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

A SIMULATION ARCHITECTURE WITH DISTRIBUTED CONTROLLERS FOR CELL-BASED MANUFACTURING SYSTEMS

Hansoo Kim
SugJe Sohn
Ying Wang
Tolga Tezcan
Leon McGinnis
Chen Zhou

School of Industrial and Systems Engineering
Georgia Institute of Technology
765 Ferst Drive
Atlanta, GA 30332-0205, U.S.A.

ABSTRACT

A number of manufacturing systems are categorized into Cell-Based Manufacturing Systems (CBMSs), which have similar physical configurations typically composed of manufacturing cells, transporters and their controllers. However, since the control modules are not separable from the model in conventional simulation frameworks, it is difficult to develop a generic high-fidelity simulator for CBMSs with flexibility in control structure.

We propose a generic simulation architecture for CBMSs, "CellSim", using the concept of Controller-In-Loop (CiL), in which all supervisory controllers are separated from simulation models, and developed as individual controller modules. CellSim has three sub-architectures, i.e., the design/modeling architecture, the simulation execution architecture, and output analysis architecture. CellSim has distinctive features and advantages for flexible, high-fidelity, and integrated manufacturing system design and modeling, compared to the conventional simulation frameworks. In this paper, we present distinctive features and architecture of CellSim for CBMSs as well as the concept and implementation of CiL.

1 INTRODUCTION

A *Cell-Based Manufacturing System* (CBMS) is a manufacturing system configured with multiple manufacturing cells. Each cell consists of multiple processing equipment and intracell material handling systems. These cells are interconnected with intercell material handling systems.

Due to the benefits coming from hierarchal controlling structure decomposed naturally by the system configuration, CBMS has been implemented in various modern manufacturing systems, namely, cellular manufacturing (Shahrukh 1999), flexible cell manufacturing (Nyman 1992), job shop, and even flow line with cells can be placed in the category of CBMS. Especially, the cell-based manufacturing is gaining a momentum in recent production environment of automated and agile manufacturing, for example, 300mm wafer fab manufacturing, one of current cutting-edge automated manufacturing systems (Quinn and Bass 1999). In addition to the hierarchical structure of control domain, the reconfiguration of a manufacturing system can be efficiently performed in CBMS since the system can be changed with individual cell unit without interfering other cells when the reconfiguration is necessary.

In this paper, we present a generic architecture for modeling and simulating manufacturing systems in CBMS. Actually, CBMSs have common properties in physical manufacturing configurations. However, since the control logics are totally different according to the control behaviors of individual manufacturing systems and since the control modules are not separable from simulation models and interchangeable between the control modules, it is difficult to develop a generic simulation architecture for the simulations of CBMSs with existing simulation frameworks.

To break through the intrinsic limitations in existing simulation frameworks, we propose a simulation framework using "*Controller-In-Loop* (CiL)" for the generic simulation for CBMSs. As it shows on the name similar to "*Hardware-In-Loop* (HiL)" which means a hybrid simulation with physical hardware systems (Maclay 1997), CiL is a distributed simulation environment linked with controllers, which are decoupled from the behavior models. Through CiL, the behavior model transmits the current state and event information to controllers and the controllers make decisions based on the information. Then, the

controllers update the states in simulation behavior model via a distributed network, say, *High Level Architecture* (HLA) (Kuhl *et al.* 1999).

CiL-typed simulations have remarkable benefits in developing a generic architecture for flexible and high-fidelity design and modeling of CBMSs. Since behavior models and control modules are separated in CiL framework, it is possible to generate a high fidelity simulation behavior model directly and automatically from manufacturing system design. In addition, controllers can be developed with high-fidelity beyond the limitation of programming capacity of specific simulation modeling tools. They are just like the physical controllers since the controllers of CiL use the high-level programming languages, e.g., C++ or Java.

Based on CiL scheme, we develop a generic and high-fidelity simulation architecture for CBMS, *CellSim*. To implement the CellSim, we use High Level Architecture (HLA) which is one of recent standard infrastructures for distributed simulations (IEEE 2000). CellSim has distinctive features such as flexibility, Integrality, high-fidelity, and rapid modeling for general modeling of CBMS. This paper describes these features in the following sections, i.e., Controller-In-Loop (CiL) framework in Section 2, main features of CellSim in Section 3, and implementation view of the architecture of CellSim in Section 4.

