Proceedings of the 2008 Winter Simulation Conference
S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler eds.

SIMULATION OF PROCESS EXECUTION MONITORING AND ADJUSTMENT SCHEMES

Russell R. Barton
Jun Shu

Dept. of Supply Chain and Information Systems
The Pennsylvania State University
University Park, PA 16802, U.S.A.

ABSTRACT

Optimization and design of production and service operations has been a cornerstone of simulation applications for many years. Recently there has been increasing interest in excellence in process execution, for service processes, supply chains, and manufacturing operations. Simulation again has an important role to play, in evaluating the effectiveness of different execution monitoring schemes. The focus of analysis changes in this setting, however, from monitoring characteristics of resources, such as queue length, waiting time and utilization, to monitoring the timeliness and correctness of the progress of entities through the system, and reacting with real-time adjustments to resources and routings. This paper describes the types of simulation analysis needed, and identifies difficulties in conducting such simulation studies using current commercial software.

1 INTRODUCTION

The focus of systems management has for many years been on the efficiency and effectiveness of the design and operating rules. Few systems operate precisely as intended, however. Most experience randomly occurring aberrations that must be addressed in order to provide high-quality process execution. A recent study at Georgia Institute of Technology (Lurie and Swaminathan 2006) showed that managers tended to over-control supply chain systems in response to random variations in process status data. Process managers need ways to tell when variations are out-of-the ordinary, meriting attention.

The deleterious effects of over-control actions have been addressed in manufacturing settings by quality control methods developed by Shewhart, Deming and others (Wheeler and Chambers 1992). Deming's famous funnel experiment demonstrates the added process variability introduced by over-control: overreaction to normally occurring variation makes matters worse.

Statistical Process Control (SPC) is a fundamental tool in organizations following a Six Sigma approach to quality management. It enables one to detect quality problems based on a statistical model that balances timely detection against the cost of false alarms. For supply chains it enables one to answer questions such as: Are your suppliers' deliveries repeatable? Can you detect changes ('out of control') in the delivery timeliness before there is a crisis? How much might delivery time variation be reduced? How do you tell on a daily or hourly basis which parts of your supplier chains or delivery chains need attention?

SPC methods, long applied to manufacturing processes, are now being applied to monitor the timeliness and correctness of the movement of entities through a process. Because processes can be complex, a performance comparison of statistical monitoring methods with each other, or with traditional process management execution rules, requires simulation models that are more sophisticated than the Monte-Carlo analyses used to assess and compare ordinary SPC methods.

The kind of analysis that is useful in this setting differs from the usual objectives for discrete-event simulation. This in part is due to the historical focus on optimal system design rather than on optimal system monitoring methods to react appropriately to normal and abnormal variations in system performance. Traditional analysis has focused on operational efficiency, usually by examining metrics associated with system resources. These metrics include average queue lengths, waiting times and their distributions, and they are associated with specific resources such as machines, doctors, or call center operators. These quantities are easily captured by simulations maintaining data for each resource. This is a typical structure, regardless of whether the modeling paradigm focuses on event-scheduling or process-interaction (Banks, et al. 2005).

Unfortunately this resource focus for data collection and analysis makes it difficult to monitor (transient) entities or groups of entities for timely and correct movement through a process. Maintaining times and movement by individual entity identifier is required, and this can increase

978-1-4244-2708-6/08/$25.00 ©2008 IEEE

model complexity and slow execution time using current modeling approaches.

Further, process execution monitoring strategies are usually paired with adjustment actions to be taken in the event of an out-of-control signal. These adjustment actions can include modifications to real-world system resources and process flows. Unfortunately, it is difficult to simulate these adjustment actions, because resource assignments and characterizations and process flow logic are generally difficult to change during a simulation run.

