# DISTRIBUTED SUPPLY CHAIN SIMULATION
## ACROSS ENTERPRISE BOUNDARIES

Boon Ping Gan
Li Liu
Sanjay Jain

Stephen J. Turner
Wentong Cai
Wen-Jing Hsu

Gintic Institute of Manufacturing Technology
71 Nanyang Drive
SINGAPORE 638075

School of Computer Engineering
Nanyang Technological University
Nanyang Avenue
SINGAPORE 639798

## ABSTRACT

The effective practice of supply chain management (SCM) is crucial to improve corporations' competitive advantage. Many corporations have built simulation models to facilitate the application of simulation in designing, evaluating, and optimizing their supply chain. Traditionally, a supply chain involves only a single enterprise with multiple facilities and distribution centers. Hence, sharing of detailed simulation models is not a problem in this scenario. But in recent years, the scope of SCM has evolved to cross the enterprise boundaries. Applying simulation in designing, evaluating, and optimizing the supply chain becomes more difficult since the participating corporations might not be willing to share their simulation models with partners. In this paper, distributed simulation techniques are presented as an enabling technology that allows corporations to construct a cross enterprise simulation while hiding model details within the enterprise. This can be realized by either building the simulation on top of the Runtime Infrastructure of the High Level Architecture or building the simulation on top of a customized distributed discrete event simulation protocol. These alternative approaches are compared in terms of their performance and interoperability. The comparison of the performance is done through a benchmarking test of a semiconductor supply chain model.

## 1 INTRODUCTION

Supply chain management (SCM) involves managing the flow of material and information through multiple stages of manufacturing, transportation and distribution with the objective of maintaining low inventories without compromising customer service level. The effective practice of SCM is critical to participating companies especially in today's business trend whereby companies are geographically distributed throughout the globe. Traditionally, SCM involves only a single enterprise with multiple facilities and distribution centers. But in recent years, the scope of SCM has evolved to cross the enterprise boundaries, as vertical integration is no longer the emphasis of large corporations (Archibald, Karabakal and Karlsson 1999).

Commercial simulation tools for supply chain planning have been released in recent years, for example the Supply chain Analyzer by IBM (Archibald, Karabakal and Karlsson 1999) and the integrated tools of simulation and optimization by i2 (Padmos et al. 1999). This illustrates the importance and applicability of simulation to supply chain planning. The main area of application of supply chain simulation has been on performing *what-if* analysis, by varying various aspects of the chain. To draw accurate conclusions, it has been shown that a detailed model of the whole chain needs to be built (Jain et al. 1999). Building a detailed model of the supply chain does not pose a problem when the chain involves only a single enterprise. In contrast, not many participating companies are willing to share detailed model information when the chain crosses the enterprise boundaries. This obstructs the use of simulation in supply chain planning.

In this paper, we discuss distributed simulation techniques as the enabling technology to eliminate this obstacle. Distributed simulation technology allows each participating corporation to run their own simulation model at their own site. Detailed model (application codes and data) information is encapsulated within the corporation itself and the participating corporations only need to define essential information flows from one model to another. To realize the model encapsulation, two approaches are discussed here. One is based on the High Level Architecture (HLA) Run Time Infrastructure (RTI) (Kuhl, Weatherly and Dahmann 1999) while another is based on an extended asynchronous simulation protocol (Gan and Turner 2000) that is implemented using the Message

Passing Interface (MPI) (Gropp, Lusk and Skjellum 1994). MPI is a library of functions and macros that facilitates the development of parallel programs running on shared and distributed multiprocessor systems. The latter approach is referred to as MPI-ASP hereafter.

This paper is organized as follows: Section 2 discusses how distributed simulation technology can serve as an *enabler* to achieve model encapsulation. HLA and MPI-ASP are then compared together with a detailed coverage of how the simulation can be built on top of these two technologies. A supply chain model based on the semiconductor industry is then presented in Section 3. Section 4 presents the benchmarking test of the HLA RTI and MPI-ASP. Finally, conclusions are drawn in Section 5.