2 CONTROLLER-IN-LOOP SIMULATION FRAMEWORK

2.1 Features of Control-In-Loop (CiL)

The main concept of Controller-In-Loop (CiL) is the extraction of control logics from simulation model. In here, *control logic* means a process of decision making to determine the behaviors of simulation models. For example, in manufacturing systems, product releasing, dispatching, sequencing, and controlling transporters are included in the control logics. In general, as these control logics are getting complicated, the complexity of simulation modeling is also rapidly increased. Hence, we extract control logics from simulation model, and design a protocol used to communicate between *behavior model* and *controllers*. The controllers are the implemented modules of control logics. We intend to reduce the complexity of simulation modeling process from this separation of behavior model and controllers. This is the key idea of CiL.

Figure 1 shows the conceptual configuration and benefits of CiL with comparison to conventional modeling. Conventional simulation model has been using the control codes inside the behavior model, therefore we should make another simulation model when we build similar models with different control logics. For example, 4 simulation models in Figure 1 (a) need the costs for 4 behavior models and 4 controllers, since it is difficult to reuse a controller with different behavior model. Figure 1 (b) presents a snapshot of CiL. If we have an interaction mechanism, we only build 2 behavior models and 2 controllers to make 4 simulation models in Figure 1 (a). In addition, the models are reusable for any other different models, so the benefit of CiL is apparent in this simple illustration.

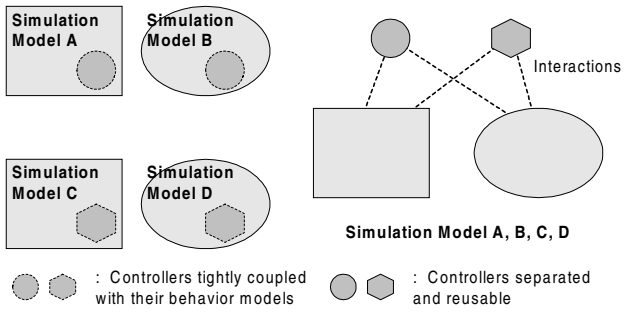

Figure 1: (a) Conventional Model (b) Control-In-Loop

Distributed simulation execution in CiL framework is as follows. When simulation is started, all state information is transferred to the corresponding controllers subscribing that information. Based on the information, controllers are initialized and send control commands to the simulation behavior model. According to control commands, simulation behavior model updates its state information, and sends back the updated state information to the controllers. This process is repeated until the termination of simulation run. In distributed simulation (Fujimoto 2000), the most important factor is time synchronization management. Mostly, since the simulation behavior is mapped to a discrete event simulation model, without strict time synchronization mechanism, the simulation results would unreliable disturbed by the propagation delay in controllers. In order to prevent these causality errors, a robust time synchronization mechanism is essential in CiL framework.

2.2 Advantages and Disadvantages of Controller-In-Loop Simulation Framework

Compared to conventional control frameworks, i.e., control logics within simulation behavior model, CiL has following comparable advantages.

At first, CiL framework provides high-fidelity simulation modeling environment. Regarding to this advantage, we can consider two aspects; one is the reduced complexity in simulation behavior model, and the other is the realistic implementation of control logics. Since the simulation behavior models exist separated from control modules, we can avoid the abstraction of behavior model due to the complexity of tightly coupled control logics in conventional simulation models. In addition, the control logics can be developed in more realistic and elaborated way.

CiL supports “Plug and Playable” controller evaluation environment. If a newly introduced controller program follows the rules and regulations of CiL framework, we can utilize it with no changes in the behavior model. Hence, just

by "plugging" a controller, we can "play" with the controller to evaluate its performance. This feature provides a consistent evaluation environment for various control logics.

