for real-time decision making.
2. To conceptualize a generalized framework for domain based on-line simulation systems.

3. To demonstrate the feasibility of developing domain based on-line simulation systems that could be used for making real-time decisions for various problem situations in the chosen domain.

2 PAST RESEARCH ON ON-LINE SIMULATION SYSTEMS

The major components of on-line simulation systems discussed in the existing literature generally consist of a data acquisition module, a simulation model and a control program. Additional features such as expert systems and knowledge bases are also included in some on-line simulation systems. Past research in on-line simulation has focused mainly on production scheduling and monitoring and control and they are briefly discussed in the following sections.

2.1 On-Line Simulation for Real-Time Scheduling

A major portion of the past research has been on on-line real-time scheduling of manufacturing systems. Rogers and Flanagan (1991) have developed a framework for an on-line simulation system for real-time scheduling. The on-line simulation system gets shop floor status, material plan and planning options and evaluates the performance of these options. The output is analyzed by production control personnel or an expert system to come up with the best alternative. This framework is applicable for real-time scheduling in manufacturing industries.

Jain et al. (1989) used an on-line simulation system in the development of an Expert System Scheduler. The framework used in this system consists of an Expert System Scheduler and a Factory Control System. The scheduling computer acquires shop floor information from the factory control computer, generates a new schedule based on simulation runs and sends it back to the factory control computer. The factory control computer interacts with the shop floor, materials department and the marketing department to get the necessary input and to send the new schedule. The simulation models have been developed using LISP and they use backward chaining concept to go from the desired output to the necessary input.

On-line simulation was used in a work order release mechanism for a flexible manufacturing system developed by Muller et al. (1990). The framework of this system consists of a simulation model and a control system interface. The simulation model acquires necessary data directly from the databases. The model is run by the control system interface for a specified time window to find the effect of various order release policies. The analyst selects the best schedule based on simulation results. The simulation model was written in SIMAN and the data interfaces were developed in FORTRAN.

2.2 On-Line Simulation for Monitoring and Control

On-line control of a manufacturing cell was achieved by Manivannan and Banks (1991). The framework of this system contains knowledge bases, databases, simulation models and control interfaces. The model monitors the cell continuously and the user initiates the system emulation any time a control decision is needed. A knowledge base is used to selectively simulate alternatives for which prior knowledge does not exist. The simulation models are written in SIMAN and the knowledge bases are written in LISP. Based on this system, Manivannan and Banks have come up with a design for a knowledge-based on-line simulation system to control a manufacturing shop floor.

Manivannan and Banks (1990) have also developed an on-line knowledge based simulation system for diagnosing machine tool failure. The framework of this system is similar to the previous system. The data from sensory devices is analyzed by a controller and the simulation model is used to calculate the time of failure of machine tool whenever an impending failure is sensed. A knowledge base stores the results which are used for eliminating simulation runs if prior knowledge exists. On-line simulation systems have also been used in other applications such as signalized intersection control for evaluating various signal control strategies by Chang (1989), and in developing efficient and flexible operations and training of air traffic control trainees by Kornecki et al. (1991).

2.3 Need For This Research

The above mentioned applications involve simulation models that monitor the real-life system which are interrupted to make decision runs whenever control decisions are needed. The architectures and the implementations of these systems are based on the specific problem situations for which they were developed. This limits their application in real-time decision making. The cost of building an on-line simulation system just for a single problem situation affects the economic feasibility of using on-line simulation for real-time decision making. Jain et al. (1989) have concluded that developing an expert system scheduler is very expensive due to the expertise necessary in developing such a system. This limitation

could be overcome by a generalized approach towards the real-time decision making problems. The framework developed by Rogers and Flanagan and that developed by Manivannan and Banks are in this direction. However, the architectures of the on-line simulation systems have to be based on a group of problem situations, rather than for individual problems. This will improve the cost effectiveness of on-line simulation systems. It is therefore necessary to prove the feasibility of developing such systems so that future on-line simulation systems could be generalized.

3 A FRAMEWORK FOR DOMAIN BASED ON-LINE SIMULATION

The concept of domain based on-line simulation systems is based on the concept of simulators. A simulator is a simulation program that can used for simulating a group of problems with minimal customization. The simulation model is generated by the program based on the customization. The group of problems constitute a domain; for example, job shop scheduling problems in general could constitute a domain. A domain based on-line simulation system is a simulation program that can be customized to various specific problem situations in the domain. Before conceptualizing and implementing a domain based on-line simulation system, its needs, requirements, and related issues must be addressed.