## 2 DISTRIBUTED SUPPLY CHAIN SIMULATION

### 2.1 Distributed Simulation for Model Encapsulation

Traditionally, the main objective of parallel (and distributed) discrete event simulation (PDES) technology has been to improve the execution time of complex and time-consuming simulations. Another important aspect of this technology, which does not gain as much attention, is its ability to hide the modeling details of the simulation, which is referred to as model encapsulation here. Using this technology, the overall simulation system is broken down into component models that are able to run independently. Each model executes without knowing the details of what others are doing. The only interaction among the component models are the messages that they send to each other. These messages have a predefined format according to the information that independent models are going to share. Synchronization is achieved through the use of a parallel discrete event simulation protocol. We refer to this as a PDES protocol as it may be used on both parallel and distributed computer systems.

In a supply chain simulation, each company can be seen as an independent model, as described in the previous paragraph. The modeling details of each company (model) are then encapsulated within the company itself. In order to simulate the supply chain, participating companies need to define the data that are going to be shared. Shared data is then exchanged as messages during the simulation run. Messages can be transmitted through a network connecting the participating companies. As discussed earlier, the simulation time synchronization of these independent models can be achieved by the application of a PDES protocol. From this discussion, one would see that PDES offers the advantage of encapsulating the detailed model information while still allowing a cross enterprise boundaries supply chain simulation.

In this paper, we discuss two alternative approaches to realizing this objective. One is through the application of a readily available distributed simulation engine, namely the

Runtime Infrastructure (RTI) of the HLA. The High Level Architecture (HLA) is a common framework within which specific simulation system architectures can be defined. It allows users to describe the objects to be simulated and the interactions among these objects in a standard way. HLA also defines an interface specification which is implemented by the RTI, to provide standard application interfaces (API). These interfaces are used to facilitate the integration and execution of separate simulation models.

The alternative approach is to develop a tailor-made distributed simulation engine, implementing some well-known PDES protocol. In this paper, we extend our parallel simulation engine, which was developed originally for shared memory multiprocessor systems, for distributed simulation. It is based on an extended asynchronous simulation protocol (Gan and Turner 2000). The algorithm was modified by incorporating the MPI library for message passing.

### 2.2 HLA vs. MPI-ASP

Distributed simulation based on HLA and MPI-ASP is compared in two ways. One is interoperability while another is performance. Interoperability covers several aspects:

a) whether the approach provides any standard ways of defining the model and shared data,
b) whether the approach provides standard application interfaces (API) that separate the applications from the underlying simulation kernel,
c) whether the approach facilitates "plug-and-simulate" of models built using different simulation languages,
d) whether the approach facilitates reuse of models.

HLA has an edge over MPI-ASP in terms of interoperability. It has a well-defined Object Model Template (OMT) that facilitates users to describe the component models to be simulated and the interactions among component models. More specifically, the data produced or consumed by a component model is defined by its Simulation Object Model (SOM). It also provides an interface specification that defines standard application interfaces (API) to the RTI simulation kernel, without restricting the implementation of the kernel. This offers user the flexibility of building their simulation using different simulation languages. With this flexibility, companies that are involved in the supply chain simulation can develop their model using any simulation language (the simulation language must have an HLA API) that is most appropriate for their model. Simulation models developed can then be integrated seamlessly together ("plug-and-simulate" capability), provided they are using the same RTI implementation. In addition to the flexibility

of different simulation languages, HLA also eases the reuse of model. See Kuhl, Weatherly and Dahmann (1999) for more details.

The interoperability advantages offered by HLA are possible to be realized using MPI-ASP, but not without major effort. Even though MPI-ASP does provide a clear logical separation of the simulation kernel and the application model, it still lacks standard APIs that interface to the simulation kernel. It also lacks a standard way of defining shared data among the component models of a simulation system. Once these interfaces are available, "plug-and-simulate" and reuse of model using MPI-ASP will be easier to realize. One point to consider is that even though MPI-ASP can offer similar capabilities to HLA, it still loses out to HLA since HLA has been nominated as an IEEE standard and has been adopted by the Object Management Group as an industry standard.

Interoperability aside, it is intuitive that a simulation running on top of the MPI-ASP simulation engine should be able to achieve a better performance (in terms of overall execution time) as compared to a simulation that is running on top of the HLA's RTI. This claim is verified through our benchmarking test, using our own supply chain model, in a later section.