Also, the simulation behavior model and controllers can be developed concurrently. The concurrency in modeling reduces the simulation modeling cost in time and labor. CiL framework has an important merit in behavior model and controller reusability. If the simulation behavior model is designed in the object-oriented methodologies, the entities in system will be modeled as object classes that are independent from control logics. These object classes can be structured in accordance with object types and used as building blocks for simulation behavior modeling. The "building-blocks" concept enables the rapid prototyping of simulation models. In addition, if a control logic is implemented just like a general algorithm, the implemented controller could be used for various simulation behavior models. Hence, CiL provides a simulation modeling repository for reusing both simulation models and controllers.

Although there are a lot of advantages in simulation modeling, CiL framework has disadvantage in simulation executions. Since CiL framework is implemented as distributed simulation in which there are propagation delays in communication among federates, simulation execution can be relatively slower than that of conventional simulation frameworks. However, we expect the effects of these disadvantages be minimized with help of rapidly growing technologies in high-speed networking and parallel and distributed computing.

3 CELLSIM: A GENERIC SIMULATION ARCHITECTURE FOR CBMS

With using the beneficial features of CiL framework and the characteristic of common control structure in CBMS, we developed generic simulation architecture for CBMS, *CellSim*. The simulation modeling in CellSim is much easier and intuitive since the simulation modeling process is equivalent to the system design process describing system layout and specification for equipment in the manufacturing system; the design itself is the modeling at the same time. The features of CellSim are described in this chapter.

3.1 Cell-Based Manufacturing System (CBMS)

The category of Cell-Based Manufacturing Systems (CBMSs) includes any types of production systems composed of manufacturing cells and their corresponding control systems. Each cell has a number of processing or metrology equipments and material handling systems for material flows inside of the cell area. These cells are interconnected with another material handling system called the intercell material handling systems. In general, to interface between intercell material handling system and manufacturing cells, there are central buffers or stockers for incoming or outgoing products.

A CBMS has three levels of common control hierarchy. The factory domain on the top level deals with product release and intercell material handling system. Main issues of this control domain are how to manage work-in-process (WIP) efficiently satisfying production schedule and how to control the intercell transporter control to avoid congestions and deadlock situations. The middle control domain is the cell level control, which handles sequencing WIPs waiting in the central buffer in each cell for available process equipment and it also controls the intracell material handling systems. The bottom level is the equipment level control domain. In this level, the equipment controller makes decisions for product sequencing and batching for each machine.

3.2 Features of CellSim

CellSim is general simulation architecture for CBMSs. In this section, we discuss main distinctive features of CellSim. These features can be viewed with two aspects. One is about the architecture of CellSim, and the other is about systems modeling and simulation using CellSim.

3.2.1 Features of the Architecture of CellSim

- *General simulation architecture for CBMS.* Paradoxically, it means that CellSim is a special simulator only for CBMSs, but since many manufacturing systems are implemented as CBMSs, CellSim can be named as a general architecture for CBMS. The core framework of CellSim is the "Controller-In-Loop (CiL)." Hence, CellSim inherits all advantages of CiL framework. Since CBMSs typically have identical or similar control structures, they can be characterized only by the simulation behavior model.
- *Intuitive simulation modeling.* The simulation behavior models are generated automatically from system design descriptions such as layout design and equipment characterization. In another word, it means that simulation modeling is substituted by intuitive system design. This feature removes the system designers' burden of simulation modeling, which has been separated from system design. More details are described in Section 4.2.
- *Using HLA for CiL framework.* Since CellSim is based on CiL framework, implementation of message communication and time synchronization are required. For these functions, High Level Architecture (HLA), which is standard structure for distributed simulation, is used. CellSim supports both real time and fast execution for discrete event simulations.
- *Message protocol.* For seamless interactions between controllers and simulation behavior models,

a set of well-defined message protocol is required to specify the controls and actions.

- *Animation and numerical analysis module.* Animation and numerical analysis are the functions of non-HLA connected programs. That means the output analysis is performed separated from the simulation execution architecture. Only after each simulation run, animation and numerical analysis begins. More details are described in Section 4.4.
- *Object oriented programming structure.* CellSim assumes the object-oriented programming (OOP) using C++ or Java. OOP has the advantages in database manipulation, web-based applications, HLA framework, etc. as well as general benefits of OOP such as the efficient maintenance.

