Grand Challenges in Modeling and Simulation of Complex Manufacturing Systems

John W. Fowler
Department of Industrial Engineering
Arizona State University
Tempe, AZ 85287-5906
john.folwer@asu.edu

Oliver Rose
Lehrstuhl für Informatik III
Am Hubland
D-97074 Würzburg
rose@informatik.uni-wuerzburg.de

Even though we have moved beyond the Industrial Age and into the Information Age, manufacturing remains an important part of the global economy. There is a need for the pervasive use of modeling and simulation for decision support in current and future manufacturing systems, and several challenges need to be addressed by the simulation community to realize this vision. First, an order of magnitude reduction in problem-solving cycles is needed. The second grand challenge is the development of real-time, simulation-based problem-solving capability. The third grand challenge is the need for true plug-and-play interoperability of simulations and supporting software. Finally, there is the biggest challenge facing modeling and simulation analysts today: that of convincing management to sponsor modeling and simulation projects instead of, or in addition to, more commonly used manufacturing system design and improvement methods such as lean manufacturing and six sigma.

Keywords: Modeling, simulation, manufacturing systems, real-time simulation, interoperability

1. Manufacturing Systems

Modern high-technology manufacturing systems, such as those in the electronics, semiconductor, aerospace, and automotive industries, can be extremely complex. The complexity of these systems is due to factors such as multiple part types made in the same facility/line, numerous manufacturing steps (300-500 steps is not uncommon), batch processing, very complex equipment that leads to high levels of preventive maintenance and downtime, and multiple levels of subassemblies, just to name a few. This complexity, combined with the high cost of setting up and maintaining such a system, necessitates the use of formal *models* of the system, rather than just relying on experience or simple rules of thumb for performance evaluation and decision making.

SIMULATION, Vol. 80, Issue 9, September 2004 469-476

DOI: 10.1177/0037549704044324

Models are intended to support management decisions about the system, and a single model often will not be capable of supporting all decisions. Rather, different decisions require different models because various aspects of the design and operation of the system will be important for the questions being asked of the model. While spreadsheet and queuing models are useful for answering basic questions about manufacturing systems, discrete event simulation models are often needed to answer detailed questions about how a complex manufacturing system will perform [1]. Simulation models lend themselves to incorporating additional details about the manufacturing system and therefore often give more accurate estimates of manufacturing system behavior than the simpler models mentioned above, but usually at the cost of more computation. In general, simulation is a practical methodology for understanding the high-level dynamics of a complex manufacturing system. According to Yücesan and Fowler [2], simulation has several strengths, including the following:

- Time compression—the potential to simulate years of real system operation in a much shorter time.

- Component integration—the ability to integrate complex system components to study their interactions.
- Risk avoidance—hypothetical or potentially dangerous systems can be studied without the financial or physical risks that may be involved in building and studying a real system.
- Physical scaling—the ability to study much larger or smaller versions of a system.
- Repeatability—the ability to study different systems in identical environments or the same system in different environments.
- Control—everything in a simulated environment can be precisely monitored and exactly controlled.

2. Grand Challenges

A grand challenge is a problem that (1) is difficult, with the solution requiring one or more orders-of-magnitude improvement in capability along one or more dimensions; (2) should not be provably insolvable; and (3) has a solution that results in a significant economical and/or social impact. We see three grand challenges in the modeling and simulation of complex manufacturing systems. Each of them will be discussed below, with the grandest challenge discussed first.

2.1 Grandest Challenge 1: An Order-of-Magnitude Reduction in Problem-Solving Cycles

It currently takes too long to design, collect information/data, build, execute, and analyze simulation models to support manufacturing decision making. This leads to a smaller number of analysis cycles than is desirable. While there are opportunities for efficiency improvements in all phases of the simulation process, we particularly see opportunities to reduce the time needed to collect and synthesize the required information and data, as well as opportunities to reduce the time to carry out the experimentation. The reduction in information/data collection and synthesis time can partially be achieved by proactive data analysis and by instilling better factory discipline in maintaining current information systems. The reduction in experimentation time can be approached from a number of different angles, including exploring models of reduced complexity that still give high-fidelity results, employing variance reduction techniques, and using distributed and parallel simulation. We discuss this further using the overall process for simulating manufacturing systems as a guide.