3.1 Needs and Requirements

The issues involved in implementing on-line simulation systems for scheduling have been addressed extensively by Harmonosky (1990) and Rogers and Flanagan 1991). The following sections discuss some of the issues related to the conceptualization and implementation of a domain based on-line simulation system.

3.1.1 Real-Time Data Acquisition

For an on-line simulation system, the acquisition of accurate data in real time that truly reflects the current status of the system is essential. This can be accomplished by using sensors interfaced to physical systems and automatic data acquisition systems. In a domain based on-line simulation system, it is necessary to customize the data acquisition module based on the parameters that characterize a particular problem situation.

3.1.2 Multiple Simulation Runs

Most simulation models characterize stochastic problem situations, and therefore, it is generally necessary to make multiple runs to get acceptable confidence in the output measures. This may consume a considerable amount of time and so the meaning of real-time decision may be lost. However, real-time decisions are needed only in cases where the variations in the system parameters prevent the system from reaching a steady state behavior. Hence, in situations where the time window for simulation is short so that the system may not achieve steady state, it may be possible to use a single run to arrive at a rough cut solution using on-line simulation. A domain based on-line simulation system should have the flexibility to customize the simulation run based on user input.

3.1.3 Decision Alternatives to be Evaluated

A decision making situation can result in a number of alternative solutions and it may or may be possible to simulate all of them within a given time window. Further, it may not be necessary to simulate all possible situations every time, since information may exist for certain cases from prior simulations. It is, therefore, possible to restrict the number of alternatives to be simulated and evaluated using other means such as expert systems or knowledge bases which can maintain information on prior simulations. In the case of a domain based on-line simulation system, it is essential to include the means to restrict the alternatives to be evaluated based on prior knowledge.

3.1.4 Simulation Models for Different Alternatives

Evaluating each alternative for a problem situation generally requires changes in the simulation model. This may necessitate recompilation of the simulation model thus consuming a substantial amount of time. To avoid this recompilation, it may be possible to use a separate model for each alternative. Even though it will add to the memory space overhead, it will speed up the decision making process. Alternately, it may be possible to have a single simulation model that can be configured for each alternative by the use of configuration files. A domain based system should be capable of handling automatic simulation configuration without any user initiated recompilation.

3.1.5 Speed of Simulation Run

Current microprocessors are quite fast and therefore simulation runs can be made reasonably fast. If needed,