3.2.2 Features of the Modeling of CBMS in CellSim

While the features of CellSim structure are focused on CellSim itself, the features in this section are those related with modeling CBMS. These features describe the level of detail of CBMS modeled in CellSim.

- *Whole factory-scope modeling with high-fidelity.* Due to computational load and/or modeling cost, it has been common way of modeling to model a specific scope of manufacturing lines in most of conventional general-purpose simulation tools. Sometimes, though the modeling covers whole system, the fidelity of the model has been frequently too abstract and purpose-specific, for example, a simple model to calculate the average total cycle time. While, CellSim supports a whole factory-wide simulations with high-fidelity, and it can be used to analysis the design, operation and control problems.
- *Consistent evaluating environment for various alternatives.* From the plug and playable feature in Control-In-Loop framework, various simulation alternatives can be evaluated within the same simulation behavior model. In addition, for the cell reconfiguration, CellSim also provides a consistent platform.
- *Equipment based modeling.* If some factory alternatives have the same typed equipments, then we can use the equipment models in library like a template. This feature enables us to build the equipment library for reusability.
- *Reusable controller programs.* Since control modules are separated from simulation model, we can test various control logics conveniently. For example, CONWIP (Constant Work In Process) policy (Spearman et al. 1990) with different CONWIP sizes can be installed and tested with behavior models. Likewise, through implementing controller program by parameterizing simulation behavior model dependent data, controller can be generalized and reused.

4 SYSTEM ARCHITECTURE OF CELLSIM

4.1 Architecture Overview

Figure 2 presents the architecture overview of CellSim. It is conceptually divided into three sectors, i.e., *Design/ Modeling Architecture*, *CellSim Simulation Execution Architecture*, and *Output Analysis Architecture*. These three sub-architectures are roughly coincides with the sequence of simulations, i.e., design/modeling, run, and analysis.

In Design/Modeling Architecture, Design/Modeling Module generates the simulation input information. It includes design factory layout, specification equipment modeling with their detailed properties, and selection of material processing/handling policies, process steps/recipes of the lots in production, product mix. The information is stored in central database, Simulation Model/Log Database, and ready to retrieved to the simulation entities at the beginning of distributed simulation runs. The flexibility in simulation design and modeling comes from the use of model libraries, e.g., equipment and product libraries, and HLA-structured federates, e.g., transporter controllers and factory federates.

In the second sector of architecture, CellSim Simulation Execution Architecture, the simulator main body consists of several distributed federates, i.e., Behavior Models and Controllers. Behavior model has a number of simulation objects, e.g., process tools, and Controllers present a number of controllers, e.g., intercell controller. They can perform real-time, scaled real-time, and discrete-event-driven simulation according to the time advance scheme in HLA. All activities and interactions in control and behavior models in simulations are logged in central database after the simulation runs.

Finally, in the Output Analysis Architecture, Animation module displays the states of distributed federates for visualized monitoring purpose. Animation Module depicts the simulation log graphically, and assists to understand the complicated behaviors during simulations. Particularly, in verifying the abnormal behavior in controlling material handling systems, this module plays important roles. The simulation log is also arranged and transmitted to the Analysis Module. Analysis Module performs various analysis works.

4.2 CellSim Design/Modeling Architecture

4.2.1 Design/Modeling Module

The simulation design and modeling begins at Design/Modeling Module. It helps the users design the target