These alternative approaches share one common advantage of allowing a modular model development strategy to be applied. Individual companies can develop their models independently of each other. Models can also be developed in a modular approach within the company. This advantage can significantly shorten the model development time.

## 2.3 HLA for Distributed Supply Chain Simulation

In HLA, each individual model is a federate. A collection of federates that form the whole simulation system is a federation. To apply this to a supply chain simulation, the federate is thus the basic simulation model of each individual company. Each company defines only data that they are willing to share in the SOM using the OMT of the HLA. Details of HLA's object model template can be found in Lutz (1998). Sensitive information on the model are hidden within the company.

The simulation time synchronization of federates is achieved automatically through the time management services of HLA (Fujimoto 1998). Figure 1 shows the main simulation loop that is common to all federates in the federation. The pre-requisite to using the time management services of HLA is that each federate must define a non-zero lookahead value (line 1). In HLA, lookahead is associated with a federate. It is a value which determines the next earliest time that the federate will send an external event. Before a federate executes each event, the simulation time of this event has to be checked against the

time granted by the RTI earlier. If the simulation time is not greater than that previously granted, the event is then simulated. Otherwise, the federate needs to request for time advancement from the RTI. This is done by passing control back to the RTI, using the call to *tick*() (line 10). When the *tick*() is called, the RTI also tries to receive all the events from remote federates. The RTI will grant the requested time only when it knows that the requesting federate will not receive any events with timestamp less than the requested value in the future. One important point in Figure 1 is that the *topEv* (event on top of the event list) in line 3 and 13 could be two different events. While a federate is waiting for the RTI to grant the time request (line 10), it is possible that the federate has received some remote events (interactions) with timestamp smaller than the event currently on top of the event list. These new events are inserted into the federate's local event list. More details on how HLA is adapted to the supply chain simulation can be found in Turner, Cai and Gan (2000).

```
1  init federate's lookahead
2  while (not end of simulation)
3   topEv = eventList.top()
4   ; whereby eventList is an ordered list
5   ; sorted by events' simulation time
6   evTS = topEv.get_timeStamp()
7   if (evTS > grantedTime)
8    timeAdvanceRequest(evTS)
9    while (request not granted by RTI)
10     tick()
11    end while
12   end if
13   topEv = eventList.pop()
14   simulate topEv
15   foreach generated event, e_g
16    if e_g is a local event
17     eventList.insert(e_g)
18    else
19     send e_g as interaction
20    end if
21   end for
22  end while
```

Figure 1: Main Simulation Loop for Simulator based on HLA

## 2.4 MPI-ASP for Distributed Supply Chain Simulation

As discussed earlier, distributed supply chain simulation can also be realized by customizing a PDES protocol for this purpose. In PDES, each simulation model is mapped to a logical process (LP). A collection of logical processes can then form the whole simulation system. To apply this to the supply chain simulation scenario, each LP thus represents the model of a company. Similar to the HLA approach, these models can then be implemented independently. Participating companies only need to define data that are going to be shared and the message format with which this data will be transmitted to one another.

The PDES protocol that is used to realize distributed supply chain simulation in this work is an extended asynchronous simulation protocol (Gan and Turner 2000), MPI-ASP. It incorporates the idea of null messages (Chandy and Misra 1979) to ensure that the simulation is deadlock free. Null messages carry time guarantee information which represents the next possible earliest simulation time that an external event will be sent from an LP to its immediate downstream LP. It is computed by adding the lookahead between the LPs to the current simulation time of the source LP. Figure 2 shows the general outline of the distributed simulation protocol. Safetime in Figure 2 is an upper limit to which an LP can advance its simulation time.

```
1  while (not end of simulation)
2   Recv external events and null messages
3   safeTime = min(timeGuaranteed_{k,i})
4     where k is list of upstream LPs for LP_i
5   while (eventList.top().get_timeStamp()
6             < safeTime)
7    topEv = eventList.pop()
8    simulTime = topEv.get_timeStamp()
9    simulate topEv
10   foreach generated event, e_g
11    if e_g is a local event
12      eventList.insert(e_g)
13    else
14     timeGuaranteed_{i,d} = simulTime
15                    + lookahead_{i,d}
16       where LP_d is destination LP of e_g
17     send e_g as message with piggybacked
18         timeGuaranteed_{i,j}
19    end if
20   end for
21  end while
22  simulTime = safeTime
23  old_timeGuaranteed_{i,j} = timeGuaranteed_{i,j}
24  timeGuaranteed_{i,j} = simulTime
25                  + lookahead_{i,j}
26    where j is list of downstream LPs
27         for LP_i
28  if timeGuaranteed_{i,j}
29      ≠ old_timeGuaranteed_{i,j}
30   send null messages with
31     timeGuaranteed_{i,j} to LP_j
32  end if
33 end while
```