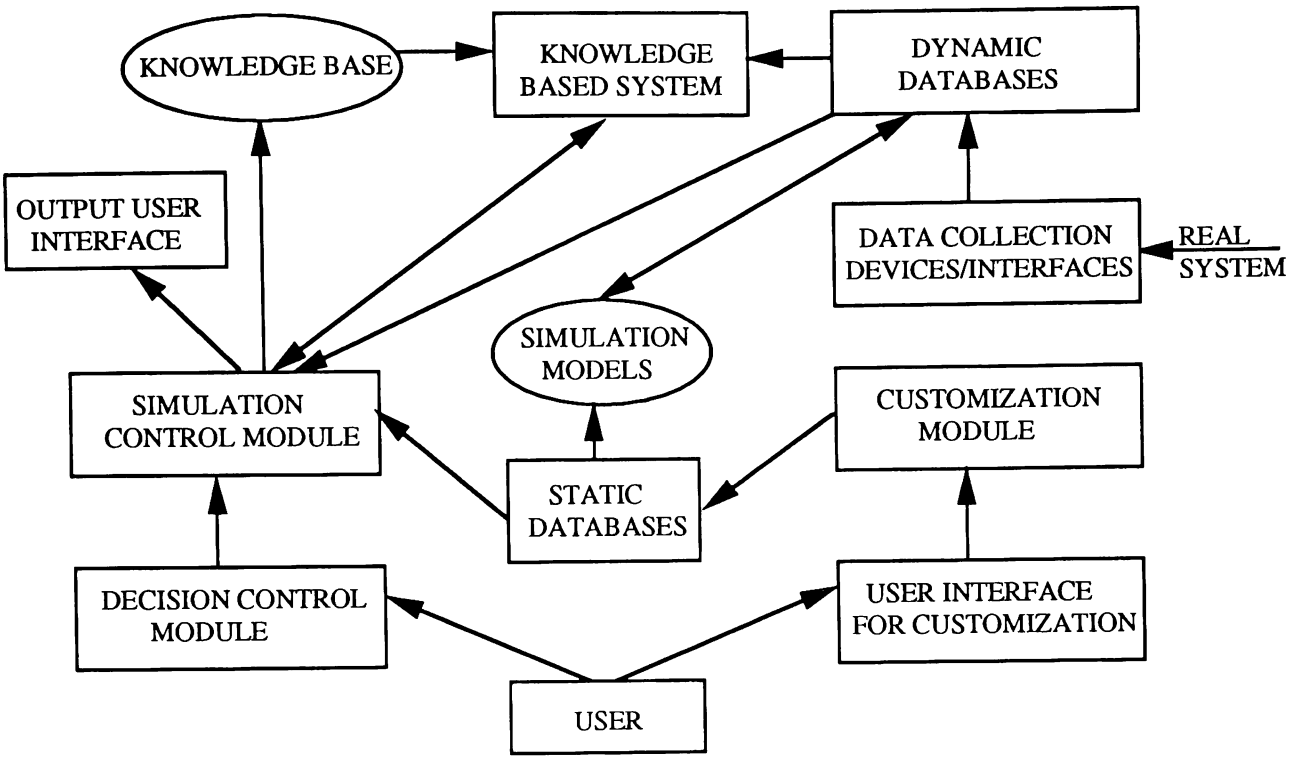

Figure 1: Framework of a Domain Based On-Line Simulation System

workstations may be used to speed up the processing. Distributed simulation using parallel processors may also be used to speed up the simulation runs.

3.1.6 User Interfaces

Domain based on-line simulation systems need user input for customizing to a particular problem situation. This requires user friendly interfaces that can prompt for the necessary input along with all the available options to ensure smooth user interaction. It is also necessary to include output user interfaces to convey the decision to the user.

3.1.7 Interrupt Capabilities

In certain situations, it may be necessary to stop the on-line simulation system and restart the decision making process with new input parameters. This requires interruption capabilities that have to be built into the control module of the on-line simulation system. In cases where simulation replications consume a considerable amount of time, it may be necessary to include the interruption capability within the simulation model itself. Based on the needs and requirements analyzed in the previous section, a framework for a domain based on-line simulation system has been conceptualized.

3.2 Conceptualization of a Framework

The essential features for a domain based on-line simulation system identified in the previous section can be captured in a generalized framework through the following modules: a simulation module, a customization module, a data acquisition module, a simulation control module, and static and dynamic databases. Additional features such as expert systems and knowledge bases can be included depending on the domain requirements. The static databases can be used to store the parameters of the problem situation and the dynamic databases can be used to store the input data from the data acquisition module. Figure 1 shows the generalized framework for a domain based on-line simulation system that can be used for real-time decision making.

The *customization module* will acquire the parameters of a problem situation from a decision analyst through an user interface, and update the static databases. It will also initialize the knowledge base. The *data acquisition module* will collect the necessary system data, as specified in the static databases, continuously from a real-world system and update the dynamic databases. The *decision control module*, invoked by the user, will initiate the decision making process by invoking the *simulation control module*. The simulation

control module will call the knowledge based system to check the knowledge base for prior decisions for the given input parameters. If a prior decision exists, the decision will be conveyed through the output user interface. In the case of no prior decisions, a simulation run will be initiated for the first decision alternative. Simulation models will utilize the data stored in the static as well as the dynamic databases to emulate the system status for a specified time window, as specified in the static databases, into the future. The output from the simulation run will be analyzed by the simulation control module. This process of running the simulation models and analyzing the output will continue for all the alternatives or till the performance objective is achieved. The decision made by the simulation control module will be stored in the knowledge base as well as conveyed through the output user interface.

The mentioned framework is used in the domain based on-line simulation system developed in this research. However, the framework for a specific domain based on-line simulation system could vary slightly from the given framework depending on the domain requirements. The domain selected for this research and the design and development methodology are described in the next section.

