VIRTUAL MACHINES FOR MESSAGE BASED, REAL-TIME AND INTERACTIVE SIMULATION

Hansoo Kim
Chen Zhou

Department of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, GA 30332-0205, U.S.A.

Hua X. Du

Siemens, EAE
3140 Northwoods Pkwy
Norcross, GA 30071, U.S.A.

ABSTRACT

An advanced processing machine interacts with the material handling system, personnel and cell or shop floor controller in real-time via messaging and control. However, current simulation models are normally built with simulation software tools that are not designed to explicitly model machine interactions. In this research, we develop a modular design of simulation tools. One of the fundamental building blocks is the virtual machine module that simulates machine behavior in terms of state change and its interface. The modular design offers the ability to interact with the surroundings via messaging, similar to real machines. The virtual machine can be used to help develop shop floor controllers and machine controllers, and to test different shop floor control strategies interactively. Its control console can be used for operator training as well.

1 INTRODUCTION

Discrete Event Simulation is a method to model and analyze complex queuing networks not possible by analytical methods. It is used widely in modeling manufacturing systems. Simulation models are normally built using simulation software tools. The tools provide random number generation, entity list processing, queue, event and clock handling. To build a simulation model, a modeler specifies the time and logical behavior of entities, statistics to be collected, and stopping criteria of the model. It can be used to analyze very complex manufacturing systems in high fidelity not possible by any analytical means. “It is an indispensable problem-solving methodology for solution of many real-world problems. It can be used to describe and analyze the behavior of a system and ask what-if questions.” (Banks, 1998).

There are limitations in building models with current simulation tools. One of the limitations is related to the inconvenience in explicit modeling of interactions of processing machines with its controller, materials and the operator. Once started, a model is intended to execute until some pre-specified terminating conditions are met. The frequent interaction with running simulation is not intended although possible with some tools (Narayanan et al., 1997).

An advanced processing machine is a computer controlled machinery that interacts with the material handling system, personnel and cell or shop floor controller in real-time via messaging and control from control panel. It is difficult to build a high fidelity model with typical "all-in-one" model between the processing machines and their controllers.

In this research, we propose to develop a modular design of simulation tools. One of the fundamental building blocks in this modular tool is a virtual machine. A virtual machine is a model based software module that simulates the behavior of a machine in terms of state change and its interfaces. The interface will be via messaging, similar to real machines. The separation of machine controller from cell or shop floor controller allows more natural mapping from physical systems to simulation systems and modular development of a system. The modules can be used to test different shop floor control strategies interactively. It allows the development, prototyping and testing of shop floor controllers and machine controllers. The control console feature can also be used for operator training.

2 STATE MODEL OF VIRTUAL MACHINE

The development of generic machine module must strike a balance between simplicity and functionality, between generality and modeling speed.

A machine handles two main types of materials: those directly related to each product, we refer to as raw materials, and those not directly related to individual product, we refer to as supplies. The first type can be a wafer, a stock for machining or a circuit board. The second type can be the chemicals, cutting tools, solders or IC chips. The time and logical behavior of the two types of materials are distinctive but not easily modeled by common simulation tools (Davis, Brook and Lee, 1997). In a rough partition, a machine can be in one of the three states: Down, Setup and Ready. In the Down state, a machine is not able to perform its intended process without some setup. In the Setup state, a machine is being prepared or repaired for service. In the Ready state, a machine may be either idle or processing.

Down state can be further partitioned into errors or simply run out of supplies. An example for the later can be a copier. If a copier runs out of supply, we post a sign "DOWN" on it. We do not normally consider it being idle.

In the Setup state, it is important to distinguish "who" is setting up the machine from messaging point of view, whether some personnel or the shop floor controller.

For a machine to process, it must be set up properly, free of error and have both types of materials.

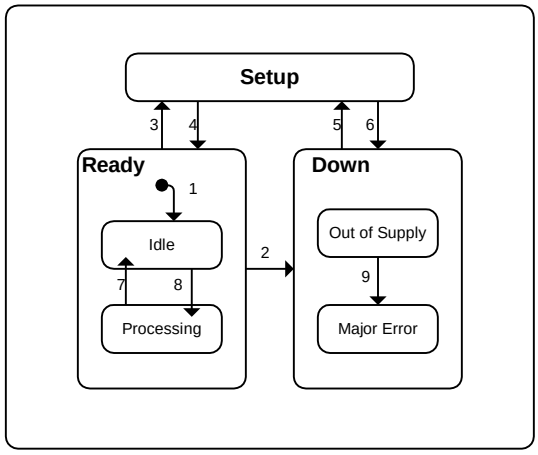

Figure 1: State model of virtual machine

Table 1: State transition table for the state model

#	Current State	Trigger	New State
1	Ready	(Default entry of state Ready)	Idle
2	Ready	Upon severe errors, or run out of supplies	Down
3	Ready	Start regular changeovers	Setup
4	Setup	Complete Setup, Repair, or Maintenance	Ready
5	Down	Start error correction procedure	Setup
6	Setup	Upon sever error	Down
7	Processing	Material out or Upon command of stop or abort	Idle
8	Idle	Material in, with the authorized command of start	Processing
9	Out Of Supply	Upon severe errors	Major Error

The differences between different processing machines are in the state of Processing. This state can be further partitioned for modeling convenience.