2.1.1 The Simulation Process for Manufacturing Systems Analysis

The process of simulating manufacturing systems involves the following phases and steps (modified from Chance, Robinson, and Fowler [1] and Yücesan and Fowler [2]):

A. Model Design

1. Identify the issues to be addressed.
2. Plan the project.
3. Develop the conceptual model.

B. Model Development

1. Choose a modeling approach.
2. Build and test the model.
3. Verify and validate the model.

C. Model Deployment

1. Experiment with the model.
2. Analyze the results.
3. Implement the results for decision making.

Opportunities for efficiency improvements for each phase will be discussed below.

2.1.2 Model Design

The model design phase is a very important but often overlooked part of the simulation process. In this phase, the project participants are identified, the project goals clearly delineated, and the basic project plan developed. Chance, Robinson, and Fowler [1] describe project management techniques that can significantly improve this portion of the process and the likelihood of having an overall successful project.

If these activities are not done well, the model developed will likely be a very detailed model. While incorporation of detail may increase the credibility of the model, excessive levels of detail may render a model hard to build, debug, understand, deploy, and maintain. The determination of how much detail to add to the model is a primary goal of the design stage. Experienced simulationists seem to instinctively know the proper amount of detail. Perhaps an expert system or at least a system that keeps track of previous analyses and the level of detail used could be developed to assist in the model design phase.

The issue of determining the appropriate level of detail becomes even more important when one models a manufacturing supply chain. While a lot of work has been done to model operations at the machine and factory levels, considerably less has been done at the supply chain level. For the most part, models are used at just one of these levels, and little information is shared between them. In the future, it will become increasingly necessary for these models to be used in conjunction with one another. To do this, several key questions remain: what is the right level of abstraction for each model? Can parallel and distributed simulation capabilities be employed? What is the right way to share information between the levels?

While manufacturing was one of the earliest simulation application areas [3], simulation of manufacturing supply chains is just starting to become widely used. Supply chains can be very complex, spanning multiple manufacturing sites in various locations around the globe. For simplicity, most researchers have limited the scope of their analyses to only a few broad categorical links in the chain and have taken an aggregate view of the supply chain, making various assumptions about the internal workings of each chain member. Supply chain analyses are often restricted to a single representative product.

Barker [4] indicates that it is the lack of models and frameworks for analyzing the value-adding capability throughout the supply chain that represents the greatest weakness in our supply chain knowledge base and current literature. The dearth of supply chain models is not industry specific but rather seems endemic to all industries due to a lack of overall comprehension of the intricacies of the supply chain.

Traditionally, simulation models of these systems have been either (1) a discrete event simulation model that tracks lots through each factory in the supply chain by considering the queuing at various work centers or (2) a high-level continuous simulation model that does not track individual lots through the factories but simply considers the gross output of each factory. The first approach can be quite accurate, but it generally takes a long time to build the model, and the execution of the model is extremely slow, making the exploration of many different scenarios prohibitive. Umeda and Jones [5] present an example of this type of model. Lendermann et al. [6] discuss a distributed simulation approach to detailed (high-fidelity) modeling of semiconductor manufacturing supply chains. This will be discussed further in the third grand challenge.

Models of the second type can generally be built fairly quickly and their execution is much faster, but a large amount of accuracy is generally lost. Heita [7] presents an example of this type of model. Jain et al. [8] examine the trade-off between these two approaches. Duarte et al. [9] discuss an effort that combines the two approaches by continuing to track lots through the factory but treating the daily output of the factory in a somewhat gross fashion. Preliminary results comparing the accuracy and speed of the approach to the traditional approaches are quite promising.

2.1.3 Model Development

After the preliminary work is done and the conceptual model designed, the next phase is to develop the model. This involves choosing the modeling approach, building the model, and doing verification and validation of the model. While there typically is not a lot of efficiency to be gained in choosing the modeling approach, the choice of approach can make a large difference in the subsequent model building and model execution times.