4 SYSTEM DESIGN AND DEVELOPMENT

The first step necessary in the design and development of a domain based on-line simulation system is the selection of a domain. For this research, a queuing system with single stage service, parallel identical servers and a single queue has been chosen. This domain consists of a variety of problem situations characterized by the number of servers, the arrival pattern and the service distribution. Some real-world examples of this type of queuing system could be: airline check-in counters at airports, service counters in post offices, ticket counters at bus stations, and registration counters in universities, etc.

In the chosen queuing system, customers arrive at a service counter, wait for a server, get the server, perform the necessary transaction and then leave the system releasing the server for the next customer. The real-time decision needed for this type of system is the number of servers required to satisfy a performance objective such as not exceeding a specified queue length at any point of time or not making anyone wait for more than a specified length of time. The decision should be based on the simulation of the system for specified time window into the future. The on-line input parameters will be the current queue length and the arrival rate in the time window specified. A specific problem in this domain will have a given maximum number of servers available, an arrival distribution, a service distribution, a specified time window and a performance objective.

4.1 Design Features of the System

The various aspects involved in the design of a domain based on-line simulation system for the chosen domain are described in the following subsections.

4.1.1 Data Acquisition/Decision Output

In practical situations, the on-line input parameters could be acquired directly from a system through sensors, factory control computers or on-line data bases. However, in this research effort on-line data acquisition is emulated using two personal computers as shown in Figure 2. The data is acquired by a user end computer and is transferred to the decision support computer containing the on-line simulation system through a serial interface. The input data comprises of the current queue length and parameters for the arrival distribution. The decision from the on-line simulation system is transferred back to the user end computer and is displayed through a user interface. The decision produced by the system is the number of servers required for the specified time frame to satisfy the performance objective.

The programming modules for sending and receiving the data were developed using low level functions in C language and the programs were compiled using Turbo C software (1988). The sending program reads the real-time data file, stores the information as characters in an array and signals the receiving program in the other computer. Once the receiving program signals back when it is ready, the sending program transmits the ASCII values of the characters stored in the array. The receiving program receives the values and stores them in an array. Upon the completion of transmission, the receiving program writes the ASCII values to a data file in the receiving computer. These programming modules were tested using a sample data files to make sure that the data was being transferred accurately. The modified versions of these programs were used for transfer of arrival distribution and decision from the decision support computer to the user end computer.

4.1.2 Customization

The customization module is basically an user interface with dialog boxes for user input to customize the