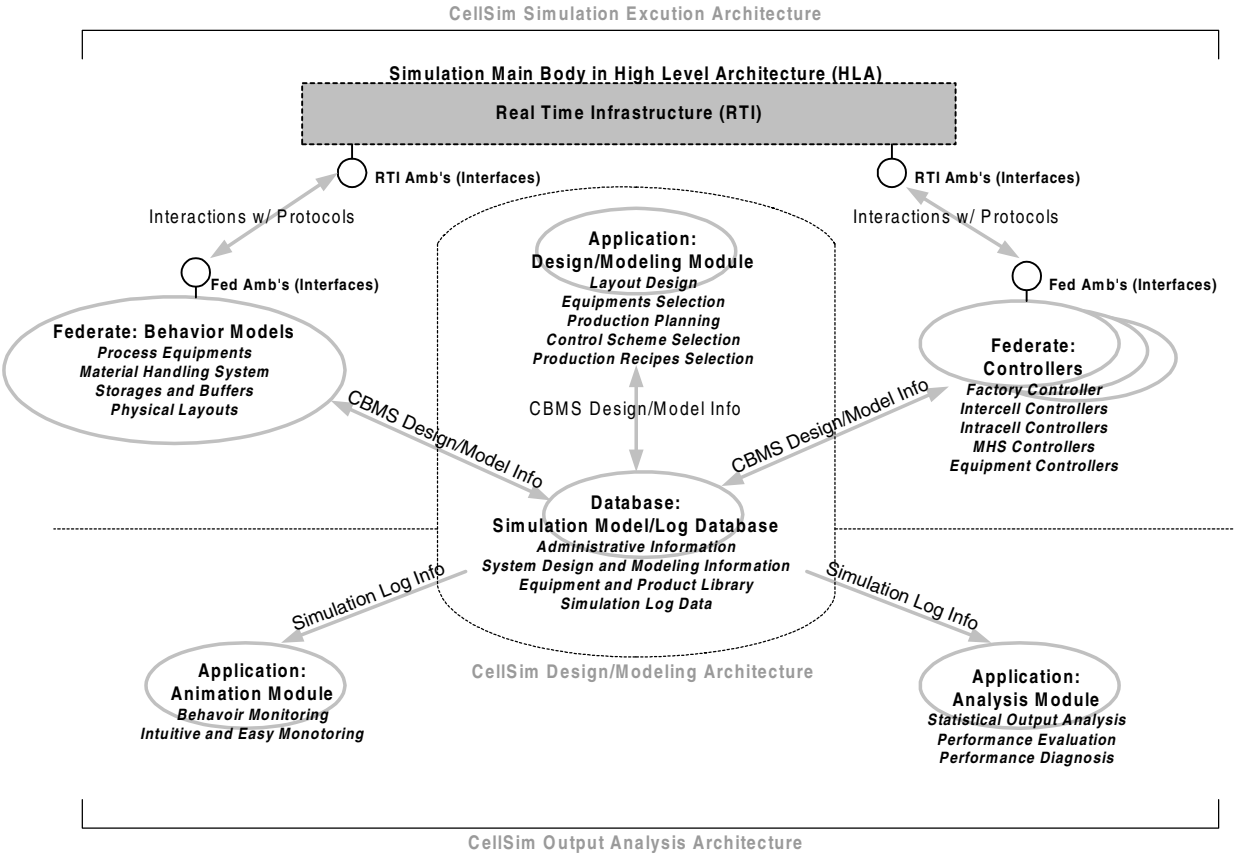

Figure 2: Architecture Overview of CellSim

system into convenient and systematic modeling format. A web-based application with 3-dimensional presentation could be more efficient for complicated systems. As seen in Figure 2, this application is not involved in simulation execution part, which is based on HLA framework. It is connected to central database and exists on network, possibly on Internet. Figure 3 illustrates an example of design/modeling module, which has been developed in the project, High Fidelity Virtual Environment for 300mm Wafer Fab (HiFiVE) at Georgia Institute of Technology (Kim *et al.* 2001).

The functional features of Design/Modeling Module are categorized in 3 parts as follows.

- *General project administration part*: It includes the secured login and accessibility management, project information management in databases, and importation / exportation / modification of simulation models.
- *Layout and equipments specification part*: It includes the geometric design of facilities and transportation paths in factories, and process and transportation equipments selection and deployment.