The remainder of this paper describes in some detail some process execution monitoring strategies, their value and the need for study via simulation. The required capabilities are summarized, and the 'easy-to-implement' aspects are distinguished from the more difficult. Solutions and workarounds from the literature are cited. The paper concludes with recommendations for data collection capabilities for simulation modeling software. Some of these capabilities currently exist in some packages, but we do not try to present a comprehensive, up-to-date characterization.

2 PROCESS EXECUTION MONITORING

In this section we give an example of a process execution monitoring method. The method applies statistical process control to the timeliness and correctness of the movement of entities through a process.

2.1 SPC for Process Execution Monitoring

One aspect of process performance is the timely progress of an item (i.e., a part, product or transaction) through the process. A second aspect of process performance is the correctness of the progress of each entity through a series of process steps. For example, an order may inadvertently be processed at the wrong station, or a product may be sent to the wrong warehouse.

Effective process execution requires monitoring the timeliness and correctness of the movement of entities through the system. For example, a major automobile manufacturer identifies approximately one hundred milestones in the vehicle delivery chain between the end of the manufacturing line and the arrival to the dealer's parking lot. They employ a large staff of expediters to track vehicle progress through key steps in this process, to detect delays and errors and to take action to notify carriers and resolve problems. Statistical process control is useful in this setting, to avoid expediting items experiencing normal variation, and to ensure that unusually delayed or misrouted items receive attention.

In spite of potential benefits, statistical methods for supply chain process characterization and control charting are only beginning to be implemented. A key reason for this lack of adoption is that timely data on the location of an entity in the system at each instant in time is often not readily available. Even when the data are available, existing SPC technology has not been well suited to multiscale, multivariate, and right-censored data. Recently, statistical process control paired with an RFID-enabled information system (Bhuptani and Moradpour 2005) offers the opportunity to improve supply chain monitoring effectiveness and to reduce the staffing levels needed to perform routine monitoring (Barton and Shu 2008).

2.2 Characteristics of Process Execution Data

Process execution data yields a problem that is distinctly different from the traditional SPC conditions. First, the data and statistical models encompass multiple scales in time and location. For example, the detailed movement of a part through processing in a wafer fabrication facility or through a number of steps in a machine shop has a statistical model based on shop operations and processing characteristics, and is of interest to a particular set of stakeholders in the supply chain, the manufacturing management. Movements of items are over a few meters in terms of distance, and times in minutes or hours. These times and sequences are absorbed into a larger-scale characterization of the supply chain that monitors order-to-delivery times, with different statistical models and different stakeholders, e.g. business-level product managers. Times and distances are on a different scale, with distances perhaps in hundreds or thousands of kilometers, and times in days, weeks, or months. There is a need to understand the linking between these scales, in order to use accurate low-level characterizations to inform aggregate statistical models, and to allow a drill-down for special cause variation when it is observed.

In addition to having multiscale properties, process execution data is inherently multivariate if more than one step of the process is monitored or more than one type of item or transaction is monitored. Understanding the variance-covariance structure of data at varying levels of modeling aggregation is critical to forming appropriate multivariate monitoring statistics and characterizing their behavior.

2.3 SIT-based Process Execution Monitoring

Barton and Shu (2008) describe a framework for monitoring process timeliness and correctness. The key data structure is a state-identity-time triple that captures the progress of uniquely identified entities through process states. The monitoring method requires 'individualized trace data' consisting of real-time observations, each of which is a data triple: a location or state, a unique entity identifier, and a time stamp. An entity can be physical or virtual. An RFID tag (EPCglobal 2004) provides such an identifier or ID for a physical item as it moves through a process. For example, an observation may be recorded as the SIT triple

(Dock #5 Inbound, Item EPC123456789, 10:00 AM 01-25-2006).