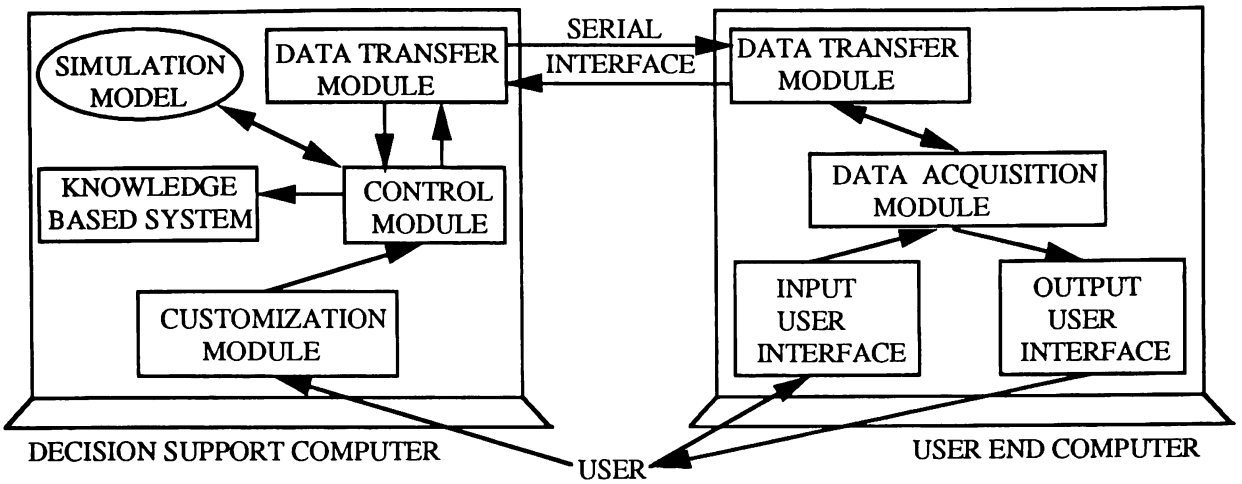

Figure 2: Data Transmission and Acquisition Between Computers

domain based on-line simulation system. This module was developed using C language and existing graphic utility programs developed by Stevens (1989) and was compiled in Turbo C. This module prompts the user for the set up parameters. Wherever options are limited, the options were displayed so that the user could select one of them. The customization module writes all these information to static data files. It also transfers the information about arrival distribution to the user end computer to configure the user interface for input in the user end computer. On-line help for any of the parameters could be accessed by pressing the F1 key. In the domain based on-line simulation system developed for this research, the system can be customized through user input for the following parameters:

1. Maximum Number of Servers Available
2. Arrival Distribution (5 options)
3. Service Distribution (5 options)
4. Time values for Service (based on selected Service Distribution)
5. Performance Criterion (4 options)
6. Performance Value

The arrival and service distributions handled by the simulation model and the performance objectives that can be handled by this system are shown in Table 1. The decision to limit the options for arrival distribution, service distribution and performance objective is to restrict the scope of this research to demonstration of developing domain based on-line simulation systems. The system developed can easily be expanded to include more options for these parameters.

4.1.3 Simulation Model

The simulation model is capable of reading the configuration and data files to simulate the customized problem situation for the length of time specified and writing the output values of the performance parameters to output files. The simulation model is built with the arrival distributions, service distributions and the performance objectives shown in the Table 1.

The simulation model for a single stage, multiple server queuing system with a single queue was developed using SIMAN. The parameters input to the simulation module and the parameters output from the simulation module are shown in Figure 3. The model reads the input parameters from input data files to simulate the system. The output parameters are written to the output data files. The model was tested with various input values to ensure proper operation.

4.1.4 Control Module

The control module is a C program that was compiled using Turbo C compiler. The control module calls the receiving program to receive the input data from user end computer, runs the simulation model with different alternatives (i.e. in this case, with increasing number of servers) till the alternative that satisfies the performance objective is found, writes the input values and decision to a file, referred to as knowledge base and then calls the sending program to transfer the decision to the user end computer. The control module was tested using a specific problem situation.

Table 1: Arrival and Service Distributions Handled By the System and the Choice of Performance Objectives

ARRIVAL DISTRIBUTION	SERVICE DISTRIBUTION	PERFORMANCE OBJECTIVES
Uniform Distribution	Uniform Distribution	Maximum Queue Length
Traiangular Distribution	Traiangular Distribution	Average Queue Length
Normal Distribution	Normal Distribution	Maximum Queue Length
Exponential Distribution	Exponential Distribution	Average Waiting Time
Poisson Distribution	Weibull Distribution	Maximum Time in System
		Average Time in System

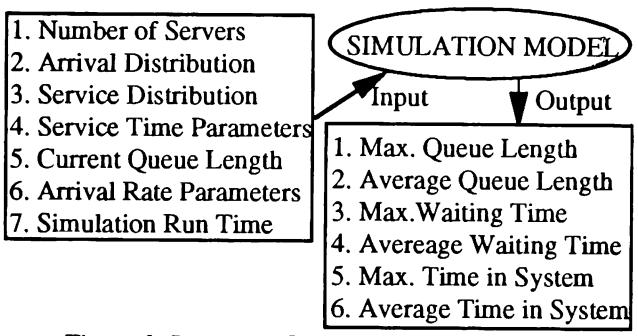

Figure 3: Input and Output Parameters of the Simulation Model

4.1.5 Knowledge Based System

The knowledge based system was developed using rules in CLIPS (1991) expert system environment. This system reads the input values and compares it with that in the knowledge base to check for prior decisions. If prior decision exists, then it writes the decision to a file; otherwise, it writes a failure code to a file. The knowledge based system was tested and then the control module was modified to include the call to knowledge based system before starting the simulation runs. If the knowledge based system finds a decision from the knowledge base, control module will just transfer it to the user end computer; otherwise, it proceeds as described in the previous subsection.

4.1.6 User Interfaces

Two user interfaces were developed and installed in the user end computer. The interface for input is based on the arrival distribution selected in customization. The input interface is graphical and it prompts the user for the current queue length and the arrival parameters based on the distribution. It then writes this information to a file and calls the sending program to transfer the values to the decision support computer. The input user interface contains on-line help for the input parameters which can be accessed by pressing the F1 key. The output interface receives the decision from the decision support computer and displays to the user end screen using a graphic window.