Figure 2: Main Simulation Loop for Simulator based on Asynchronous Simulation Protocol

To keep the overhead of null messages low, null messages are only sent after an LP finishes simulating all events with timestamp less than the computed safetime (line 28-32). But this might slow down the simulation time advance. To overcome this problem, time guarantee information is piggybacked to each external event (line 17). This helps LPs to keep a more up-to-date view of the time guaranteed and in turn improve the protocol's performance.

## 3 THE SUPPLY CHAIN MODEL

Figure 3 shows the supply chain model that is used to benchmark the distributed simulation based on HLA RTI and MPI-ASP. Even though the supply chain model presented here is simple, it is reasonably representative of the semiconductor industry. In this model, the supply chain involves only two stages of the whole chain. Multiple wafer fabrication plants are modeled to supply wafers to a single assembly and test factory. The wafer fabrication plants produce wafers based on their own forecasted demand. The release rate of wafer products is varied each month based on three parameters. These parameters are the current inventory level of the wafer product at the assembly and test factory (as perceived by the wafer fabrication plants at the time they adjust the release rate), current work-in-progress of the wafer product, and the forecasted demand of the wafer product. For details, see Jain et al. (1999).
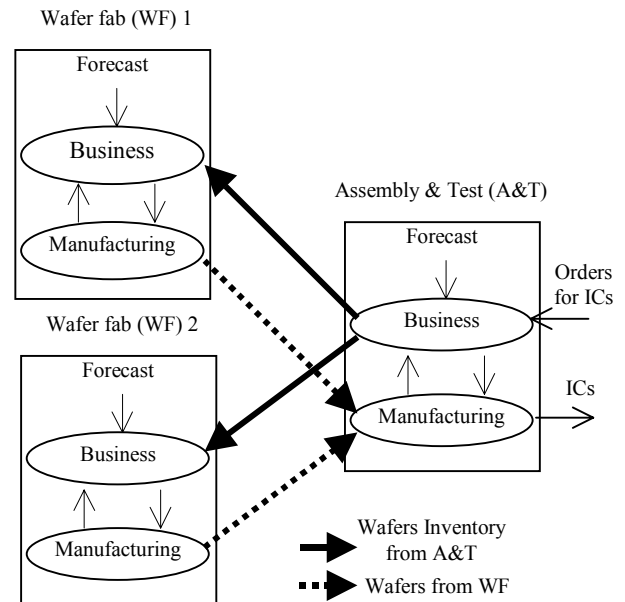


Figure 3: The Supply Chain Model

In this simplified supply chain model, there is an information flow (inventory level of each wafer product) from the assembly and test facility to the wafer fabrication plant and a shipment flow (shipping of all produced wafer products) in the reverse direction. Inventory information is sent on a monthly basis while shipment of products is sent on a daily basis. Corresponding to the information and material flows, two types of messages are needed, one for the inventory information and another for the wafer shipment. Having identified the message types, the message format is then defined. The supply chain simulation can then be integrated through distributed simulation once each module is implemented.

Each individual factory being modeled in this study is based on some real world wafer fabrication factories and assembly and test plant. The models of the wafer fabrication factories are obtained from the sample data model of Sematech (Sematech 1997), which is available through the Internet (MASMLab 1999), while the model of the assembly and test plant is obtained through our past projects with the industry. The matching of wafers to IC was developed based on volume considerations; that is, large volume wafer products supply large volume IC products. Table 1 shows the results of this matching algorithm for the six sample data models of wafer fabrication factories to the assembly and test facility. Multiple wafer fabrications are needed to fulfill the volume requirement of the assembly and test factory.

