Proceedings of the 1990 Winter Simulation Conference
Osman Balci, Randall P. Sadowski, Richard E. Nance (eds.)

IMPLEMENTATION ISSUES USING SIMULATION FOR REAL-TIME SCHEDULING, CONTROL, AND MONITORING

Catherine M. Harmonosky

Department of Industrial and Management Systems Engineering
207 Hammond Building
The Pennsylvania State University
University Park, Pennsylvania 16802

ABSTRACT

As Computer Integrated Manufacturing (CIM) Systems become more prevalent, more emphasis is being placed upon effective production control methods to insure the system's operational success. Real-time scheduling and control decision-making aids that provide insight into the long-term effect upon system performance from possible alternate decisions made in real-time would provide a powerful tool for the scheduler. Although using simulation as a real-time decision-making tool in a CIM System seems conceptually straightforward, there remain many implementation issues that must be solved prior to a successful actual application in a physical system. This paper discusses some of the most significant implementation issues, based upon personal experience trying to apply simulation as a real-time decision-making tool in a laboratory setting as well as interchanges with other industrial experts working in this area.

1. INTRODUCTION

In the manufacturing environment, the terms Computer Integrated Manufacturing (CIM) Systems and Flexible Manufacturing Systems (FMS) have become more commonplace over the last ten to fifteen years. More people are realizing that the success of Computer Integrated Manufacturing (CIM) Systems depends upon effective production scheduling and control, often accomplished through real-time decisions made on the shop floor. The flexibility often heralded as a major advantage of these types of systems [Roch 1986] actually complicates the scheduling and control task a) by increasing the number of routes needed for material handling, b) by making tracking of jobs more difficult, or c) by increasing the demand upon the scheduler (human or automated) for evaluation of the environment and subsequent real-time decision making [Harmonosky 1990].

Due to the dynamic nature of manufacturing processes, intelligent real-time scheduling has always been a desirable, but elusive, goal. The premise is that real-time adjustment of scheduling or sequencing rules, e.g. SPT, EDD, FCFS, may lead to better long-term system performance. Ideally, a system scheduling or control decision made in real-time should have a positive impact upon long-term system performance. Tools are needed to aid in this decision assessment, which have a capability to look ahead into the future system conditions. With increased shop floor level computing power and more emphasis placed on networked computer communications, there has been renewed industrial interest and increased academic research in the real-time scheduling area.

This interest leads to the natural attempted application of simulation techniques, proven to assist the manufacturing community in initial system design evaluation, to the problem of real-time control [Harmonosky and Robohn 1990; Grant, et al. 1988; Erickson et al. 1987; Sadowski 1985]. More specifically, it is logical to try and use discrete event simulation languages, which have been specially designed for application in environments such as manufacturing and are accepted by the manufacturing community. Although this may seem to be an easy match conceptually, an attempt at actual application to a physical system highlights many important application issues and impediments that must be discussed, studied, and overcome before simulation could be successfully implemented as a real-time scheduling tool.

This paper discusses these operational issues and possible solutions to some problems. The information is based upon experience working with an existing discrete event simulation language being applied as a real-time decision tool in a laboratory setting as well as interchanges with other industrial experts working in this area.

2. PROPOSED STRUCTURE FOR REAL-TIME CONTROL

The scenario for using simulation as a real-time decision making tool is illustrated in Figure 1. A computer simulation of a CIM system is linked with the actual physical system. Once the link between the simulation and the system is established, the simulation logic will be controlled by the actual system communication signals, dictating start and stop of robot movement, equipment processing, and cart movement. The simulation will always reflect the current system status, and it will be in effect monitoring the system. Then, when a system production control decision is needed, the starting condition for the simulation is the actual system status. For each different control decision option, a simulation run may be executed for some period of time. The future impact upon the system due to different decisions may be evaluated by analyzing simulation statistical results [Harmonosky and Barrick 1988]. In this mode, the simulation model is used as a real-time production control tool with look-ahead system assessment capabilities.