4.2 Operational Algorithm of the System

The first step in the operation of the system is the customization of the system to a particular problem situation characterized by maximum available servers, arrival distribution, service distribution with service time parameters, simulation run length, decision criterion and the objective value. Once customized and evoked, the system operates based on the following algorithm:

Step 1. [input] Prompt and receive current queue length and arrival parameters from the user.
Step 2. [prior knowledge?] Check the knowledge base for prior decisions for the given input. If prior decision exists, go to Step 8.
Step 3. [initialize] Set current number of servers = 1.
Step 4. [simulation] Run simulation with current number of servers and current queue length.
Step 5. [analyze] Compare the simulation output with specified objective value. If objective is satisfied, set decision value = current number of servers and then go to Step 8.
Step 6. [any more servers?] If current number of servers equals maximum available servers, go to Step 9.
Step 7. [increase servers] Set current number of servers = current number of servers + 1 and then return to Step 4.
Step 8. [decision] Send the decision value and then return to Step 1.
Step 9. [servers not enough] Send message that servers are insufficient; return to Step 1.

Table 2: Results From the On-Line and Off-Line Simulations

S. NO.	QUEUE LENGTH	MEAN ARRIVAL	REQUIRED SERVERS	
			ON-LINE	OFF-LINE
1	3	1.0	8	8
2	4	1.5	7	7
3	5	2.0	5	5
4	6	2.5	4	4
5	7	3.0	5	5

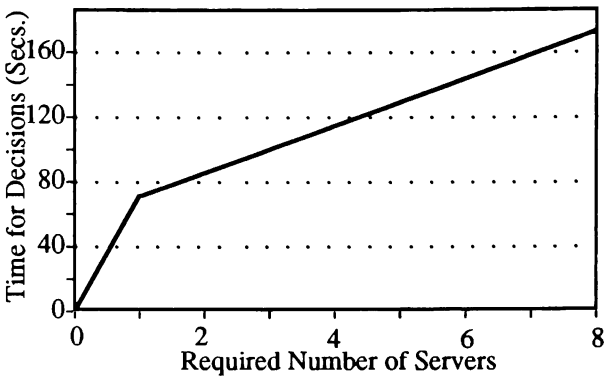

Figure 4: Decision Time Vs. Number of Servers

5 SYSTEM VALIDATION AND ANALYSIS

5.1 Validation of the System

The system was validated using a specific problem in the selected domain of the single stage system with multiple servers and single queue. The problem situation is based on the performance objective of a pizza delivery outlet with hypothetical service and arrival rate parameters as follows:

1. Decision needed: *Number of delivery persons*
2. Maximum available servers: *8 (drivers)*
3. Arrival Distribution: *Poisson (arrival of telephone orders)*
4. Service Distribution: *Triangular(10.0,12.0,14.0)*
5. Performance Objective: *Maximum time from order to delivery within 30 min.*
6. Simulation Time Window: *60 minutes.*
7. Input Parameters: *Mean arrival and current pending orders*

In order to validate the system, an off-line simulation model was developed for the above problem along with a control module for input and output. Five sets of input parameters, i.e. five sets of current queue length and mean poisson arrival rate, were used to run both the developed system and the off-line model. For the off-line model, the simulation was run with an increasing number of servers and the simulation results were analyzed till performance objective was achieved. The decisions produced by the developed system and the decisions made by the user using the off-line model are shown in Table 2. These results ensure the validity of the developed system.

5.2 Analysis of the System

The developed system and the off-line simulation were analyzed to find the amount of time saved by the on-line simulation system. The time for simulation runs is the same in both the cases; however, the on-line simulation eliminates the simulation runs for cases where prior knowledge is available. This results in a considerable savings in time which is otherwise spent on running the simulation models. The time for data entry and transfer is a constant for the developed system whereas the off-line model requires the user to input the parameters for each and every run. Hence, off-line simulation takes a longer time than the developed system. Further, the possibility of human error in analyzing the simulation results is eliminated by the developed system.