The definition for the state model in this study is shown in Figure1 and Table 1.

3 DEVELOPMENT OF THE VIRTUAL MACHINE

A virtual machine can be implemented in many different ways. A server side application that communicates via HTTP protocol using XML messaging provides best interoperability and ease of access. Specifically, the virtual machine is implemented as a set of servlets running within a Java Web Server (JWS). It can communicate with shop floor controller module, other virtual machines and operator over the Internet with browser interface.

A servlet is a small Java application that runs on the server side to extend and enhance the web servers. Servlets provide a component-based, platform-independent method for building interactive web applications (Patzer, A. et al., 1999).

JWS supports the generation of dynamic web sites that are extensible, easy to administer, and secure. Currently, it is a popular platform for servlet development, which provides an easy way to make dynamic web applications.

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems. It is a request/response protocol, and has been in use by the World-Wide Web global information initiative since 1990. The Extensible Markup Language (XML) is the universal format for the structured documents and data on the web, and is used to express HTTP requests and responses.

With servlet, JWS, and HTTP/XML protocol, the virtual machine can be easily interfaced with platform-independent clients. Figure 2 illustrates the architecture of virtual machine system consisted of virtual machine engine running on the server side and remote console applet on the web browser.

In the following sub-sections, we give details about virtual machine engine and remote console applets.

3.1 Virtual machine engine (Server side application)

The virtual machine engine itself is a set of servlets, automatically launched by JWS. It consists of *VMParser*, *VMServlet* and *VMCom* servlets.

After activation, *VMParser* servlet will wait for the external request messages. Once a message comes to the JWS, *VMParser* interprets the message and sends the execution command to *VMServlet*.

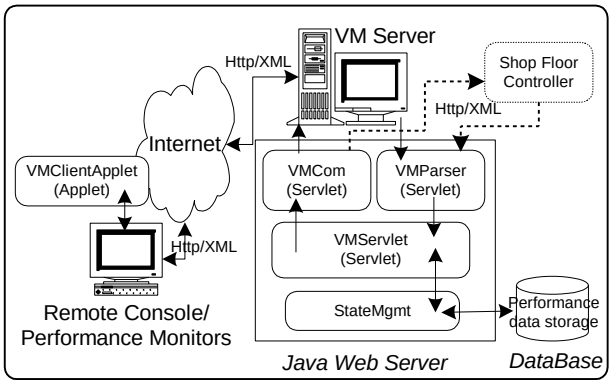

Figure 2: The architecture of virtual machine

VMServlet plays a role of a simulation engine to manage the event calendar and simulation time control. It generates the random numbers for the processing time and random events that request operator interactions such as supplies replenishment and error collection. All of the events happened in the virtual machine are generated and managed in *VMServlet* module. Another function of *VMServlet* is the state management related with manipulation and performance tracking of virtual machine. While the virtual machine is running, *StateMgmt* object gathers and stores the relevant data based on the state model in the database.

VMCom servlet provide the function of broadcasting the response messages to the clients. If *VMServlet* generates the response message, *VMCom* servlet spreads the message to the multiple clients running as a remote console and performance monitors. This servlet also has a function to communicate with single client that requests the performance data individually.

This architecture also can be used directly for the testing bed of shop floor controllers. Since the interface is compliant to the HTTP protocol and the XML message schema is defined in the DTD (Document Type Definition) file, *VMParser* and *VMCom* servlets in virtual machine engine are designed to communicate with higher-level controllers without modification.

The main features and functionalities of the virtual machine can be summarized as follows:

- Respond to the synchronous machine status inquiry
- Respond to external interactive control command for machine manipulation. For example,
  Start/ Stop / Pause/ Resume Machine Simulation
  Error Correction
  Supplies Replenishment
- Generate asynchronous event report during machine simulation. These events include:
  ItemEnter, ItmExit, ItemProcessStart, ItemProcess End, ItemProcessed, StateChange,…

3.2 Remote console (Client side VM user interface)

The virtual machine can also simulate the operator interface via a remote console. It is used for on-line monitoring and machine manipulation from a web browser. Depending on the access right, the user interface may act as a remote console or performance monitors. Like real machine, a remote console can manipulate the virtual machine with control command, but other performance monitors are used only for monitoring virtual machine performance.

The remote console offers Control, Performance, and Communication panels for an operator to configure, manipulate and monitor a virtual machine. In control panel, there are sub panels for Production Management, Process Management and Material Management. Figure 3 shows the Production Management sub-panel.