Most conventional simulation software packages used for modeling manufacturing systems take a “job-driven” worldview (also called a process interaction worldview). In this approach, manufacturing jobs are the active system entities while system resources, such as machines, are passive. The simulation model is created by describing how jobs move through their processing steps, seizing available resources whenever they are needed. A separate record for every job in the system is created and maintained for tracking wafers or lots through the factory. A lot of execution time can be consumed when sorting lists of these jobs in a given queue or in searching the queue for a given job. Therefore, the speed and space complexity of these simulations must be at least on the order of some polynomial of the number of jobs in the factory.

An alternative simulation methodology focuses on resource cycles (see Schruben and Roeder [10] for more details). In a “resource-driven” simulation, individual jobs are passive and are “moved” or “processed” by active system resources such as machines and operators. Rather than maintaining a record of every job in the system, only integer counts of the numbers of jobs of particular types at different steps are necessary. The state of the system is described by the status of resources and these job counts, and the execution speed and memory footprint does not change significantly as the system becomes more congested.

The events in a resource-driven simulation involve simple elementary integer operations, typically incrementing or decrementing job counts and numbers of available resources. Very large and highly congested queuing networks can be modeled this way with a relatively small, finite set of integers. In addition to their simplicity, resource-driven simulations have a number of advantages over conventional job-driven process flow models that describe the paths of individual jobs. Resource-driven simulations of highly congested systems have been created that execute orders of magnitude faster than corresponding job-driven process flow simulations [10].

The hope is that resource-driven factory simulations can be developed that are able to provide much of the same information as job-driven simulators while executing many times faster. The expectation is that the two types of factory models can be used together. For example, high-speed, resource-driven factory simulators would be used for large-scale experiments that identify key opportunities for improvement. These opportunities can then be studied with detailed job-driven simulators.

After determining the modeling approach, the next step is to build the model and to determine the appropriate distributions to use for various system elements. Model building can be very time-consuming and is an area where there potentially is room for significant efficiency improvements.

There is a rich history of efforts to improve the efficiency of the model-building process, with much of this work led by the software community. In the early days of computing, most simulations of manufacturing systems used an event orientation and were written in assembly language or a high-level programming language such as FORTRAN. This led to models that were relatively efficient to execute because only the events desired were modeled. However, each new model development effort led to new model code being written (i.e., there was little reusability of code).

In the next wave, simulation languages were developed to speed up the modeling process by allowing reuse of simulation constructs. These languages included GPSS, SIMSCRIPT, GASP, SLAM, SIMAN, and so on. In these languages, very basic logic/functionality was included in each construct. In addition, simulation overhead activities, such as managing the event calendar, generating (pseudo)-random numbers and random variates, and collecting statistics, were included as a part of the package. The academic community played a major role in determining efficient ways to accomplish the simulation overhead activities.

The widespread availability of microcomputers in the early to mid-1980s led to an increased use of simulation to model manufacturing systems. It also led the simulation software community to develop two related but somewhat different approaches to making model building more efficient. The first of these was the development of user interfaces that allow multiple base constructs to be combined into higher level constructs such as QUEUE-SEIZE_RESOURCE-DELAY-FREE_RESOURCE to model a workstation. ARENA was perhaps the most successful of these systems. The second approach was the development of simulators. In a simulator, the model is already built, and the user simply supplies the appropriate data. Obviously, simulators can save considerable model-building time if the underlying model of the simulator can adequately model the system being studied. AutoSchedAP and Factory Explorer are two simulators that are used to model semiconductor manufacturing systems. A nice overview of the history of simulation language development can be found in Nance [11]. While all of the developments described above have significantly reduced the time to build some models, there is still considerable room for improvement.