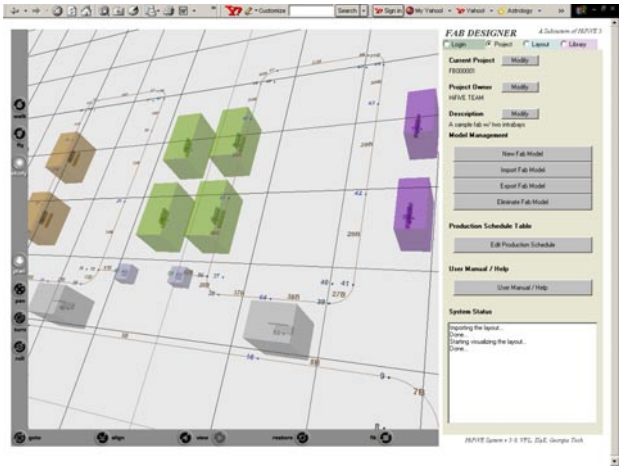

Figure 3: A Design/Modeling Module (*FactoryDesigner*, HiFiVE Project, Virtual Factory Laboratory, Georgia Institute of Technology)

- *Operations and control specification part*: It includes the master production planning, equipment library management, lot type library management, process recipes and setup information specification, and control scheme selection for process and transportation equipments.

4.2.2 Simulation Model/Log Database

The central database existing in the heart of CellSim is connected to all the subsystems. It has three groups of data tables.

- *Administrative data tables*: They are used to manage the user information and project information. The administrative data tables should be constructed to accommodate a number of system design and the modeling alternatives of each design.
- *Model data tables*: They maintain the geometric information such as locations of equipments and paths transporters, facilities dimensions. Also the mapping information of each equipment to its reference equipment model in equipment library. One set of model data tables presents one alternative modeling of a system to be simulated.
- *Library data tables*: They are shared to all the modeling alternatives. The libraries composed of equipment library, lot (product model) library, and process steps library. In operational view, the lot information should have recipe information, for example, a lot type A should undergo the recipe X which has step 1, 2, 3 inside of it.
- *Log data tables*: They are generated only after each of simulation run and used in Animation Module and Analysis Module. The log data has a compact structure just enough to generate the animation play and analysis process. For example, to perform a mean value analysis (MVA) of process equipments, we should have its simulation log of material arrival and departure times, the state changes, etc. We also collect the information about individual production history, i.e., which lots have been processed in which process equipment when and how long.

4.3 CellSim Simulation Execution Architecture

4.3.1 Behavior Models

Behavior Models embraces all the virtual behavioral objects, such as virtual process equipments, inside a HLA federate. We could deploy multiple federates for behavior models if the size of behavioral objects gets huge. At the beginning of each simulation, it retrieves the simulation models from the central database, and then automatically and dynamically generates the behavioral objects, i.e., virtual equipments (Kim *et al.* 2000). During the simulations, generally speaking, Behavior Models receives commands on interaction channels from controllers and the corresponding virtual equipments respond it and send back their interactions. They repeat these exchanges of interactions. Some behavioral objects are passive ones. They are just updated by controllers and maintain its static or dynamic properties. For example, the buffers and stockers have a property of current number of materials. It is updated and referenced by control federates, however, the buffers or stockers do not modify it themselves. The other behavioral objects are active ones. They update the properties of themselves and sometimes the others' also. A transporter, for example, can update the location information of both a work-in-process and itself.

Real Time Infrastructure (RTI) serves the robust time management and simplicity in exchanging interactions. RTI manages the time synchronization, which has been conventionally a complicated and cumbersome work in building distributed simulation systems, particularly supporting real-time scheme.

4.3.2 Controllers

Controllers are also constructed in federates and connected to RTI framework. The descriptions on essential controllers for CBMSs are as follows.

- *Factory Controller*: It controls the factory-level behaviors, i.e., start / terminate / pause the simulation. It should communicate with the other controllers and behavior models to check if they are ready to receive the factory-level commends.
- *Intercell Controller*: It governs the material releaser, final product stocker, intercell transporters, and intracell stockers. Intracell interconnects the intracells. It releases the raw materials, delivers the work-in-process, and collects the completed products. Therefore it is a critical function of Intercell Controller to manage the part releaser according to the production plan. Suppose we use CONWIP policy in a CBMS design and we follow a product mix of several lots, for example, then Intercell Controller should decide next lot type that must be released at every loading of completed product. It should also monitor the current states of intracells in its decision domain and make commands for the intercell transporters to dispatch products. It cooperates with intracell controllers and Behavior Models.
- *Intracell Controller*: Intracell Controller controls the intracell, in which several process equipments, stockers, and transporters are located. Intracell Controllers make a logical decision at every event in its intracell, e.g., arrival of a material into stockers, completion of processes in the process equipments, completion of delivery of transporters. Actually, it is complicated control problem to control transporters and equipments avoiding any logical errors such as deadlocks on the load ports of process equipments. We could design highly

intelligent and sophisticated dispatching logics inside the Intracell Controllers against the problems.

