Proceedings of the 2002 Winter Simulation Conference
E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, eds.

INTEGRATED DEVELOPMENT OF NONLINEAR PROCESS PLANNING AND SIMULATION-BASED SHOP FLOOR CONTROL

Sambong Kim
Jungyoup Woo
Sungsik Park
Buhwan Jung
Hyunbo Cho

Division of Mechanical and Industrial Engineering
Pohang University of Science and Technology
San 31 Hyoja, Pohang 790-784, KOREA

ABSTRACT

Although several methods of simulation-based SFC have been suggested for the SFCS, these researches paid only attention to the generation of a target simulation code and could not be fully integrated with the SFCS. Hence, this paper focuses on the conceptual architecture for the rapid and adaptive realization of a simulation-based SFCS for a discrete part manufacturing system. The developed simulation-based SFCS can process non-linear process plans. To this end, the new simulator engine must be developed. It advances the simulation clock and drives the simulation-based SFCS by investigating the information contents specified in the process and resource models.

1 INTRODUCTION

A shop floor control system (SFCS) is the central part of a computer integrated manufacturing (CIM) system. It governs the "what", "where", "when", and "how" in a shop floor by managing production activities to fill required orders. Receiving process plans from a computer aided process planning (CAPP) system and production orders from a business system, the SFCS is responsible for resolving various problems such as planning and scheduling, task execution, recovery from errors, etc. In particular, the process plan generated from the CAPP system should provide the SFCS with the information requirements necessary to solve these problems.

This paper focuses on the conceptual architecture for the rapid and adaptive realization of a simulation-based SFCS for a discrete part manufacturing system. The developed simulation-based SFCS can process non-linear process plans. To this end, the new simulator engine must be developed. It advances the simulation clock and drives the simulation-based SFCS by investigating the information contents specified in the process and resource models. The proposed architecture can reduce the cost and time significantly in developing and maintaining the SFCS.

2 RELATED WORK

For the implementation of the SFCS, several methods have been suggested. Firstly, the approach based on simulation model generation was proposed. Murray and Shepppard generated the simulation code in SIMAN using the structured interactive dialog for the acquisition of a model specification (Murray and Sheppard 1988). Schroer and Tseng generated the simulation code in GPSS using the interactive dialogue (Schroer and Tseng 1989). Using a batch file containing a list of operation equations and system specification, Yuan *et al.* generated the simulation code in SIMAN (Yuan *et al*. 1993). Lee *et al.* generated the simulation code in Witness (Lee *et al.* 2000). However, these researches only paid attention to the generation of a target simulation code and could not be fully associated with the SFCS.

There have been many commercial simulators for SFC, but they have either process-oriented view or resource-oriented view. Most of the early simulators, including GPSS (Gorden 1975), SIMAN (Pegden 1982), SIMSCRIPT (Law and Larmey 1984), and SLAM (Pritsker 1986) are examples of process-oriented languages. These simulators use a complex grammar and model systems as networks in which the nodes represent queues and the branches represent the paths connecting those queues. Each queue is connected to a process, and the emphasis is on the flow of entities through those processes – not the processes themselves; the processes themselves and the resources that perform them are hidden. This means that the impact of the resources' properties and their distribution layout are hidden from the user. Consequently, those simulators can-

not easily support the dynamic decisions related to the concurrent movement of parts.

Recent simulators, including WITNESS (AT&T 1995), ProModel (Production 1989), AutoMod (AutoSimulations 1989), and SIMFACTORY (CACI 1990), are resource-oriented languages. They view a system as the specification and arrangement of resources that performs a number of different operations on entities as they flow through the resources. The emphasis is on resources, not the entity flow. The process sequences are hidden within and across resources, as are the part characteristics, which makes it complicated to build a simulation model and modify the built simulation model.

3 DEVELOPMENT OF A SIMULATION-BASED SHOP FLOOR CONTROL SYSTEM

A simulation-based SFCS consist of a simulator engine, decision maker, execution system, and simulation model (resource and process models), and communicates with a business system and equipment controllers. The simulation-based SFCS have the exemplary process as follows. 1) The simulator engine receives the order with the information of a part type, quantity, due-date, and priority from the business system 2) and then requests the part information, process plan, resource information, and relationships among resources to the simulation model. 3) The simulator engine generates concrete tasks for equipment controllers (for example, EID274-D for a milling machine and VAL II for a robot) by calling the decision maker that is responsible for planning and scheduling the given order. 4) The simulator engine operates equipment controllers via the execution system. The architecture of the simulation-based SFCS is described in Figure 1.