One potential opportunity for improvement is in using existing sources of data available about a particular manufacturing system. Indeed, some analysts have done this to some degree already. In many industries, it is very common for the process-routing information to be kept electronically in the manufacturing execution system (MES). This information can sometimes be extracted to populate a simulator, or a simulation model could be built automatically from the data using model generation techniques. Ozdemirel, Mackulak, and Cochran [12] developed an expert system called MASCOT that generated a model based on these data and the system performances measures of interest. Existing data sources can also be used to determine the appropriate distribution to use for random processes such as interarrival times, processing times, and so forth. Today, when data from the actual system are available, this is typically done manually using software packages such as BestFit [13] or ExpertFit [14]. However, it may be possible to also automate this activity. Both the automatic generation of the simulation model and the automated determination of distributions should provide significant speed-up possibilities in the overall time required to build a simulation model.

The automatic generation of the simulation model should also reduce the time needed to verify a model by reducing the time required to debug the code. The time required to perform verification for some manufacturing systems has been reduced by the use of animation. Validation, on the other hand, generally requires expert opinion, so there is probably not a lot that can be done to reduce this time other than to make sure that the output of the model is in a format that facilitates understanding the performance of the model.

2.1.4 Model Deployment

In the model deployment phase, the main area for efficiency improvement is in executing the model. While simulations of some manufacturing systems may not take much time to run, models of complex manufacturing systems may take several hours for a single replication, particularly when details of automated material handling systems (AMHS) are included. Shikalgar, Fronckowiak, and MacNair [15] indicate that the average time for a 160-day simulation run of an IBM wafer fabrication facility with AMHS is approximately 24 hours, but removing the AMHS system may reduce this to 3 to 4 hours. Mercier, Bonnin, and Vialletelle [16] report that a single replication of their wafer fabrication model takes 1 to 2 hours of CPU time. Anecdotal evidence suggests that these runtimes are not uncommon for models of wafer fabrication facilities. Clearly, simulation runtimes of the magnitudes mentioned above limit the number of problem-solving cycles that can be achieved. In fact, the IMTI Integrated Manufacturing Technology Roadmapping Project [17] report on modeling and simulation identifies that a key goal is the need for "fast, accurate exploration of many more product and process design options" and the "requirement for simulation techniques that enable complex simulations to run orders of magnitude faster and more cost effectively than today."

As mentioned above, the modeling approach (job driven vs. resource driven) may make a difference here. In addition, the use of variance reduction techniques (VRTs) offers some potential savings (see Law and Kelton [18, chap. 11] for more information on VRTs). While these techniques have successfully been used in some manufacturing simulations, we believe their potential has not been fully exploited. Specifically, the automated use of common random numbers and antithetic variates could be better supported by simulation packages. Finally, parallel simulation offers some hope for reduced simulation execution times

[19], particularly when modeling the manufacturing supply chain.

The time to analyze the results of the simulation can be shortened by automatic generation of graphs, charts, confidence intervals, paired *t*-tests, and so forth. However, ultimately, human judgment is critical, and this part of the analysis process probably does not provide an opportunity for significant time savings.

2.2 Emerging Grand Challenge 2: Development of Real-Time Simulation-Based Problem-Solving Capability

Currently, most simulation models are used in single projects for tactical or strategic decision support (i.e., with a long time horizon). Often, simulation models are used to plan equipment purchases or to evaluate planned changes of the material flow control. To build these models from scratch requires considerable effort. It is not uncommon that the models are not used again after the decision is made. Even when this is the case, the time and money spent on these traditional simulation projects leads, in most cases, to a large return on investment [1].

Relatively little is known, however, about approaches to use simulation for operational (real-time) decisions in manufacturing. Simulation models for this purpose are possible today due to the increased amount of data and information that is collected and maintained by current shop-floor information systems. The literature on real-time applications of simulation in manufacturing is sparse, but there are papers about simulation-based scheduling [20-22], order release [23], forecasting [10], and exception management [24]. In Peters et al. [21], the focus is on the requirements for an online simulation tool, whereas Rogers and Gordon [22] provide an overview of the principal problems of simulation-based scheduling.