Using $\boldsymbol{S}$ to denote all possible states and P($\boldsymbol{S}$) for the power set of $\boldsymbol{S}$, that is, the set of all possible subsets of $\boldsymbol{S}$, each element $s \in$ P($\boldsymbol{S}$) is referred to as a state subset. For example, all RFID readers at all incoming portals of a warehouse can be considered as a (single) state subset called, say, *inbound state* for that warehouse. Between explicitly detectable states, one may define implicit states, such as a transit state, that cannot be detected directly in the available real-world monitoring data. This distinction does not have a major effect on the monitoring methodology, but does require that both arriving and leaving times must be known for every entity at every detectable state.

Grouping, classification, and categorization permitted by subsetting are powerful tools that can provide different managerial insights from a large low-level data set; they permit views at multiple scales. Similarly, one can define power sets for IDs and time epochs.

The following example illustrates the use of SIT data for process execution monitoring.

2.4 Monitoring Process Execution: An Example

We illustrate the SIT/SPC approach to process execution monitoring using a small supply chain example, which is illustrated in Figure 1. The expanded section in the upper part of the figure shows the base level states for which identifier and location data are available. The fundamental unit of product is a case for this example. Each case has a unique ID. Cases of each of four possible product types enter the distribution center through state S1. Two of the four product types, A and B, are packaged into orders at state S2. The other two, C and D, are packaged into orders at state S3. All orders are loaded onto trucks at state S4, and travel to the store loading dock, state S5. Trucks travel directly to the store loading dock from the distribution center state S4.

At the store, cases of products move from the back room to the store floor through a door, denoted by state S6. After restocking shelves, partial cases are returned to the back room. Empty cases are returned to the back room, then to the box crusher, state S7.

In this monitoring scenario, states 1-7 are aggregated as a subset, and the sojourn time in this aggregated state subset is charted for a single product type, product A. This is equivalent to monitoring the time it takes for each case of product A to move from arrival at the distribution center to the final disposal of the empty case container at the box crusher. The raw data have the structure shown in Table 1. While this example focuses on timeliness, SIT-based methods also have been developed for charting the correctness of the sequence of states visited by entities of a particular type (Barton and Shu 2008).

Aggregate sojourn times can be computed by filtering the raw data structure using the specified aggregation. In this example, cases reaching the box crusher (state S7) during a particular time interval are extracted from the data file. Say the value is (s, i, t) = (S7, ID23, 4.5). This is matched against the SIT triple with the same ID, with state identifier S1, and with smallest time value for any triples meeting the previous two criteria. Suppose this is (s', i', t') = (S1, ID23, 1.5). Then the aggregate sojourn time is computed as $t - t' = 4.5 - 1.5 = 3$ time units.

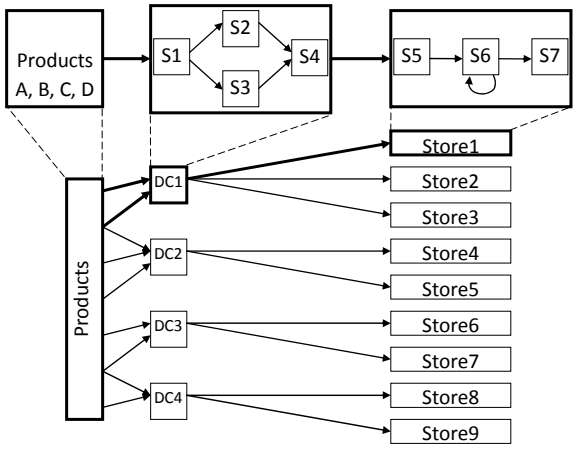

Figure 1: A supply chain process.

Table 1: Structure of raw SIT data for the example. Only a few selected rows are shown.