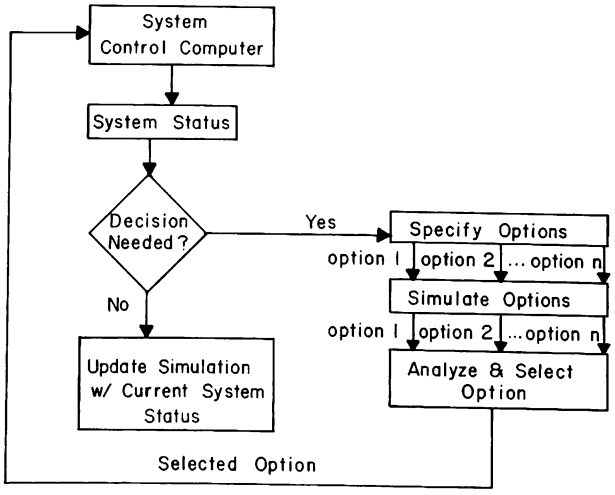

Figure 1. Scenario for Interfacing Simulation with Physical System for Real-time Control

This scenario gives two basic modes of operation for the simulation model. First, a monitoring mode is used when the model is linked directly to the physical system constantly receiving status information. In this mode, delays for part processing and movement are dictated by information received from the system indicating an end of activity event (i.e. the delay is not a value selected from a probability distribution). When in the monitoring mode, graphical animation capabilities of a simulation language can be invaluable, allowing the user to effectively "view" the physical system status without being on the factory floor.

Second, a decision-making mode is used when the model evaluates different control decision options, being utilized in a more traditional simulation role. One issue becomes how long to make the look-ahead planning horizon, which has a direct effect on the execution time of the model [Wu and Wysk 1989]. The more time each evaluative simulation runs, the longer it will take to make scheduling and control decisions. This affects the degree to which truly "real-time" control of a system is maintained. The tradeoff that must be considered is having the look-ahead horizon long enough to produce valid statistics on performance measures capturing the effects of some minimum number of observations versus having a rapid control decision keeping the spirit of a real-time response [Harmonosky and Robohn 1990; Davis and Jones 1988].

3. MAJOR ISSUES/CONSIDERATIONS

This section addresses some specific operational issues assuming the structure scenario discussed in Section 2 and application of a discrete event simulation language.

3.1 Communications to a Physical System

The mechanism for retrieving system status data from the physical system for use in the simulation poses several problems. First, the base language of the simulation language may not be the language used by the plant level control software. Therefore, any information passed from the system must be preprocessed by a program written in a general purpose language to put system data into a format the simulation will understand. Second, in most cases, this is not simply passing the same message or numerical value by translation from the physical system to the simulation. The form of the message or value needed by the simulation may be completely different. For example, the physical system may send simple flags between personal computers (pc's) that control machines and higher level pc's; however, in order for the simulation to process correctly, a different unique value may be needed or a particular variable in the simulation may have to be set to some specific value based on the physical system flag. The necessary mapping between system flags and model data requirements may not be an easy task depending on the complexity and size of the system and the number of messages passed per minute.

Another interfacing issue concerns what detail of data is necessary for the simulation to properly emulate and monitor the system and where to get the data in the physical system. If the computer communication hierarchy is facilitated through hardwiring links to one central controlling computer, it may be possible to access all data from monitoring this single central computer. If a network is used to accomplish the communication hierarchy, it may be possible to access all data from monitoring the network by "camping" the simulation on the line.

However, in either case, the central control computer or the network may not be receiving (or have access to) all the messages passed in the system due to the division of tasks associated with hierarchical control schemes. Therefore, a simulation that needs system status information at a very detailed machine operational level may not be able to attain this information from the central controller or the network. For example, Figure 2 shows a possible hierarchical communication scheme with the names associated with the pc's indicating the piece of equipment it controls. Note that the Work Station A computer is responsible for all part movement within the station, including transfers between the robot and machines and buffer. It does not transmit this information over the network, because the central control computer does not need this detailed information of intra-station activity. On the other hand, a simulation that is emulating and monitoring the physical system finds the Work Station A detailed information necessary to run.

Thus, the interfacing issue involves language compatibilities, types of data necessary, and from where to access the data in the communication hierarchy. One other point is worth mentioning regards use of networks. It may seem that networks make the interfacing problem much easier; however, a language compatibility issue still exists. Also, the rate at which a network transfers information will have an impact on how "up-to-date" the simulation monitor will be at any point in time.

3.2 Saving Simulation Information at Decision Points

When using simulation as a means of real-time control, monitoring the system with the simulation (as suggested in Section 2) provides a means of always maintaining the current system status. This way when a real-time production control decision is needed, the system status is available and no time delay is associated with obtaining that information. Each different scheduling/control alternative should be evaluated using this current system status as a starting point. However, conflicts may be encountered when executing different evaluative simulation model runs with even a simple type of scheduling alternative.