In this context, real-time, simulation-based problem-solving capability means that if the status of the factory changes abruptly, we can run a simulation nearly instantaneously to decide about appropriate actions to be taken. For example, when a key piece of manufacturing equipment fails, plans for minimizing the impact of the failure need to be developed. Simulations of various recovery alternatives could help determine the best course of action. Another example deals with batch operations in wafer fabrication. In this case, when there is a partial batch available to begin processing, a decision needs to be made whether to start the partial batch now or to wait for future arrivals. Fowler, Hogg, and Phillips [25] describe non-simulation-based techniques that can be helpful in making this decision, but the capability to generate a set of simulations to see what might happen for either decision (start now or wait) could be extremely beneficial since these decisions have a major influence on the performance of the system.

Due to the need for a quick response to the types of problems described above, model-building and data collection times must be very short. In particular, the simulation-based solution time requirements for the operational problem are considerably more aggressive (shorter) than for classical approaches. However, providing accurate results is still important for the results to be accepted. Because of this time challenge, we see two ways to realize real-time simulation capabilities:

- use of a simulation model that is permanently running and synchronized to the factory,
- automated building of a model from the factory databases.

2.2.1 Permanent, Always-On, Synchronized Factory Models

These models mimic the behavior of the real factory. The models are continuously updated and synchronized with factory data. If an additional model is required for decision making, a clone is generated from the prototype model. The main problem of this approach is the availability of on-time and correct data from the factory. Although announced for years, all-in-one factory databases do not exist yet. Due to database infrastructures that grew (evolved) over several years and the fact that the single databases are implemented in different (sometimes incompatible) software products, it requires considerable effort to obtain up-to-date, consistent, and useful data. The different database standards and access techniques lead to a wide variety of interfaces between simulation software and factory databases. Often, these databases are not physically in one location, and data transfer times of several minutes may arise for updating the factory model. Even if the interface issues between simulator and factory databases are solved, the updates of the databases from the factory workstations and MES have to be considered. The data collection, processing, and transfer capabilities of the tools range from very basic to highly sophisticated. Some only report machine states, while others provide all types of information a simulation model needs. As a result, the following problems have to be solved:

- A clear definition of the term *factory state* is missing: before we can start to collect data, we have to know exactly what kind of data we require to obtain a clear picture of the factory.
- There is a lack of data: data required by the simulation are not available and cannot be generated in an automated way from other factory information.
- The data quality is too low: for example, the simulation model requires a histogram, and the tool can only provide average values of a parameter of interest.
- The update frequency is too low: the tool can only generate data reports after a given and long time interval (say, once a day).

It depends on the objective of the simulation-based problem solver whether these problems have an effect on the results. For instance, if only a coarse overview on the factory behavior for the next few days is required, the problems have only a minor impact on the results. If, however, the intention is to provide a simulation-based scheduling system, the factory data have to be very accurate, and all of the issues above must be solved, or there must be at least an estimation of the error in the schedule induced by wrong, missing, or late data.

To mimic the real factory's behavior, the simulation model must have appropriate MES functionality. This can be achieved either by using a copy of the real MES with interfaces to the simulator or by rebuilding the MES inside the simulator. The second approach is time-consuming and error prone because most MES software companies will not be willing to provide detailed information about the implementation of their products. Therefore, the rebuilt MES will only behave approximately like the real one.

If all data problems are solved, we have to find a simulation package that supports real-time capabilities, such as Arena RT from Rockwell Software. At the moment, however, there is a lack of off-the-shelf software that can be used for large-scale manufacturing simulation models. During recent years, more effort has been spent trying to speed up simulation engines than to try to run them in real time. In addition, the models must have the ability to be synchronized with the factory state. The factory state not only summarizes tool states and lot positions but also must comprehend adding or removing equipment. The persistent, constantly synchronized factory simulation model is the master copy for all simulation models required for decision making. As soon as an operational problem arises, a clone is created from the master copy, and the analysis is only based on this clone. At the moment, there does not appear to be a simulation package that provides this feature.

