Development of virtual foundry fab based on distributed multi-agents

Chih-Yuan Yu*, Han-Pang Huang**

Robotics Laboratory, Department of Mechanical Engineering,
National Taiwan University, Taipei, 10660, Taiwan.
TEL/FAX: (886) 2-23633875, E-mail: hphuang@w3.me.ntu.edu.tw
**Professor and correspondence addressee *Graduate student

Abstract

The key success of the foundry fab is based on the order fulfillment and the customers' satisfactory. The main objective of order fulfillment is to deliver the products on time. However, many processes, including due date setting, planning and scheduling of the order and real-time shop floor operations, are involved in accomplishing this goal. In order to enhance the customers' satisfactory, the foundry company should not only promise on-time delivery but also provide the real-time information of orders. Virtual foundry fab can realize these two important issues and will be constructed in this paper based on distributed agents and a proposed event-driven platform.

The core of virtual foundry fab is the order *fulfillment process*. The process is triggered by the customers' orders in the foundry fab environment and handled by order management system. This paper focuses on the entire order fulfillment process. It includes order management process, production planning, lot priority setting, tool capacity allocation, shop-floor operation, and event monitoring process. Each subsystem is established as an agent and performs its task on the distributed framework. The result shows that the overall virtual foundry fab can be easily constructed using the distributed multi-agents.

Keywords : Virtual Foundry Fab, Event-driven, Distributed agent

1. Introduction

With the growth of Internet, the legacy system cannot fulfill all the functions needed by an enterprise. The integration of application islands is getting more important. Moreover, it is impossible to renew all existed applications in the enterprise to communicate with a new application. To solve this problem, a distributed environment which can provide a transparent, distributed, loading balancing, fault tolerant, extendable, scalable, and secure integration environment, is developed. Many researches [2, 6] discussed about the properties of the distributed environment. The distributed environment has two types. One is distributed computing performed by several processors, the other is distributed applications executed their own tasks. The distributed environment mentioned in this paper is the latter one.

In the semiconductor foundry fab, the satisfaction of the customers is the key to succeed. Recently, the way to reduce the cycle time in the high yield rate fab and to provide the customer real-time information of orders are widely discussed.

Su *et al.* [12] proposed an enabling framework for virtual fab and concluded that virtual fab is the critical aspects for achieving competitiveness in the semiconductor industry. Lin *et al.*[9] used Queueing Colored Petri net (QCPN) to model the virtual fab.

In short, service, scalability, transparency and extensibility are the keys for a foundry fab to construct its manufacturing system. The scope of this paper is to (1) model the OFP (Order Fulfillment Process) in the foundry fab in the distributed environment using multi-agents, (2) provide the functionalities for each application (agent).

The OFP includes four sub-processes, i.e., order management process, planning process, Manufacturing Execution System (MES) and event monitor. These four sub-processes describe the activities triggered by receiving an order in the foundry fab. Agents are constructed in each sub-process. Each agent has its goals to achieve and its functions to execute. Hence, the functionality of each agent is also the emphasis of this paper.

2. Multi-agents in the Order Fulfillment Process

2.1. Multi agents in the order fulfillment process

An agent is an active object, which possesses certain abilities to perform task and communicate with other agents to complete the goal of the system [10]. An agent has an internal behavior model, a functional component consisting of procedures/heuristics/strategies, and a protocol for interacting with other agents [11]. Agents in the same environment form a society to achieve the global goals. Hence, multi-agents technology is widely adopted in manufacturing system [11], supply chain coordination [10], simulation [13] and shop floor control. The OFP can be viewed as the enterprise-scale integration of applications. Each application in the OFP is regarded as an agent and performs its task based on its knowledge constrained by its objectives. Besides, the OFP requires the coordination and collaboration of agents in the enterprise to achieve the goals. Therefore, they can communicate with each other on the generic message-passing platform discussed in [3].

0-7803-7087-2/01/$10.00 © 2001 IEEE

2.2. Agent communication and coordination

An agent should communicate with others. One may invocate the other's procedure. The invoking agent passes the parameter through the function to inform another agent in order to accomplish its goal. Then the return values tell the invoker the status of the function call. The methods for agent communication can be classified into three types: direct message passing, blackboard discussion, and mediator.