Figure 4 shows a plot of time for decision in the on-line simulation system versus the number of servers required. It can be seen from this plot that after the initial time required for customizing the system, the time for decision is a factor of the number of servers. The amount of time taken by the developed system, for the specific problem used for validation, was between 58 to 72 seconds. The amount of time taken by the off-line process, for the same problem, was between 68 to 122 seconds. On the average, the time saved by the developed system worked out to nearly 30% out of the time taken by the off-line process. The one-time cost of automating the data retrieval will be easily offset by the savings in cost of professional labor needed for simulation analysis. Cost savings could be analyzed by comparing the cost of automating the data retrieval and the overhead cost of the domain based on-line simulation system with the cost of developing an off-line simulation model and the cost of employing a simulation analyst. A detailed cost analysis is beyond the scope of this research since the implemented system is a prototype and it also emulates real time data acquisition instead of using actual real time data acquisition.

6 CONCLUSIONS AND RECOMMENDATIONS

In this research, the use of on-line simulation for real-time decision making was explored. The issues involved in developing domain based on-line simulation systems were analyzed and a framework that can be used for developing domain based on-line simulation systems was developed. A prototype domain based on-line simulation system was designed and developed for a selected domain. The feasibility of developing domain based on-line simulation systems was demonstrated by validating the developed system with off-line simulation models for the same problem situations used in the on-line simulation runs.

The prototype system developed for this research is applicable for solving a variety of practical simulation problems. However, the prototype system has limited options for the arrival and service distributions and also for the performance objectives. Further, the interruption capability has not been included in this system due to the emulation of automatic data acquisition using two computers. The user end computer waits for the decision from the decision support computer so that interrupting the simulation will affect the data interface between the two computers. The system could be further generalized by including more options and the interruption capability.

REFERENCES

Chang, E. C. 1989. Simulated real-time intersection signal control. *Proceedings of the 1989 Winter Simulation Conference*, 1085-1091.

CLIPS User's Guide. 1991. NASA Software Technology Branch, Georgia.

Haeme, R. A., J. L. Huttinger, and R. W. Shore. 1988. Airline performance modeling to support schedule development: An application case study. *Proceedings of the 1988 Winter Simulation Conference,* 800-805.

Harmonosky, C. H. 1990. Implementation issues using simulation for real-time scheduling, control and monitoring. *Proceedings of the 1990 Winter Simulation Conference*, 595-598.

Jain, S., K. Barber and D. Osterfeld. 1989. Expert simulation for on-line scheduling. *Proceedings of the 1989 Winter Simulation Conference*, 930-935.

Kornecki, A., J. Cieplak and A. Schneider. 1991. Real-time simulation of air traffic control radar terminal with trainer interface. *Simulation*, 381-389.

Manivannan, S. and J. Banks. 1990. Towards a real-time knowledge based simulation system for diagnosing machine tool failure. *Proceedings of the 1990 Winter Simulation Conference*, 603-608.

Manivannan, S. and J. Banks. 1991. Real-time control of a manufacturing cell using knowledge-based simulation. *Proceedings of the 1991 Winter Simulation Conference*, 251-260.

Muller, D. J., J. K. Jackman and C. Fitzwater. 1990. A Simulation-based work order release mechanism for a flexible manufacturing system. *Proceedings of the 1990 Winter Simulation Conference*, 599-602.

Nolan, P. J., G. M. Lane and J. M. Fegan. 1991. ISI - An environment for the engineering use of general purpose simulation languages. *Simulation,* January 1991, 41-47.

Pegden, D. C., R. E. Shannon and R. P. Sadowski. 1990. *Introduction to Simulation using SIMAN.* McGraw-Hill, New Jersey.

Rogers, P. and M. T. Flanagan. 1991. On-line simulation for real-time scheduling of manufacturing systems. *Industrial Engineering*, December 1991, 37-40.

Stevens, A. 1989. *Turbo C: Memory-Resident Utilities, Screen I/O and Programming Techniques*, BPB Publications, New Delhi.

Turbo C User's Guide. 1988. Borland International, California.

AUTHOR BIOGRAPHIES

MURALI KRISHNAMURTHI is an Assistant Professor in the Department of Industrial Engineering at Northern Illinois University. His research interests include simulation, manufacturing systems, AI/ES, operations research and information systems.

SURESH VASUDEVAN is a graduate student in the Department of Industrial Engineering at Northern Illinois University. His research interests include simulation, manufacturing, and decision support systems.