2.2.2 On-Demand, Automatically Built Factory Models

Taking into account the problems discussed in the previous section, particularly the lack of real-time simulation software packages, it is worthwhile to consider a different approach. Instead of generating clones from a factory simulation model that has to be perpetually synchronized, the experimenter generates the model on demand directly from the factory databases. In this case, the requirements for the factory databases remain the same, but the simulation software can be far less sophisticated. Most of the simulation packages on the market today facilitate building models from formatted text files. Thus, we only need an application that retrieves the current factory data from the databases and then transforms them into simulation model files. Because considerable amounts of factory data, including tool sets and their properties, current and future orders, product specifications and routing, and so forth, have to be retrieved and transferred, it may take considerable time to finish the model-building process. As a consequence, this approach is less suited for time-critical decisions in which the simulation results have to be available within minutes or seconds after the problem occurred. In that case, the first approach is more appropriate because cloning from a master copy can be done in seconds. To simplify and speed up the model generation, the required data should not be stored in the databases in raw format but preprocessed and ready to use for the simulation package. For instance, it causes unnecessary delays if machine availability histograms or distributions have to be computed when the model is being built. This can be done proactively right after each database update from the factory.

2.3 Emerging Grand Challenge 3: True Plug-and-Play Interoperability of Simulations and Supporting Software within a Specific Application Domain

If we begin to have persistent models of the manufacturing system, it is likely that there will actually be models of many different subsets of the factory. These models will need a seamless way to interact with each other. In addition, more and more of the information/data will be provided from other manufacturing support software such as the MES, available-to-promise systems, analysis software packages, and so on. It will be increasingly more important for all of these systems to be able to quickly communicate with each other and the outside world in an unambiguous way. The High-Level Architecture (HLA) is one partial solution to this challenge.

The HLA is a general-purpose architecture for simulation reuse and interoperability. In the HLA, each simulation or other software system is run as a separate federate (process), and the collection of all federates is called a federation. Each federate can be developed independently and implemented using different software languages and different hardware platforms. Thus, faster cycles of analysis may occur due to less time spent developing new integrated software and less execution time due to the distribution of the software on multiple processors. The HLA was developed under the leadership of the U.S. Defense Modeling and Simulation Office (DMSO) to support reuse and interoperability across a wide range of different types of simulations. The HLA Baseline Definition was completed in 1996 and was adopted as the Facility for Distributed Simulation Systems 1.0 by the Object Management Group (OMG) in 1998. The HLA was approved as an open standard through the Institute of Electrical and Electronic Engineers (IEEE)—IEEE Standard 1516—in 2000. Unfortunately, the software community has been slow to incorporate the HLA in their product offerings. This is due primarily to lack of pull from their users, which is at least in part due to a lack of understanding what the HLA can provide.

However, some users have applied the HLA to study their manufacturing systems. Schumann et al. [26] present an example of using the HLA for factory simulation. Lendermann et al. [6] discuss a distributed simulation prototype of a semiconductor manufacturing supply chain based on the HLA.

Straßburger et al. [27] discuss the fact that while the HLA is helpful in the application of the distributed simulation of the "digital factory," it is not sufficient. Specifically, they point out that it is unfortunate that (1) there are no time-managed versions, (2) there is not a way to transfer an object/attribute to a specific receipt, and (3) there is no possibility of transferring the final state of an object regarding attribute values. They refer to papers by Fujimoto and Tacic [28], Sauerborn et al. [29], and Myjak et al. [30] that study these deficiencies but indicate that no solutions have been implemented in the HLA interface specification. While it is not clear if the HLA will ultimately provide (or be a major part of providing) true plug-and-play interoperability, it is clear that this functionality is needed as analysts strive to adequately model supply chain operations using the logic embedded in the software used in the real system.