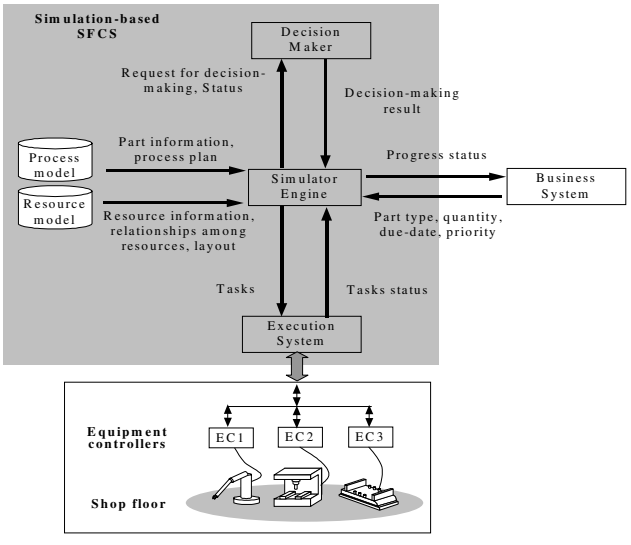

Figure 1: Framework of simulation-based shop floor control system

4 DEVELOPMENT OF THE SIMULATION MODEL

The simulation model is the basic information of the shop floor. In this paper, the simulation model is developed in both process-oriented and resource-oriented views for the following reasons. First, nonlinear process routings, which include both AND and OR branches, could be modeled more easily. Second, all of the important information about the parts and the resources would be visible, hence changeable, in the model. Third, the decisions made by the SFCS, which require up-to-date information about both the parts and the resources, could be better analyzed.

The process model is comprised of part types, process types like milling or turning processes, AND/OR junctions which represent part flow logic for flexible nonlinear process plan, and links that represent temporal precedence among processes. The resource model is composed of processing like machines, storage like a buffer and AS/RS, and transport like a robot, conveyor, and AGV. The types and symbols used in the process model and resource model are shown in Figure 2 and Figure 3.

	Head	Process	Junction	Link
Symbol	Part type	Process type	Junction type	→
Type	Part types	Milling, Drilling, Turning, Chamfering, etc.	AND(A), OR(O)	None
Characteristics	Prefixed symbol for every process plan	Any process type that requires "duration"	Part flow logic for flexible nonlinear process plan	Temporal precedence among processes

Figure 2: Symbols used in the process model

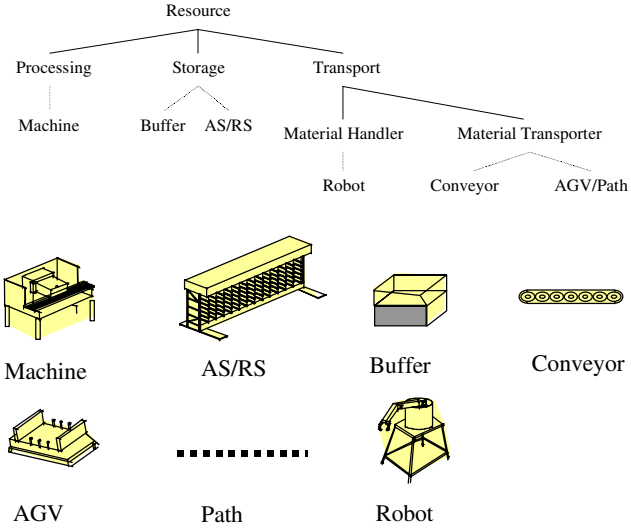

Figure 3: Types and symbols in the resource model

The process model and resource model should be updated whenever the associated changes occur in the shop floor. In particular, the changes can be classified as followings: 1) arrivals of new parts with new process plans, 2) changes of resource properties, 3) changes of shop layout, and 4) resource breakdowns (resources not available any more).

5 DEVELOPMENT OF THE SIMULATOR ENGINE

The simulator engine has four modules: initializer, clock manager, task manager, and kernel. The initializer sets the preliminary values for the simulation clock, system's state, and task list. The clock manager selects the next task to be fired from the task list and pushes forward the simulation clock to the time of that task (Law and Kelton 1991). The task manager updates the system's state, generates future tasks, and adds them to the task list. The kernel manages the other modules. It invokes the initializer and then repeatedly invokes the clock manager and the task manager until the end of the simulation.