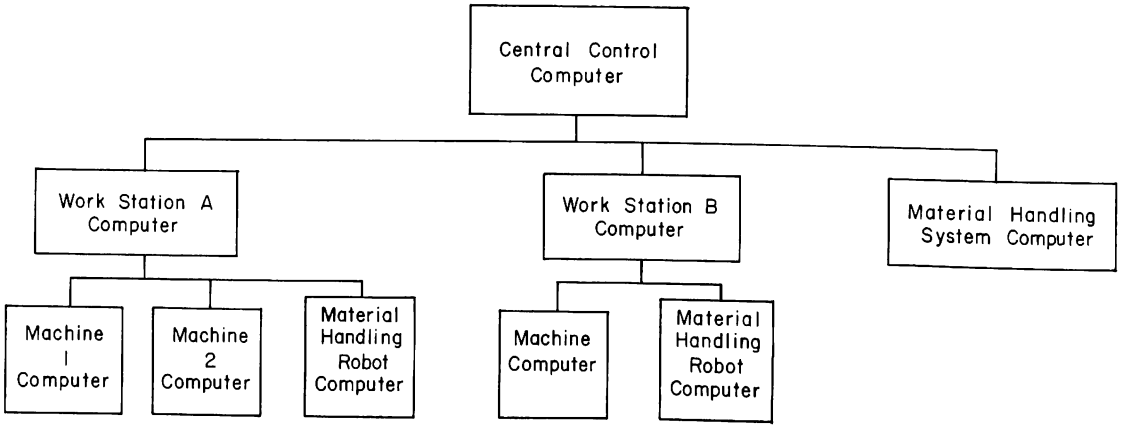

Figure 2. Hierarchical Computer Communication Scheme Example

This issue is best discussed with an example. Consider applying different queue disciplines in a model to reflect the execution of different sequencing rules in the physical system. The system may be running for 100 minutes in a first come first served (FCFS) sequencing mode, when some kind of interruption occurs, such as a machine breakdown or the introduction of a new part with abnormally high processing priority. The statistical information for performance measures during these 100 minutes of FCFS operation as well as the exact system status at the time of interruption must somehow be saved into a status file that the simulation package can read as initialization information to evaluate different decision alternatives. Now in the decision-making mode, the simulation of each sequencing rule uses a model of the physical system with a specific queue priority, such as shortest processing time (SPT), earliest due date (EDD), etc., using the first 100 minutes of FCFS data as the initialization information.

In many simulation languages, a capability exists to save a particular system state and then recall that state at some later time. It may be reasonable to assume that only statistical information from the first 100 minutes of operation and system status data (e.g. entities in queues, in process, etc.) would be recalled. However, if the queue priority rules are pieces of data automatically saved by the simulation when a system snapshot is taken, queue priorities will remain the same as those first 100 minutes, effectively inhibiting the evaluation of more than one sequencing rule. To address this problem, it may be necessary to write extensive user-code that saves selective data referring to system status and write user-code to reinitialize the simulation to evaluate another possible decision, bypassing the language-provided snapshot saving functions. This may not be a trivial user-code to write, depending upon accessibility into the simulation language's structure (e.g. possibly needing to modify variables that are in proprietary code).

3.3 A priori Decision Alternative Knowledge

As the previous section 3.2 suggested, the types of alternative production control decisions that someone may wish to evaluate may create specific evaluation problems. Also, the simulation of different alternatives may require modification of the existing model that would necessitate recompiling the model before a new run could be made (e.g. new queue priority, alternative job route). When trying to implement a real-time decision making tool, the user cannot afford to waste time making model modifications and recompiling code at each decision point, which would effectively negate the "real-time" nature of the tool.

To circumvent this problem, an individual simulation model could be constructed for each type of alternative decisions the user expects to evaluate. The individual models would typically be extensions and/or slight modifications of the base model, but would exist in a compiled state and could simply be selected and initialized with the appropriate information. This is much faster than constantly generating each model desired. Further, when considering eventual application of the tool in an industrial setting, the scheduler who may be extensively using the tool may not be an expert at simulation model generation, necessitating the availability of existing compiled models.

Then, one necessary element for the rapid response needed for real-time implementation is some a priori knowledge of the alternative scheduling/control decisions that are to be simulated. Using the same example as the previous section, if the base model is the physical system operating under a FCFS scheduling rule and candidate rules include SPT and EDD, those extended models corresponding to the SPT and EDD rules should already exist when the simulation is ready for decision-making mode.