3. Big Challenge 4: Greater Acceptance of Modeling and Simulation within Industry

We decided to label this challenge as a "big" challenge instead of as a "grand" challenge because it is not really a technical challenge but more of a social challenge. In fact, this may make this the most difficult challenge of all the challenges discussed in this paper. While the use of modeling and simulation in manufacturing is steadily gaining acceptance for certain applications (such as capacity planning), there is still a long way to go before it is commonly applied for a multitude of other applications. Currently, modelers often spend much of their time convincing management of the need for these services. Simulation is generally only one of several manufacturing system design and improvement approaches that are presented to management for possible implementation. Other approaches include lean manufacturing, six sigma, just-in-time manufacturing, total quality management, and so on, and the simulationist should not try to convince management that simulation is better than these techniques. Indeed, simulation, by itself, does not improve the performance of a manufacturing system. It is only by the use of the model to answer specific questions about ways to change the system that realizable improvements are identified. Thus, simulationists should try to convince management that simulation is complementary to the other approaches mentioned above and that it can be used to assess the potential improvements that can be made to the system when the other approaches are employed. Finally, simulationists must be careful to resist the temptation to oversell the use of a model's results; this may be a good short-term strategy, but it can have very negative long-term consequences if the expectations of the users of the model results are not met. Instead, simulationists should indicate that while a particular model could be extended to help to answer numerous questions, including those that it was not originally designed for, these questions should be tackled in a systematic and planned manner.

4. References

[1] Chance, F., J. Robinson, and J. Fowler. 1996. Supporting manufacturing with simulation: Model design, development, and deployment. In *Proceedings of the 1996 Winter Simulation Conference*, San Diego, CA, pp. 1-8.

[2] Yücesan, E., and J. Fowler. 2000. Simulation analysis of manufacturing and logistics systems. In *Encyclopedia of production and manufacturing management*, edited by P. Swamidass, 687-97. Boston: Kluwer Academic.

[3] Naylor, T. H., J. L. Balintfy, D. S. Burdick, and K. Chu. 1996. *Computer simulation techniques*. New York: John Wiley.

[4] Barker, R. C. 1996. Value chain development: An account of some implementation problems. *International Journal of Operations & Production Management* 16 (10): 23-36.

[5] Umeda, S., and A. Jones. 1998. An integration test-bed system for supply chain management. In *Proceedings of the 1998 Winter Simulation Conference*, pp. 1377-85.

[6] Lendermann, P., N. Julka, B. P. Gan, D. Chen, L. F. McGinnis, and J. P. McGinnis. 2003. Distributed supply chain simulation as a decision support tool for the semiconductor industry. *Transactions of the Society for Modeling and Simulation International* 79 (3): 126-38.

[7] Heita, S. 1998. Supply chain simulation with LOGSIM-simulator. In *Proceedings of the 1998 Winter Simulation Conference*, pp. 323-26.

[8] Jain, S., C. Lim, B. Gan, and Y. Low. 1999. Criticality of detailed modeling in semiconductor supply chain simulation. In *Proceedings of the 1999 Winter Simulation Conference*, pp. 888-96.

[9] Duarte, B. M., J. W. Fowler, K. Knutson, E. Gel, and D. Shunk. 2002. Parameterization of fast and accurate simulations for complex supply networks. In *Proceedings of the 2002 Winter Simulation Conference*, pp. 1327- 36.

[10] Schruben, L., and T. Roeder. 2003. Fast simulations of large-scale highly congested systems. *Transactions of the Society for Modeling and Simulation International* 79 (3): 115-25.

[11] Nance, R. E. 1993. A history of discrete event simulation programming languages. *Proceedings of the Second ACM SIGPLAN History of Programming Languages Conference* 23 (3): 149-75.

[12] Ozdemirel, N. E., G. T. Mackulak, and J. K. Cochran. 1993. A group technology classification and coding scheme for discrete manufacturing simulation models. *International Journal of Production Research* 33 (3): 579-601.

[13] Jankauskas, L., and S. McLafferty. 1996. BESTFIT, distribution fitting software by Palisade Corporation. In *Proceedings of the 1996 Winter Simulation Conference*, pp. 551-55.

[14] Law, A. M., and M. G. McComas. 1989. Pitfalls to avoid in the simulation of manufacturing systems. *Industrial Engineering* 31:28-31.

[15] Shikalgar, S. T., D. Fronckowiak, and E. A. MacNair. 2002. 300mm wafer fabrication line simulation model. In *Proceedings of the 2002 Winter Simulation Conference*, pp. 1365-68.

[16] Mercier, D., O. Bonnin, and P. Vialletelle. 2003. Achieving optimal cycle time improvement in a 300mm semiconductor fab using dynamic simulation and design of experiements. *Transactions of the Society for Modeling and Simulation International* 79 (3): 171-79.