Table 1: Matching of Wafer Fabrication Plants to Assembly and Test Plant

| Model | 1 | 2 | 3 | 4 | 5a | 6 |
|---|---|---|---|---|---|---|
| No of wafer fabrication plants | 3 | 5 | 3 | 15 | 6 | 10 |

## 4 BENCHMARKING

Figure 4 shows the system architecture of the two approaches. The architecture was designed in such a way that the application layer (supply chain simulation) is independent of the underlying simulation engine. For the purpose of this study, the underlying simulation engine is implemented using a version of the HLA RTI and the distributed simulation protocol, MPI-ASP, discussed earlier. The RTI is version RTI1.3NG-v2 developed by the Defense Modeling and Simulation Office of the Department of Defense, USA. To implement the asynchronous simulation protocol, a version of MPI, MPICH version 1.2 available from the Argonne National Laboratary of University of Chicago, is used to facilitate message passing among simulation models. The simulator is implemented using C++, compiled with GNU GCC compiler version 2.95.1. The simulation is run on a homogeneous platform, with four Sun UltraSparc II workstations (three of which are 248 MHz and the remaining is 167 MHz). The workstations are running Sun Solaris 2.6 and interconnected by a 100 Mbps Ethernet.

As observed from Table 1, the total number of factories for our supply chain simulation is larger than the number of workstations that are available. Thus, we partition the simulation based on the workload contributed by each factory. Table 2 shows the final partitioning information. For the purpose of this benchmark, we allow multiple factories to share the same workstation. One very important point to note here is that they can be run as separate operating system processes. These processes make simulation progress without knowing the details of what other processes are doing. Thus, the primary requirement of model encapsulation is still satisfied.
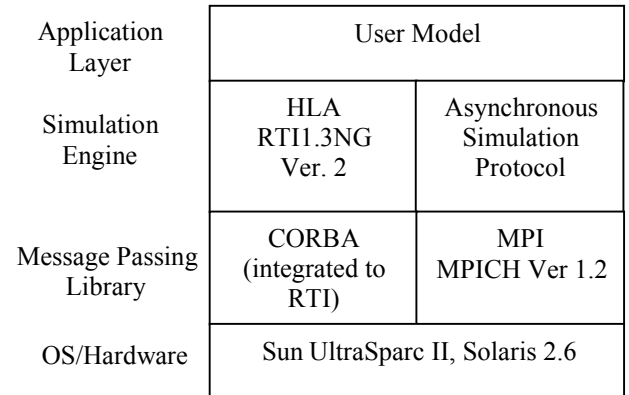
| Application Layer | User Model | |
|---|---|---|
| Simulation Engine | HLA RTI1.3NG Ver. 2 | Asynchronous Simulation Protocol |
| Message Passing Library | CORBA (integrated to RTI) | MPI MPICH Ver 1.2 |
| OS/Hardware | Sun UltraSparc II, Solaris 2.6 | |

Figure 4: System Architecture

Table 2: Partitioning Information

| Model | 1 | 2 | 3 | 4 | 5a | 6 |
|---|---|---|---|---|---|---|
| Workstation 1 | 1 wfab | 2 wfab | 1 wfab | 5 wfab | 2 wfab | 4 wfab |
| Workstation 2 | 1 wfab | 2 wfab | 1 wfab | 5 wfab | 2 wfab | 3 wfab |
| Workstation 3 | 1 wfab | 1 wfab | 1 wfab | 5 wfab | 2 wfab | 3 wfab |
| Workstation 4 | 1 a&t | 1 a&t | 1 a&t | 1 a&t | 1 a&t | 1 a&t |

Figure 5 compares the execution time achieved by using the alternative distributed simulation approaches. As one can see, the execution time achieved by using MPI-ASP is better than the execution time achieved using the HLA RTI. This is mainly due to higher overhead of HLA RTI underlying implementation. The overhead of HLA RTI is mainly contributed by grant time computation. We would like to stress here that we have made all necessary steps to optimize the way our simulation engine makes use of the RTI. The optimizations are:

a)  calling *tick()* of the RTI without specifying the minimum and maximum amount of time that a federate allows its local RTI component to execute its tasks,
b)  requesting for time advancement periodically instead of for every event to reduce the overhead,
c)  running rtiexec (the server process of RTI) at a separate workstation that does not run any federates.

More details on the optimization can be found in Turner, Cai, and Gan (2000).