State	ID	Time
3	sgtin:10010000100114594093	1/2/2006 7:05
5	sgtin:10010000100116455468	1/2/2006 12:08
1	sgtin:10110000100119311748	1/3/2006 11:08
3	sgtin:10110000100113521084	1/3/2006 14:26
7	sgtin:10210000100113385512	1/4/2006 8:25
4	sgtin:10210000100111028774	1/4/2006 8:34
2	sgtin:10310000100113984913	1/5/2006 10:02

The first 150 sojourn times, for this example, were placed in thirty subgroups of five to estimate the mean and standard deviation, and to check for an initial in-control condition. The fitted mean and standard deviation were 26.7 and 6.27 hours, respectively. All points fall within the control limits over the initial 30 subgroups. Figure 2 shows the control chart for all 730 values, in subgroups of five, with observations ordered on aggregate state departure times. There appears to be an upward shift in subgroup mean shortly after subgroup 121, but this shift does not trigger an out-of-control signal until subgroup 130.

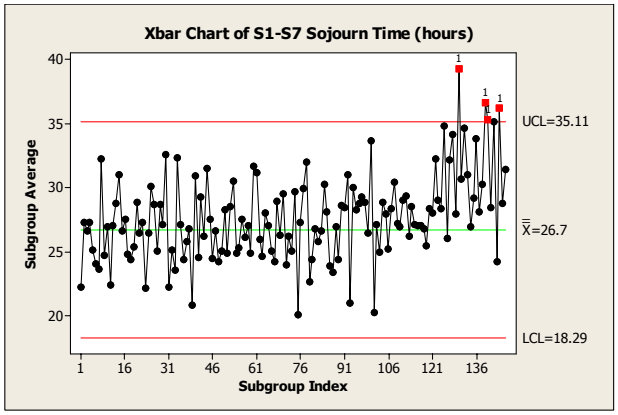

Figure 2: Execution monitoring for aggregate SIT data.

The appropriate adjustment action for this example would depend on examining charts based on disaggregate constructs. Suppose that the order packaging machine at station S2 was improperly wrapping cases of product A, which resulted in significant delay at the store unloading dock, S5, in order to repair and reconstitute cases separated from their order-mates. This might be detected by examining the sojourn time at S5 and finding an out-of-control state. In that situation, product A items might be routed to S3 until the repairs at S2 were completed, but this might require a change in the processing times for the items (C and D) that normally are processed at S3, in addition to changing the item A routing and identifying a new processing time for A items.

2.5 Implications for Simulation

Statistical process monitoring methods are not appropriate for every situation. Some processes operate consistently in control, and for other processes non-statistical methods suffice to maintain reliable process execution. When should statistical methods be used, and how should they be designed? What levels of aggregation are appropriate? What benefit might be had by monitoring correctness in addition to timeliness? What is the best adjustment action to take when an out-of-control situation is detected? How might one quantify the potential benefits of statistical process execution monitoring schemes? Most processes are too complex to permit analytical answers to these questions.

Simulation provides a powerful tool to examine the potential benefits of alternative real-time process execution monitoring methods. These questions can be answered by simulating alternative process monitoring schemes, and simulating the real-time process adjustments implemented as a result of out-of-control signals. This requires the ability to simulate the SIT data collection, as well as simulating the behavior of statistical methods operating against such data, and the real-time corrective actions that might be triggered. This may require the ability to make changes to the simulation model logic as the simulation runs. These issues are discussed in more detail in the next section.

3 CAPABILITIES AND SHORTFALLS OF CURRENT SIMULATION SOFTWARE

A key feature of process execution monitoring is the SIT data structure. One must record the triple of state, ID and time over multiple instants in time. For the measures of timeliness and correctness discussed in Barton and Shu (2008) this data can be reduced to the instants in time when an entity leaves one state and when it arrives at another state. The SIT data structure does not match the usual form of data collected during a discrete-event simulation.

In the rest of this section we describe the capabilities required for simulation of process execution monitoring and adjustment schemes. Then we identify which capabilities exist in current high-level simulation software, and which are still needed.

3.1 Simulation Capabilities Required

Figure 3 shows a high-level view of a simulation designed to examine the effectiveness of process execution monitoring and adjustment schemes. The simulation model includes not just the logic for the process under study, but two other modules, a process monitoring module and a process adjustment module.