[17] Integrated Manufacturing Technology Initiative. 2000. *Integrated Manufacturing Technology Roadmapping Project: Modeling & Simulation*. Integrated Manufacturing Technology Initiative (IMTI): Oak Ridge, USA.

[18] Law, A. M., and D. W. Kelton. 2000. *Simulation modeling and analysis*. 3d ed. New York: McGraw-Hill.

[19] Low, M. Y. H. 2002. Manufacturing simulation using BSP time warp with variable number of processors. In *Proceedings of the 14th European Simulation Symposium*.

[20] Chong, S., A. Sivakumar, and R. Gay. 2002. Design, development and application of an object oriented simulation toolkit for real-time semiconductor manufacturing scheduling. In *Proceedings of the 2002 Winter Simulation Conference*, pp. 1849-56.

[21] Peters, B., J. Smith, J. Curry, C. LaJimodiere, and G. Drake. 1996. Advanced tutorial: Simulation-based scheduling and control. In *Proceedings of the 1996 Winter Simulation Conference*, pp. 194-98.

[22] Rogers, P., and R. Gordon. 1993. Simulation for real-time decision making in manufacturing systems. In *Proceedings of the 1993 Winter Simulation Conference*, pp. 866-74.

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

[24] Katz, D., and S. Manivannan. 1993. Exception management on a shop floor using online simulation. In *Proceedings of the 1993 Winter Simulation Conference*, pp. 888-96.

[25] Fowler, J. W., G. L. Hogg, and D. T. Phillips. 2000. Control of multiproduct bulk server diffusion/oxidation processes part two: Multiple servers. *IIE Transactions on Scheduling and Logistics* 32 (2): 167-76.

[26] Schumann, M., E. Blümel, T. Schulze, S. Straßburger, and K.- C. Ritter. 1998. Using HLA for factory simulation. In *Proceedings of the 1998 Fall Simulation Interoperability Workshop*, September, Orlando, FL.

[27] Straßburger, S., A. Hamm, G. Schmidgall, and S. Haasis. 2002. Using HLA ownership management in distributed material flow simulations. In *Proceedings of the 2002 European Simulation Interoperability Workshop*, June, London.

[28] Fujimoto, R., and I. Tacic. 1999. Time management of unsynchronized HLA services. Paper no. 99F-SIW-165. In *1999 Fall Simulation Interoperability Workshop*, September.

[29] Sauerborn, B., G. Tan, G. Moss, P. Oxenberg, F. Moradi, and R. Ayani. 2000. HLA ownership management services: We almost got it right. Paper no. 00F-SIW-076. In *2000 Fall Simulation Interoperability Workshop*, September.

[30] Myjak, M., S. Sharp, T. Lake, and K. Briggs. 1999. Object transfer in HLA. Paper no. 99S-SIW-140. In *1999 Spring Simulation Interoperability Workshop*, March.

John W. Fowler *is Professor of Industrial Engineering at Arizona State University and is the Center Director for the Factory Operations Research Center that is jointly funded by the National Science Foundation, International SEMATECH and the Semiconductor Research Corporation. He received his PhD from Texas A&M University and was previously a Senior Member of Technical Staff in the Modeling, CAD, and Statistical Methods Division of SEMATECH. His research interests include modeling, analysis, and control of manufacturing (especially semiconductor) systems. He is a member of ASEE, IEEE, IIE, INFORMS, POMS, and SCSI. His web address is http://www.eas.asu.edu/∼masmlab/fowler/.*

Oliver Rose *is Professor for Modeling and Simulation in the Department of Computer Science at the Dresden University of Technology, Germany. He received an MS degree in applied mathematics ('92) and a PhD degree in computer science ('97) from the University of Würzburg, Germany. His research focuses on the operational modeling, simulation and analysis of complex manufacturing systems with applications in the semiconductor industry. He is a member of IEEE, INFORMS College on Simulation, and ASIM. His web address is www.iai.inf.tu-dresden.de/ms/.*