## 5 CONCLUSIONS

In this paper, we have illustrated how distributed simulation can be used to realize model encapsulation. Model encapsulation is particularly important in a distributed supply chain simulation that involves
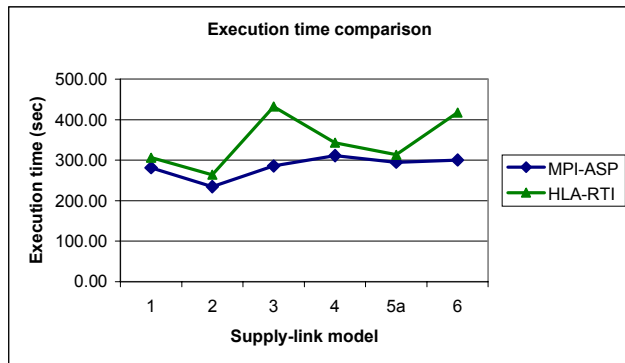
Figure 5: Execution Time Comparison

companies that cross the enterprise boundaries. By having this capability, companies will no longer need to worry about sharing their confidential information when the supply chain model is built. Detailed simulation of each enterprise can then be used as a tool to facilitate the process of supply chain design, evaluation and optimization.

Distributed simulation built on top of the HLA RTI and MPI-ASP was also discussed. Two important issues, namely interoperability and performance, were used to compare the two approaches. HLA offers an edge over MPI-ASP in term of interoperability. But from the benchmarking test, it is obvious that MPI-API outperforms the HLA RTI. With this split of advantages, the choice of approach for distributed simulation will then be determined by the primary concern of the application. If speed of execution is critical for a supply chain application, MPI-ASP will definitely be the option. Alternatively, if the supply chain application involves a number of existing simulation models (developed using various simulation languages that support HLA interface), HLA will then be the winner. Even though runtime performance might be important in the latter case, the time that is saved from rebuilding the simulation model for MPI-ASP might offset the loss in runtime performance.

## ACKNOWLEDGMENTS

## REFERENCES

Archibald, G., N. Karabakal, and P. Karlsson. 1999. Supply chain vs. supply chain: Using simulation to compete beyond the four walls. In *Proceedings of the 1999 Winter Simulation Conference*, ed. P.A. Farrington, H.B. Nembhard, D.T. Sturrock, and G.W. Evans, 1207-1214. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Chandy, K.M., and J. Misra. 1979. Distributed simulation: A case study in design and verification of distributed programs. *IEEE Transaction on Software Engineering*, SE-5: 440-452.

Fujimoto, R.M. 1998. Time management in the high level architecture. *Simulation Special Issue on High Level Architecture* 71 (6): 388-400.

Gan, B.P., and S.J. Turner. 2000. An asynchronous protocol for virtual factory simulation on shared memory multiprocessor systems. *Journal of Operational Research Society Special Issue on Progress in Simulation Research* 51: 413-422.

Gropp, W., E. Lusk, and A. Skjellum. 1994. *Using MPI: Portable parallel programming with the message passing interface*. MIT Press.

Kuhl, F., R. Weatherly, and J. Dahmann. 1999. *Creating computer simulation systems: An introduction to the high level architecture*. Prentice Hall PTR.

Jain, S., C.C. Lim, B.P. Gan, and Y.H. Low. 1999. Criticality of detailed modeling in semiconductor supply chain simulation. In *Proceedings of the 1999 Winter Simulation Conference*, ed. P.A. Farrington, H.B. Nembhard, D.T. Sturrock, and G.W. Evans, 888-896. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Lutz, R. 1998. High level architecture object model development and supporting tools. *Simulation Special Issue on High Level Architecture* 71 (6): 401-409.

MASMLab. 1999. TestBed [Online], maintained by Modeling and Analysis of Semiconductor Manufacturing Laboratory, Industrial and Management Systems Engineering department, Arizona State University, USA. Available as http://www.eas.asu.edu/~masmlab/ftp.html.

Padmos, J., B. Hubbard, T. Duczmal, and S. Saidi. 1999. How i2 integrates simulation in supply chain optimization. In *Proceedings of the 1999 Winter Simulation Conference*, ed. P.A. Farrington, H.B. Nembhard, D.T. Sturrock, and G.W. Evans, 1350-1355. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Sematech 1997. *Modeling Data Standards, Version 1.0*. Technical Report, Sematech Inc., Austin, TX 78741.