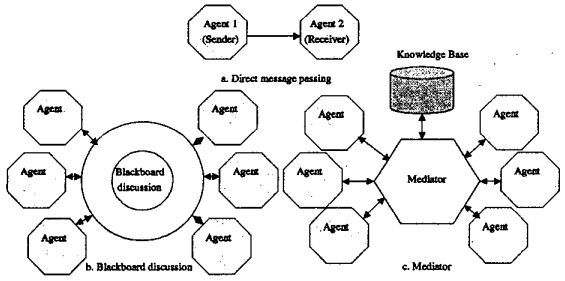

Fig. 1 The common communication methods in multi-agents

2.2.1. Direct message passing

In message-passing paradigm shown in Fig. 1-a, one agent can talk to the other directly so that the communication efficiency is the highest one. Besides, the exchanging messages between agents can form the protocols for cooperation. The flexible format of the message provides very versatile communication capabilities. The disadvantage of direct message passing is that the sender agent and receiver agent do not consider the states of other agents.

2.2.2. Blackboard

The blackboard is an open and common place for multi agents to discuss and exchange information, data and knowledge, as shown in Fig. 1-b. The agent, which has been updated or owns the new information, writes the data on the blackboard. Then any agent, which wants to obtain the new information of other agents since its last connection, can log into the blackboard. The disadvantage of blackboard discussion is that the agent cannot be informed by others for interested information. It has to regularly connect to the blackboard and seek for new information. In short, it is not an event driven mechanism so that the time will be consumed during on connecting to the blackboard and getting nothing new. However, the newest information of other agents are listed on the board, an agent can solve its problem with the global consideration and may get the better solution for the society. Usually, the blackboard discussion is used for both result sharing and task sharing [1].

2.2.3. Mediator

Mediator is the extended blackboard structure that all the information is routed by the mediator on the basis of knowledge rather than posted on the board as shown in Fig. 1-c. The mediator mechanism not only posts the new information but also routes the information to the related agents. It can compensate for the shortage of the blackboard discussion. However, each agent should register to the mediator in advance and the mediator will provide a mechanism to pass the information. As a result, the mediator will be the most difficult one to program.

2.2.4. GMPP [3] (Generic Message-Passing Platform)

The communication method used in this paper is the extension of the direct message passing. The method mentioned in [3] is used to communicate between agents while the COM+ event is not the direct message passing. The mechanism of Application-to-Application in GMPP is loosely coupled between senders and receivers. The coordination between agents is the set of all events registered in the operation system. The sender just fires a certain event and the receivers (the numbers of receivers may be larger than one) get the event and do the corresponding action. Basically, the message is not transferred within one sender and one receiver. Any agent, which is interested in getting the event, can subscribe the event in the operating system. The routing of events is important in the mediator and is handled by the operating system. The communication method used in this paper is good for real-time communication since the blackboard discussion is time consuming while the mediator is difficult to implement.

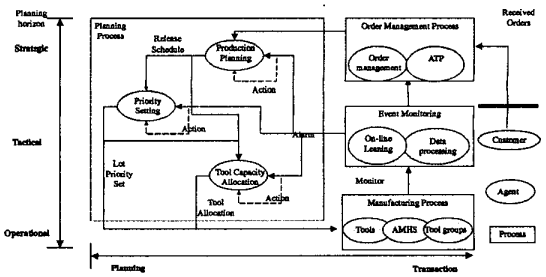

Fig. 2 The sub processes and agents in the OFP

3. Modeling of Order Fulfillment Process

The OFP is modeled as a society of agents. The modeling approach is based on the object-oriented system analysis and design. The entire OFP is divided into five sub-processes, i.e. order management process, planning process, manufacturing process and event monitoring, as shown in Fig. 2. Each sub-process contains its own agent. Sequence diagrams of some agents are shown in Fig. 3.

The following sections describe the functionalities of each agent in the corresponding sub-processes and the implementation of the multi-agents over the GMPP.

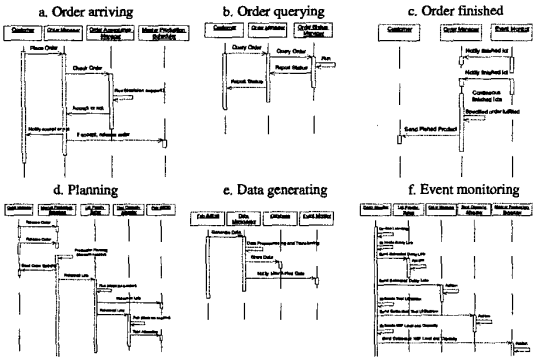

Fig. 3 Sequence diagrams for agents

3.1. MES (Manufacturing Execution System)

The MES is the manufacturing unit in the enterprise. It contains material flow, lot processing and lot movement in the shop floor. Hence, modeling of the MES should include the scheduling of lots, lot dispatching, tool dispatching, material movement and coupling effect between tool groups and tools [15]. The overall MES is controlled by the Factory Control System (FCS), as shown in Fig. 4. Tool groups, virtual tool groups and tools are modeled to represent the processing steps while the Automated Material Handling System (AMHS), which includes intrabays, interbays and stockers, are used to move lots. To complete one step of operations, the lot should flow into the processing hierarchy and AMHS hierarchy one at each time.