While a simulation is running to drive the SFCS, the simulator engine retrieves necessary information from the process and resource models, requests the decision maker to make decisions, generates future tasks, and downloads the generated tasks to the execution system. Hence, the simulator engine should have continuous interactions with the simulation model, decision maker, and execution system. The detailed information flow around the simulator engine is illustrated in Figure 4.

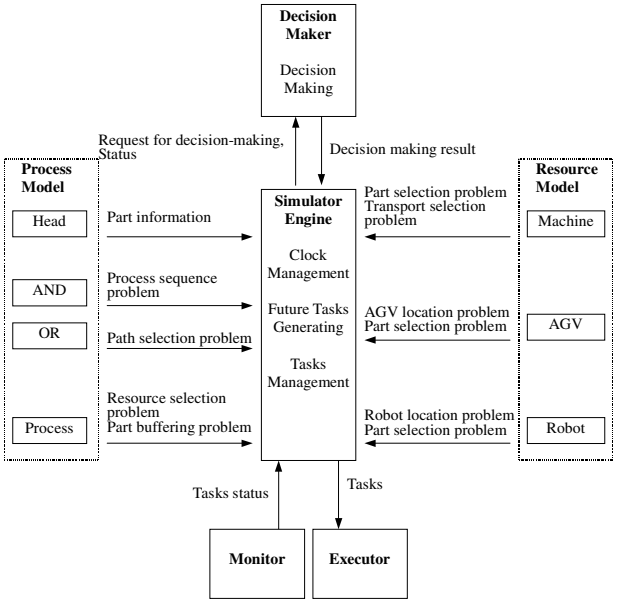

Figure 4: Information flow in the simulator engine

6 CASE STUDY

The result and progress of the simulator engine are shown in Figure 5 and Figure 6. The process model consists of one head, seven processes, and four junctions. The resource model consists of two machines and one robot.

After running the simulation, the AND and OR junctions of process plan are resolved by the process sequence and path selection rules which are determined by the decision maker. The resulting sequence of processes - FACE_MILL->SLOT1-> POCKET-> DRILL1->DRILL2->END_MILL - is shown in the top left window in Figure 5. The bottom left window in Figure 6 shows the detailed log of the progress and status of the simulation for SFC.

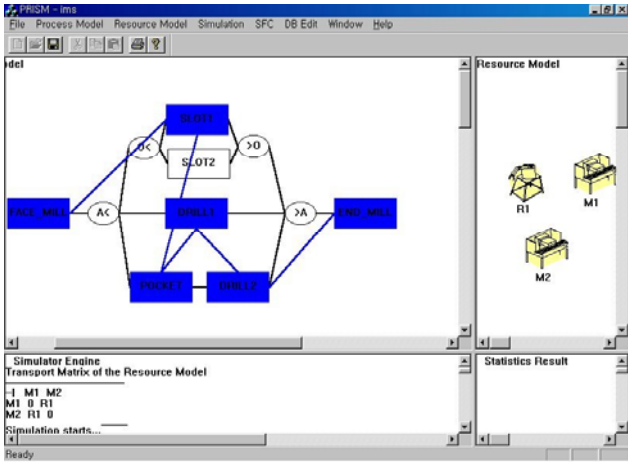

Figure 5: Case study

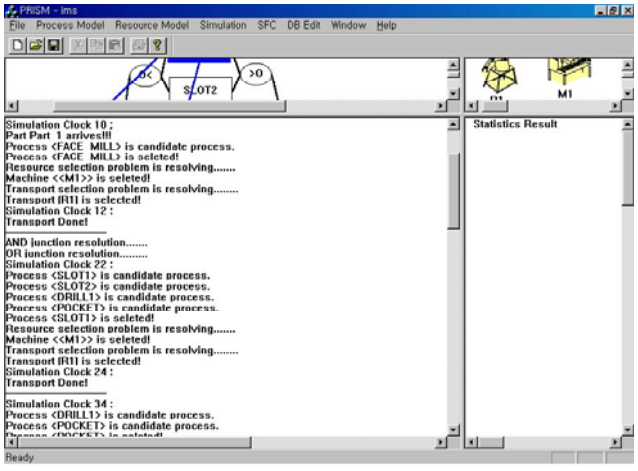

Figure 6: Progress and status of the simulator engine

7 CONCLUSION

The conceptual architecture proposed in this paper is a new paradigm of simulation for SFC. It is made possible by the integration of process and resource models. The simulator engine advances the simulation clock and manages the

tasks by investigating various pieces of information specified in the process and resource models.