The process monitoring module requires the following capabilities:

1. recording of SIT data as each entity enters and leaves a process step,
2. real-time access to SIT data filtered for the particular (transient) entities, state subsets and time windows being charted, and
3. the ability to execute a periodic scheduled 'in-control' vs. 'out-of-control' decision based on the real-time status captured by the charted statistic.

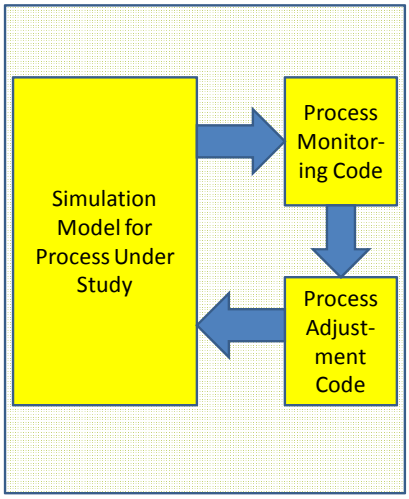

Figure 3: High-level logic for simulating process execution monitoring schemes.

The process adjustment module requires the following capabilities:

4. real-time identification of current resource allocations to all process steps, and current status of all resources and system and entity variable values,
5. the ability to execute an action-selecting decision process based on these data, and
6. the ability to implement the selected action in real time, which might include changing resources and/or resource parameters, routing tables and probabilities.

3.2 Simulation Software Capabilities

The periodic monitoring activity of SPC can be modeled within popular process-oriented simulation packages by creating 'monitor' entities according to a regular (periodic) schedule. These entities travel along their own process paths, entering a decision module in which they access system variables to calculate an SPC point and determine whether an out-of-control action is needed (e.g. seizing the offending resource for unscheduled maintenance). Similar 'setup' entities can be created at the start of the simulation to monitor early performance and compute control chart limits if that part of the process is also to be modeled.

But difficulties remain. While the monitoring entities can be created, they i) may not be able to access the data needed for process execution monitoring, and ii) they may not be able to make the process changes that would be indicated by an out-of-control signal. Each of these difficulties are discussed in the following sections.

3.2.1 Analyzing Process Execution Data

Process-execution monitoring requires SIT data. Such data can be created during a simulation run and captured in state tables for many high-level simulation languages, although this may affect execution speed. While report features in many popular simulation languages permit writing SIT data to a file or recording it to a data structure, there is a significant difficulty in this approach to simulate various process execution monitoring scenarios. Writing data to a file is slow, and recovery for real-time analysis and decision making can greatly slow execution time of the simulation model. On the other hand, SIT data may be saved in a global variable array. Figure 4 shows how to create SIT data via an 'Assign' block in Arena, by creating a two-dimensional array with state as element (column) 1, ID as element 2, and time as element 3, and with one row of this array for each SIT read.

The difficulty comes in constructing sojourn time statistics from this array, which requires computing an average of differences in time-values for certain selected subsets of the SIT array. The subset selection may be complex because of the multiscale views that may be needed.

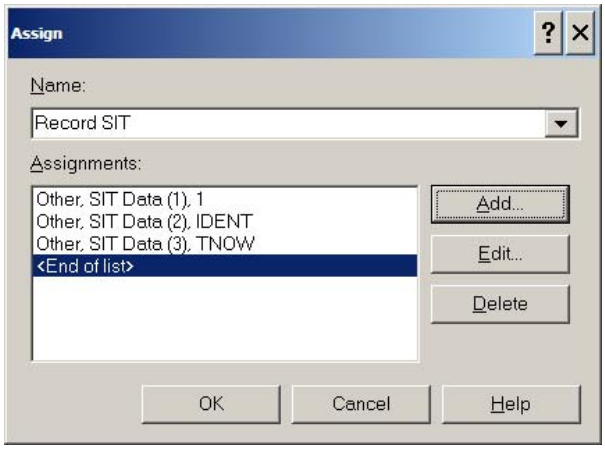

Figure 4: Recording SIT data in an SIT data array (Arena).