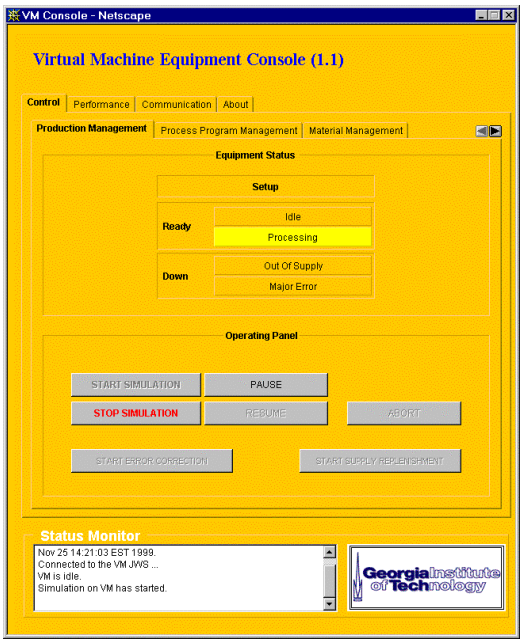

Figure 3: Production Control sub-panel in Control panel

In the Performance Panel, there are sub-panels for Equipment Time, Productivity Analysis and Quality Analysis. The Equipment Time interface is shown in Figure 4.

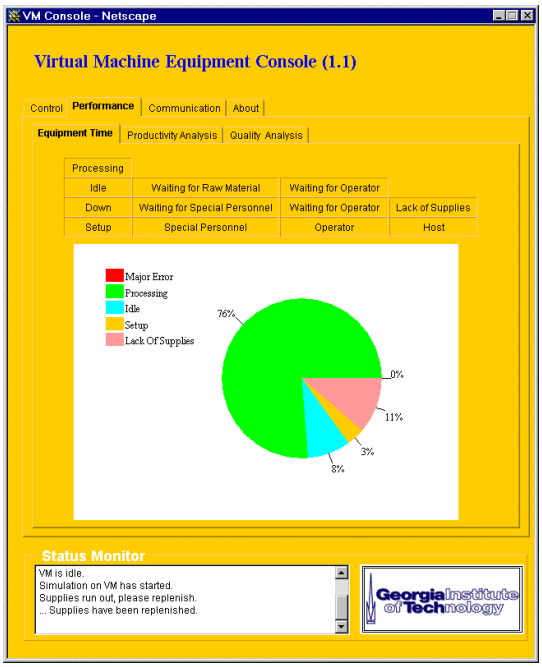

Figure 4: The equipment times from Performance console

4 SUMMARY

Virtual machine is a model based software module resides on a web server that simulates the behavior of a machine in terms of state change and its interactions. In this research, we implemented the virtual machine as a web based application consisted of servlets in Java Web Server. We used the HTTP/XML protocol to provide platform independent communication with clients.

Remote console is a user interface running on the web browser as an applet. Like the equipment console of real machine, it provides the functionalities to control the virtual machine and to monitor the state variables and performance measure of virtual machine.

Due to the structure of modular design and standard interface, it can be used to test of the different shop floor strategies interactively, and remote console feature can also be used for operator training and evaluating.

REFERENCES

Banks, J. 1998. *Handbook of Simulation*, Willy Interscience, p3.

Davis, W. J. Brook, A. C. and Lee, M. S. 1997. A New Simulation methodology for master production scheduling, *Proceedings of the 1997 IEEE International Conference on Systems and Cybernetics*.

Narayanan, S. Schneider, N. L. Patel, C. Reddy, N. Carrico, T. M. & DiPasquale, J. 1997. An object-based architecture for developing interactive simulations using Java, *Simulation*, 69(3), September, pp. 153-171.

Patzer, A. et al., 1999. *Professional Java Server Programming: with Servlets, JavaServer Pages (JSP), XML, Enterprise JavaBeans (EJB), JNDI, CORBA, Jini and Javaspaces*, Wrox Press.

AUTHOR BIOGRAPHIES

HANSOO KIM received his B.S. degree from Hanyang University, and M.S. degree from Korea Advanced Institute of Science and Technology (KAIST). After working for Samsung SDS as a simulation consultant for five years, he has studied the Ph.D. course from 1998 in Georgia Tech. His research interests include the analysis and control of discrete event system, real-time control and simulation in the fields of manufacturing and logistics.

HUA X. DU received her B.S. degree from Tsinghua University, Beijing, P.R.China, in 1994; her M.S. and Ph.D degree in industrial engineering from Georgia Institute of Technology in 1997 and 2000, respectively. Her research interests include large-scaled system integration, system analysis and real-time control. Upon graduation, she joined the research and development team of Siemens Energy & Automation, Inc. in Norcross, GA, continuing the research in system modeling, analysis and design.

CHEN ZHOU is an Associate Professor in the School of Industrial and Systems Engineering at Georgia Institute of Technology. He received his B.S.M.E. in Tianjin University in China, M.S.M.E., and Ph.D. in Industrial Engineering from the Pennsylvania State University. His email address is chen.zhou@isye.gatech.edu
and his web address is
http://www.isye.gatech.edu/people/faculty/Chen_Zhou.