3.4 Specialized Model Structure

Another issue of real-time simulation implementation is the model structure. In order to facilitate easier interface with the physical system, the model structure may have to take a form that is different from the traditional perspective of workpieces moving between operations which drives the system. Because CIM systems are driven by messages throughout the computer communication hierarchy, it may be appropriate to focus the simulation model upon signaling logic among the controlling computers. The idea is to have the simulation emulate communication signals which trigger activities (e.g. cutting on a lathe, retrieving raw material). The logic used in the actual system communication hierarchy to trigger system activities is the same logic used to control the simulation [Harmonosky 1990].

With this model structure, communication requests for service at a given work station are placed in a queue, sending a signal to the model section emulating that work station, and the request remains in that queue until another communication signal is sent indicating work station availability. This may be accomplished through a series of detached queues. Therefore, the job part entity does not seize any resources--rather, the part is delayed in various queues representing different system activities, such as communication delays and processing or handling delays. Because the simulation structure emulates the computer communication hierarchy, an interfacing framework exists to take the triggering signals from the physical system instead of from within the simulation [Harmonosky and Barrick 1988]. This facilitates the continuous monitoring allowing the simulation to always reflect the current system status.

Another advantage to this structure is the allowance of conditional delay times for entities in a processing delay (e.g. machining operation), necessary in the real-time mode. In a normal simulation the processing delays are dictated by some time sampled from a distribution. When linked to a physical system, the delay is not a fixed number but rather an interval that is ended by some triggering signal. The structure described here supports this type of delay.

3.5 The Decision-Making Mode

Once the decision-making mode has been entered, implementation issues include the stochastic or deterministic nature of the look-ahead and evaluation of the results. Typically in simulation studies, future random events, such as machine breakdowns or unexpectedly long processing times, are included in the analysis to get a better estimate of 'steady-state' expected performance. However, it is conceivable that a look-ahead window length may be the next 8 hours or 24 hours, due to the need to get quick answers and run several alternatives. The question must be addressed of what types of random events are reasonable to include during these types of run lengths and/or what would be the inaccuracies introduced with assuming more deterministic behavior.

When analyzing alternatives in the initial implementation of a simulation real-time control system, it is probable that alternative decisions for testing will come from human scheduling expertise dictating 3 or 4 alternatives. As these alternatives are simulated in the look-ahead mode, the output statistics will be available for analysis bye the scheduler. The types of data comparisons and the ultimate selection of an alternative to implement will be the scheduler's task based on the output statistics available from the look-ahead simulation of each alternative. The analysis and evaluation of results will undoubtedly be system dependent considering the system performance measure and any secondary criteria. However, as the real-time control system is used, each time a decision is made, information regarding the cause (system conditions) and the effect (specific action and result) could be recorded. After enough cause-effect relationships have been noted, the information may be arranged in an expert system, automating the alternatives analysis. Then, issues involving experimental design and automating the results analysis must be resolved.

3.6 Recovery from Decisions

The concept of interrupting the real-time monitoring activity of the simulation and using it in its more traditional role as a look-ahead evaluative tool presents a recovery problem. Although the evaluation of alternative scheduling rules should occur rapidly to keep within the "real-time" framework, the physical system will continue operating while the simulation is in

the decision-making mode. Some mechanism must keep track of shop-floor activities during that period [Erickson et al. 1987], then the simulation must again be reinitialized to current system status when returned to the monitoring mode.

One key stumbling block to this recovery will be in material handling. Material handling devices are usually in continuous motion between points and keeping accurate account of part locations in transit could pose a great challenge. One possible solution is to copy the simulation at the time a decision is needed to be used in the decision-making mode while a monitoring mode simulation continues running in parallel. This would require some type of multi-tasking environment (which may not be feasible) and depends upon the simulation language's capability to support this copying concept. Another possible solution is to continue receiving signals from the physical system and store them in a file. When the simulation returns to the monitoring mode, it would process all the signals to again reflect current system status. The success of this technique is a function of the amount of time spent in the decision-making mode and the rate of signalling in the physical system. This may be reasonable if the time spent in decision-making mode is small, processing times are on the order of several minutes, and the system is not very congested (i.e. not many concurrent entities which are not idle in queues). However, if it takes too long for the simulation to process all the stored signals, the system may be again ready for a decision before the simulation is ready to assist.