In particular, the development time and cost of a SFCS can be significantly reduced and the modification of a SFCS can be facilitated. Moreover, the SFCS can be reconfigured to meet the dynamic changes of the shop environment by regenerating the simulation model. With the proposed concepts, process planning and shop floor control can be systematically integrated and therefore its development and integration cost can be drastically decreased.

REFERENCES

AT&T ISTEL. 1995. *Witness User Manual, Release 7.0*, Visual Interactive Systems, UK.

AutoSimulations, Inc. 1989. *AutoMod II Users Manual*, Bountiful, Utah.

CACI Products Company. 1990. *SIMFACTORY II.5 User's and Reference Manual*, Version 2.0, LaJolla, Calif.

Gordon, G. 1975. *The Application of GPSS V to Discrete System Simulation*, Englewood Cliffs, N. J. Prentice-Hall.

Law, A. M. and W. D. Kelton. 1991. *Simulation Modeling and Analysis*, McGRAW-HILL, Singapore.

Law, A. M. and C. S. Larmey. 1984. *Introduction to Simulation Using SIMSCRIPT II.5*, CACI Products Company, La Jolla, Calif.

Lee, S., H. Cho, and M. Jung. 2000. A conceptual framework for generation of simulation models from process plan and resource configuration. *International Journal of Production Research*, 38(4): 811-828.

Murray, K. J. and S. V. Sheppard. 1988. Knowledge based simulation model specifications. *Simulation*, 50(3): 112-119.

Pegden, C. D. 1982. *Introduction to SIMAN with version two enhancements*, Systems Modeling Corporation, Pennsylvania.

Pritsker, A. A. B. 1986. *Introduction to Simulation and SLAMII*, Systems Publishing Corporation.

Production Modeling Corporation of Utah. 1989. *Pro-Model User's Manual*, Orem, Utah.

Schroer, B. J. and F. T. Tseng. 1989. An intelligent assistant for manufacturing system simulation. *International Journal of Production Research*, 27(15) : 1665-1683.

Yuan, Y., C. A. Dogan, and G. L. Viegelahn. 1993. A flexible simulation model generator., *Computers and Industrial Engineering*, 24(2) : 165-175.

AUTHOR BIOGRAPHIES

SAMBONG KIM is a Master's student in the School of Mechanical and Industrial Engineering at the Pohang University of Science and Technology. He received his B.S. degrees in Material Science and Engineering and Industrial Engineering from the same university in 2001. His e-mail address is <bongyi@postech.ac.kr>.

JUNGYOUP WOO is a Master's student in the School of Mechanical and Industrial Engineering at the Pohang University of Science and Technology. He received his B.S. degree in Industrial Engineering from the same university in 2001. His e-mail address is <woo@postech.ac.kr>.

SUNGSIK PARK is a Ph.D. student in the School of Mechanical and Industrial Engineering at the Pohang University of Science and Technology. He received his B.S. and M.S. degrees in Mathematics and Industrial Engineering from the same university in 1997 and 1999, respectively. His areas of expertise include shop floor control and automatic generation of execution system. His e-mail address is <spark1@postech.ac.kr>.

BUHWAN JUNG is a Ph.D. student in the School of Mechanical and Industrial Engineering at the Pohang University of Science and Technology. He received his B.S. and M.S. degrees in Industrial Engineering from the same university in 2000 and 2002, respectively. His areas of expertise include shop floor control. His e-mail address is <bjeong@postech.ac.kr>.

HYUNBO CHO is an Associate Professor in the Department of Industrial Engineering at the Pohang University of Science and Technology. He received his B.S. and M.S. degrees in Industrial Engineering from Seoul National University in 1986 and 1988, respectively, and his Ph.D. in Industrial Engineering with a specialization in Manufacturing Systems Engineering from Texas A&M University in 1993. His Ph.D. dissertation was associated with defining and implementing an intelligent workstation controller for CIM. He was a recipient of the SME's 1997 Outstanding Young Manufacturing Engineer Award. His areas of expertise include shop floor control, process engineering, and e-manufacturing. He is an active member of IIE and SME and his e-mail address is <hcho@postech.ac.kr>.