For example, consider computing the average sojourn time over a particular state subset $\mathbf{S}$, for all entities leaving this subset at time t within δ of a particular time t_0. This would allow the plotting of one point on a control chart similar to that in Figure 2 at time t_0. In this case, one would require finding:

1. each s_l, i_l, t_l triple in the SIT Data array with s_l a 'leaving' state for $\mathbf{S}$ and t_l within a neighborhood δ of t_0, and
2. the difference between t_l and, for the same i, the value t_e from the triple s_e, i_e, t_e where s_e is an 'entering' state for $\mathbf{S}$.

Such complex calculations require a real-time database-like query capability for examining the SIT array as the simulation is running.

A number of authors have addressed the difficulty of real-time process execution monitoring by working outside existing packages. For example, Low et al. (2007) developed a monitoring capability external to the simulation using IBM's WebSphere Web services package (IBM 2008). They call this approach symbiotic simulation. For process monitoring in particular, they use the WebSphere Business Monitor application (Leymann and Roller 2002).

3.2.2 Effecting Real-Time Structural Change

Structural change might be a process adjustment option, given the detection of a significant problem during process execution monitoring. For example, a particular work cell might contain both a drill press and a milling machine. If process execution monitoring indicates longer than expected setup times for the drill (because of failure or wear of parts, perhaps), then the appropriate action might be to have the drilling step temporarily taken over by the milling machine, changing both the routing (from 'drill, mill' to just 'mill') and the processing parameters (a change in distribution for the milling time to reflect fixturing and processing times for the milling and the former drilling operations).

Such real-time structural change within the high-level language and interface constructs of popular simulation packages would be a powerful capability. Real-time structural changes would be possible if the specification language for routings, resources and resource parameters permitted general expressions involving any global simulation variables.

Changes in response to system execution abnormalities might often affect only a single cell in a manufacturing operation, but the overall manufacturing operation might consist of many similar cells. This kind of structural change conflicts with object-oriented approaches that capture process flow at the object definition level (e.g., Pegden 2007). When one wants to change process flow for a single instantiation of a cell, not for all cells of that particular type, one can model each cell as an object without process inheritance from a parent object, or one must be able to modify an inherited property.

Real time simulation model structural change has been examined in the study of ambulance redeployment (Henderson 2008). In this case the stations correspond to ambulance bases, and the resources correspond to the ambulances assigned to a particular base. As ambulances respond to calls, other ambulances (either idle or returning from a previous call) may be redeployed to a different base. Thus the resource set associated with each station changes dynamically as the simulation runs, based on real-time decisions. The authors have chosen to build simulations from general-purpose programming languages, rather than use an existing simulation software package, to give flexibility in capturing such behavior (Henderson, Restrepo and Topalolu 2006).

4 CONCLUSION

Simulation has an important role to play in evaluating the effectiveness of different execution monitoring and adjustment schemes. Unfortunately, three of the six capabilities required to conduct such simulations (2, 4, and 6) can be difficult to perform in high-level simulation software environments.

Creating an SIT data structure with a structured query language for constructing evaluation expressions provides the ability to simulate a great variety of process execution monitoring methods. Pairing this with the ability to access and change resource lists, resource properties and routing tables as the simulation runs would permit modeling both process execution monitoring and the associated adjustment actions. This could be accomplished by allowing resource and state sequence definitions to use expressions with variables that can change as the simulation runs.

As business continues to focus on improvements in process execution, the importance of good process execution monitoring and adjustment schemes will grow. Simulation can play an important role in the evaluation of these schemes, and so we expect that the features described in this paper are likely to be developed where they do not already exist.

ACKNOWLEDGMENTS

This research was supported in part by National Science Foundation Grant DMI-0646687, and by a research grant from the Center for Supply Chain Research and the Smeal College of Business at Penn State. The authors also thank Shane Henderson, Barry Nelson and Dennis Pegden for helpful comments and discussion.