4. SIMULATION LANGUAGE VS. GENERAL-PURPOSE LANGUAGE

Up to this point, a discrete-event simulation language (such as SIMAN, SLAM II, or GPSS) has formed the foundation for discussion. However, it may be possible to use a general-purpose programming language such as FORTRAN or C to write all the necessary simulation and control software. This section addresses some advantages and disadvantages associated with each option.

Simulation languages developed for use in manufacturing environments may be very suitable for modeling a CIM-type system, but they require a user knowledgeable in the specific simulation language used. Most industrial engineers are familiar with some type of simulation language, but, people from other disciplines that may be involved in a real-time control project (e.g. mechanical and electrical engineers and computer scientists) are more likely to have some knowledge of a general purpose language. In addition, most simulation languages do not have the capability to directly communicate with external programs, due to their specialized structure. This necessitates the use of a general purpose language to some extent. General purpose languages may be used to construct tailor-made simulation applications from ground zero, with extensive interfacing capabilities.

However, a significant advantage of using a simulation language is that queue structures, file manipulation, event calendar maintenance, data saving, and statistical calculation capabilities are included in simulation languages. All these basic capabilities would have to be manually coded using a general-purpose language. Many simulation languages have built-in animation constructs that are crucial to the monitoring task. These could be done in general-purpose programming languages, but the overhead involved in coding animation routines is rather high.

5. CONCLUDING REMARKS

Although using simulation as a real-time decision-making tool in a CIM environment seems very natural, there is still a significant gap between conceptual design and actual operational application to a physical system. Some of the most significant issues discussed in this paper include modeling structure, interfacing to the physical system, saving system status for evaluating alternatives, and recovery at decision points. However, with more researchers and practitioners becoming actively involved in resolving these difficulties, future application of this technique appears very promising.

ACKNOWLEDGEMENT

This material is based upon work supported by the National Science Foundation under Grant No. DDM-8909760. The Government has certain right in this material.

REFERENCES

Davis, W.J. and A.T. Jones (1988), "A Real-Time Production Scheduler for a Stochastic Manufacturing Environment," *International Journal of Computer Integrated Manufacturing 1*, 2, 101-112.

Erickson, C., A. Vandenberge, and T. Miles (1987), "Simulation, Animation, and Shop-Floor Control", In *Proceedings of the 1987 Winter Simulation Conference*, A. Thesen, H. Grant, and W.D. Kelton, Eds. IEEE, Piscataway, NJ, 649-653.

Grant, F.H., S.Y. Nof, and D.G. MacFarland (1988), "Adaptive/Predictive Scheduling in Real-Time," In *Advances in Manufacturing Systems Integration and Processes: 15th Conference on Production Research and Technology*, D.A. Dornfeld, Ed. Society of Manufacturing Engineers, Dearborn, MI, 277-280.

Harmonosky, C.M. (1990), "Elements of Effective Production Control in a Computer Integrated Manufacturing Environment," In *Proceedings of Manufacturing International '90, Volume I: Intelligent Manufacturing Structure, Control, and Integration*, E. Fisher, C.L. Moodie, LA. Martin-Vega, L. McGinnis, and E.T. Sanii, Eds. The American Society of Mechanical Engineers, New York, NY, 9-13.

Harmonosky, C.M. and D.C. Barrick (1988), "Simulation in a CIM Environment: Structure for Analysis and Real-time Control," In *Proceedings of the 1988 Winter Simulation Conference*, M.A. Abrams, P.L. Haigh, and J.C. Comfort, Eds. IEEE, Piscataway, NJ, 704-711.

Harmonosky, C.M. and S.F. Robohn (1990), "Real-Time Scheduling in Computer Integrated Manufacturing: A Review of Recent Research," Working Paper No. 90-108, Department of Industrial and Management Systems Engineering, Pennsylvania State University, University Park, PA.

Roch, A.J. (1986), "Flexible Automation Holds Key to Competitive Advantage for Aerospace Manufacturer," *Industrial Engineering 18*, 11, 52-59.

Sadowski, R.P. (1985), "Improving Automated Systems Scheduling" *CIM Review 2*, 1, 10-13.

Wu, S.D. and R.A. Wysk (1989), "An Application of Discrete-Event Simulation to On-Line Control and Scheduling in Flexible Manufacturing," *International Journal of Production Research 27*, 9, 1603-1623.