- *Intercell / Intracell Transporter Controller*: The transportation commands from Intercell Controllers and Intracell Controllers do not describe any detailed path information. In other words, the controllers just transmit the commands that a certain lot should be dispatched from somewhere to somewhere else. Intercell and Intracell Transporter Controllers generate detailed and possibly optimized path information to send to transporter objects in Behavior Model. These controllers are realistically equivalent to physical transporter controllers from automated equipment vendors. We could flexibly test the transportation logics with Intercell and Intracell Transporter Controllers.

4.4 CellSim Output Analysis Architecture

4.4.1 Animation Module

Unlike conventional simulations where the material flows are not delicately controlled and described, CellSim supports a high-fidelity logistics control inside factory. Graphical animation of simulation status is very important in detecting logically abnormal operations in manufacturing facilities. Therefore, Animation Module in CellSim is required for easy and intuitive detection of the problems in material flows, as well as the overall simulation monitoring. CellSim separated the animation module from the simulation execution architecture, since the animation possibly imposes much computational load in simulation executions. This separation increases the simulation speed and provides easy play-back and forward. The simulation log data retrieved from the central database after each simulation run is designed as compact as possible including the essential information about event generating entity, event description, affected entities by the event, property changes, and the simulation time when the event occurred.

4.4.2 Analysis Module

Analysis Module uses the simulation log to extract the necessary information for each analysis factor, e.g., utilization of equipments, arrival / process / departure rates of process equipments, minimum / mean / maximum work-in-process, bottleneck analysis using MVA, etc. Since Analysis Module is an application coded in high level programming language, we can freely add or modify the analysis functions for any other analysis of interests.

5 CONCLUSION

In this paper, we defined and observed the characteristics of the *cell-based manufacturing systems* (CBMSs), and presented a genetic and high fidelity simulation architecture, *CellSim*, for CBMSs. CellSim is based on the concept of *controller-in-loop* (CiL) simulation framework, in which the control logics are separated from the behavior models.

CiL framework provides a flexible control architecture for the simulation models of CBMSs. The distributed control modules significantly reduce the cost in building various alternative models due to the reusability and flexibility of control modules.

The simulation architecture for CBMSs, CellSim, is developed with the CiL framework on *High Level Architecture* (HLA) environment. CellSim has three architectural partitions, i.e., *Simulation Execution Architecture*, *Simulation Design/Modeling Architecture*, and *Output Analysis Architecture*. CellSim provides a general simulation architecture for CBMS, intuitive simulation modeling, CiL framework using HLA, and animation and numerical analysis modules. In modeling activities using CellSim, we have the beneficial features of whole factory-scope modeling with high-fidelity, consistent evaluating environment for various alternatives, equipment based modeling, and reusable controller programs.

The simulation architecture of CellSim with CiL framework is being applied to the application of semiconductor wafer manufacturing system in Georgia Institute of Technology. It is planned to perform elaborated benchmarking and performance evaluation of various design alternatives using a complete CellSim-based simulation system. The effectiveness of CellSim architecture itself is also being studied in research activity.

ACKNOWLEDGMENTS

This research is being partially funded by *The W.M. Keck Foundation* and *Ford Motor Company*. We also appreciate *PRI Automation*'s help in this research.

REFERENCES

Fujimoto, R. 2000. *Parallel and Distributed Simulation Systems*, NY: John Wiley & Sons

IEEE Std. 1516.1.-2000. 2000. *IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) Federate Interface Specification*, New York: Institute of Electrical and Electronics Engineers.