REFERENCES

Banks, J., J. S. Carson, B. L. Nelson, and D. M. Nicol. 2005. *Discrete-event system simulation*. 4th ed. Upper Saddle River, New Jersey: Prentice-Hall, Inc.

Barton, R. R. and J. Shu. 2008. An introduction to supply chain progress monitoring: key techniques and issues. Center for Supply Chain Research Working Paper, Smeal College of Business, Penn State University. Available at: <http://php2.smeal.psu.edu/cscr/>.

Bhuptani, M. and S. Moradpour. 2005. RFID field guide–deploying radio frequency identification systems. Blue Bell, PA: Sun Microsystems Press.

EPCglobal. 2004. The EPCglobal network and the global data synchronization network (GDSN): Understanding

the information and the information networks. Technical Report, EPCglobal Inc.

Henderson, S. G. (2008). Optimization modeling of health transportation. Available via: < www.chcm.ubc.ca/OPHC/documents/Henderson.ppt> [accessed 4/16/08].

Henderson, S. G., M. Restrepo, and H. Topalolu. 2006. Towards ambulance relocation via approximate dynamic programming. Technical Report, Cornell University, November 2006.

IBM. 2008. WebSphere business process management software: understand, define, execute and optimize. Available via: <www-306.ibm.com/software/websphere/> [accessed April 11, 2008].

Leymann, F. and D. Roller. 2002. Processes in a Web services world. Available via: <www-128.ibm.com/developerworks/ webservices/library/ws-bpelwp/ > [accessed April 11, 2008].

Lin, D. K. J., R. R. Barton, and The RFID Study Group. 2006. Challenges in RFID enabled supply chain management. *Quality Progress* 39: 23–28.

Low, M. Y. H., S. J. Turner, D. Ling, H. L. Peng, L. P. Chan, P. Lendermann, and S. Buckley. 2007. Symbiotic simulation for business process re-engineering in high-tech manufacturing and service networks. In *Proceeding of the 2007 Winter Simulation Conference*, ed. S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, 568-576. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Lurie, N., H. Jayashankar, and M. Swaminathan. 2006. Is timely information always better? The effect of feedback frequency on decision making. Contact: Dr. Nicholas H. Lurie at lurie@gatech.edu.

Pegden, C. D. 2007. SIMIO: a new simulation system based on intelligent objects. In *Proceeding of the 2007 Winter Simulation Conference*, ed. S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, 2293-2300. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Wheeler, D. J. and D. S. Chambers. 1992. *Understanding statistical process control*. 2nd ed. Knoxville, TN: SPC Press.

AUTHOR BIOGRAPHIES

RUSSELL R. BARTON is a professor in the Department of Supply Chain and Information Systems at Penn State University, co-director of the Penn State Master of Manufacturing Management degree program, and associate director of the Center for the Management of Technological and Organizational Change. He received a B.S. degree in Electrical Engineering from Princeton and M.S. and Ph.D. degrees in Operations Research from Cornell. Before entering academia, he spent twelve years in industry. He is Past President of the INFORMS Simulation Society and serves on the advisory board for the INFORMS Quality Statistics and Reliability section. He is a senior member of IIE and IEEE. His research interests include applications of statistical and simulation methods to system design and to product design, manufacturing and delivery. His email address is <rbarton@psu.edu>.

JUN SHU is an assistant professor in the Department of Supply Chain and Information Systems at Penn State University. He received a B.S. degree in Computer Science from UC Irvine and M.S. and Ph.D. degrees in Industrial Engineering and Operations Research from UC Berkeley. Prior to join Penn State, he worked as a systems engineer for MCI in its networking and telecommunication consulting division. He has served as a consultant for various companies including startups and established companies such as Rockwell Semiconductor Systems and Cisco. He focuses on boundary-spanning research that combines the knowledge from engineering, economics and business. His email address is <junshu@psu.edu>.