Turner, S.J., W. Cai, and B.P. Gan. 2000. Adapting a Supply Chain Simulation for HLA. To appear in *Proceedings of 4th International Workshop on*

*Distributed Simulation and Real Time Applications*. IEEE Press.

## AUTHOR BIOGRAPHIES

**BOON PING GAN** is an Associate Research Fellow with the Manufacturing Planning & Scheduling group at Gintic Institute of Manufacturing Technology, Singapore. He received a Bachelor of Applied Science in Computer Engineering and Master of Applied Science from Nanyang Technological University of Singapore in 1995 and 1998 respectively. His research interests are parallel and distributed simulation, parallel programs scheduling, and application of genetic algorithms. His email address is <bpooi@gintic.gov.sg>.

**LI LIU** is an Associate Research Fellow with the Manufacturing Planning & Scheduling group at Gintic Institute of Manufacturing Technology, Singapore. She received her Bachelor of Science in Mathematics from Beijing University in 1991 and Master of Computer Engineering from Academic Sinica, China in 1994. Her research interests are parallel discrete event simulation, supply chain web-based simulation. Her email address is <lliu@gintic.gov.sg>.

**SANJAY JAIN** is a Senior Research Fellow and the Manager of the Manufacturing Planning & Scheduling group at Gintic Institute of Manufacturing Technology, Singapore. His research interests are in the area of using modeling and analysis techniques in development and operation of manufacturing systems, and in improving performance of simulation systems through parallel and distributed execution. Prior to joining Gintic, he worked for several years as a Senior Project Engineer with General Motors North American Operations Technical Center in Warren, MI, USA. He received a Bachelors of Engineering from University of Roorkee, India in 1982, a Post Graduate Diploma from National Institute for Training in Industrial Engineering, Bombay, India in 1984, and a PhD in Engineering Science from Rensselaer Polytechnic Institute, Troy, New York in 1988. He is a member of Institute of Industrial Engineers and of the editorial board of *International Journal of Industrial Engineering*. His email address is <sjain@gintic.gov.sg>.

**STEPHEN J. TURNER** received his MA in Mathematics and Computer Science from Cambridge University (UK), and his MSc and PhD in Computer Science from Manchester University (UK). He was a Lecturer and subsequently Senior Lecturer in the Computer Science Department at Exeter University (UK). He is currently an Associate Professor in the School of Computer Engineering at Nanyang Technological University (Singapore). His current research interests include: parallel and distributed simulation, visual programming environments, parallel algorithms and architectures, distributed computing and agent technology. His email and web addresses are <assjturner@ntu.edu.sg> and <www.ntu.edu.sg/home/assjturner>.

**WENTONG CAI** is an Associate Professor in the School of Computer Engineering (SCE), Nanyang Technological University (NTU). He obtained his B.Sc. in Computer Science from Nankai University (P.R. China) in 1985, and Ph.D. from Exeter University (UK), also in Computer Science, in 1991. Since 1990, he has been actively involved in the research in the areas of Parallel and Distributed Simulation (PADS). Dr. Cai is a member of IEEE and his current research interests include: PADS, Parallel Programming Tools and Environments, and Performance Analysis. His email and web addresses are <aswtcai@ntu.edu.sg> and <www.ntu.edu.sg/home/aswtcai>.

**WEN JING HSU** is presently the director of the Centre for Advanced Information Systems, School of Computer Engineering, NTU. Before joining NTU, he was on the faculty of National Chiao Tung University, Taiwan ROC, and the Michigan State University, USA. He also worked as a Visiting Scientist at the IBM Palo Alto Scientific Center and IBM T. J. Watson Research Center, NY. His present research interests are in the computer algorithms, architectures and applications of parallel computers, simulation and decision support, the scheduling and routing of automated guided vehicles. He is involved in two large-scale projects funded by the NSTB, concerning the simulations of port operations and for manufacturing industries. He has also been a consultant in industry related projects, including the anchorage management and vessel conflict prediction systems for the Maritime and Port Authority of Singapore. His email address is <hsu@ntu.edu.sg>.