Kim, H., C. Zhou, and H. Du. 2000. Virtual Machines for Message Based, Real-Time and Interactive Simulation. In *Proceedings of the 2000 Winter Simulation Conference*, ed. J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, 1529-1532. Orland, Florida

Kim, H., J. Park, S. Sohn, Y. Wang, S. Reveliotis, C. Zhou, D. Bodner, and L. McGinnis. 2001. A High-Fidelity, Web-based Simulator for 300mm Wafer Fabs. In *Proceedings of the 2001 IEEE Systems, Man, and Cybernetics Conference*. 1288-1293. Tucson, Arizona.

Kuhl, F., R. Weatherly, and J. Dahmann. 1999. *Creating Computer Simulation Systems*, NJ: Prentice Hall.

Maclay, D. 1997. Simulation gets into the loop. *IEE Review* 43 (3): 109-112.

Nyman, L. 1992. *Making Manufacturing Cells Work*, NY: McGraw-Hill, Inc.

Quinn, T., and E. Bass. 1999. 300 mm factory layout and material handling modeling: Phase I Report, International SEMATECH.

Shahrukh, A. 1999. *Handbook of Cellular Manufacturing Systems*. NY: John Wiley & Sons.

Spearman, M., D. Woodruff, and W. Hopp. 1990. CONWIP: A Pull Alternative to Kanban. *International Journal of Production Research* 28: 879-894.

AUTHOR BIOGRAPHIES

HANSOO KIM is a Ph.D. student of School of Industrial and Systems Engineering at Georgia Institute of Technology. He received his B.S. from Hanyang University, and M.S. from Korea Advanced Institute of Science and Technology (KAIST) in 1991 and 1993 respectively. After 5 years working for Samsung SDS as a simulation consultant, he has studied Ph.D. course since 1998. His research interests include theory of model abstraction, high-fidelity modeling methodology, and simulation applications for manufacturing and logistics areas. His email address is <kimhs@isye.gatech.edu> and web address is <http://www.isye.gatech.edu/~kimhs>.

SUGJE SOHN is a Ph.D. student of School of Industrial and Systems Engineering at Georgia Institute of Technology. He received his B.S. and M.S degrees of Naval Architecture and Ocean Engineering at Seoul National University, Korea, and M.S.I.E. degree from Georgia Institute of Technology. He worked in a project related with 300mm Semiconductor Manufacturing at Samsung Electronics Research Center as an intern in 2002. His research interests include the simulation modeling methodology and architectures, and simulation based design in manufacturing and logistics areas. His contact information is <sjsohn@isye.gatech.edu> and <http://www.isye.gatech.edu/~sjsohn>.

YING WANG currently is a Ph.D. student of School of Industrial and System Engineering at Georgia Institute of Technology. She received her MS degree from ShanDong University in China in 1998. Her research interests are mainly in building distributed simulation environment for modern automated manufacturing and logistics systems. Her email address is <ying@isye.gatech.edu>.

TOLGA TEZCAN is a Ph.D. student of School of Industrial and Systems Engineering at Georgia Institute of Technology. He received his B.S. degree from Bilkent University, Turkey, and his M.S. degree from University of Southern Colorado. His research interests include queuing networks and simulation. His email address is <ttezcan@isye.gatech.edu>.

LEON McGINNIS is Gwaltney Professor of Manufacturing Systems in the School of Industrial and Systems Engineering at Georgia Tech. He is Associate Director of the Manufacturing Research Center, and founding Director of the W.M.Keck Virtual Factory Lab.. His research interests include abstract representation of industrial logistics problems, and the integration of information technology and operations research to support decision making for design, planning, control, and improvement of industrial logistics systems. He may be contacted at <leon.mcginnis@isye.gatech.edu>.

CHEN ZHOU is an Associate Professor in the School of Industrial and Systems Engineering at Georgia Institute of Technology. He received his B.S.M.E. in Tianjin University in China, M.S.M.E., and Ph.D. in Industrial Engineering from the Pennsylvania State University. His email address is <chen.zhou@isye.gatech.edu> and web address is <http://www.isye.gatech.edu/people/faculty/Chen_Zhou >.