CTPN (Colored Timed Petri Net) is a powerful tool to model and analyze the discrete event dynamic system. DCTPN (Distributed CTPN) [5, 6] is the extension of CTPN using in the distributed environment. The detailed modeling information can be referred to [7]. The modeling of a MES using DCTPN over the GMPP as shown in Fig. 5. The models are constructed by DCTPN designers. The communication between designers is based on NTU-NET [8] since it requires real-time tokens passing. The COM servers, such as CTPNEng, OrderEng, DispatchingRuleEng, can be plugged into the designer so that it can extend the functionality of the designer. The reporter and viewer can show the real-time information of the model as receiving the update event sent from the designer through the COM+ event. The designer can also notify the engineer when an error occurs in the MES by sending him/her a short message, or emailing, or voice broadcasting via the gateway through the COM+ event. The engineer has to handle the shop floor error by making decision on the WAP site when he/she is out of the office.

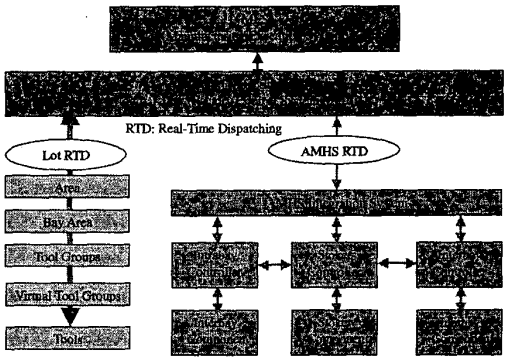

Fig. 4 The hierarchy of MES and shop floor elements

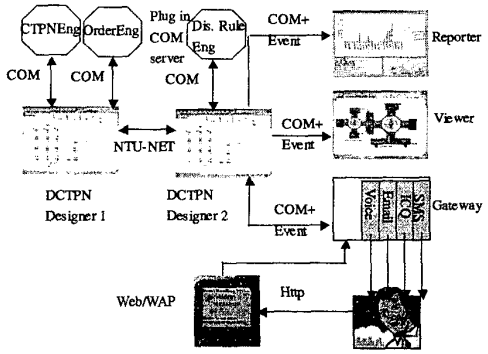

Fig. 5 The message passing within the MES over the GMPP

3.2. Event Monitoring

3.2.1. Data-processing Agent

The data-processing agent converts the collected data from the MES into the data with physical meaning of the foundry fab for information query and preprocesses the data for on-line learning agent. In the first part, the historical and real-time information of lots, orders, tools and vehicles should be prepared for customers, engineers, managers and decision support systems. Generally speaking, data-processing agent is the data source of decision support for other agents. Besides, it also monitors error signals and data exception and then sends the alarm message to the related agents over the GMPP.

3.2.2. On-line Learning Agent

The flows of the lots in an IC fab are like job shops but more complex. It is very difficult to identify the relationship between tools and lots; therefore, taking the entire fab as a unit for modeling is a tremendous task. The on-line learning system divides the entire fab into the subsystems, i.e., tool models. The detailed information can be referred to [14]. The on-line learning agent provides the following functions

The cycle time estimator applies tool model to calculate the following information.

(1) The cycle time of the lot i (CT(i)) is the summation of the time of each step. Namely,

$$CT(i) = \sum_{n=1}^{N} (W_n + P_n) \quad (1)$$

where W_n is the waiting time of lot i in step n, and P_n is the processing time of lot i in step n.

(2) The number of steps, S(i), that the lot will go through in a day can be obtained by

$$S(i) = \left\{ \arg\min_{s} \left\{ \sum_{n=k+1}^{s} (W_n + P_n) > 1\ day \right\} \right\} - k \quad (2)$$

where k is the current step, and s is the step that the lot i will arrive one day later.

(3) The ERCT (Estimated Remaining Cycle Time) of lot i which supports least slack policy can be obtained by

$$ERCT(i) = \sum_{i=k+1}^{N} (W_n + P_n) \quad (3)$$

where k is the current step and N is the total steps of the lot i.

(4) The Slack value of lot i, s(i), is calculated as

$$s(i) = \delta(i) - t - ERCT(i) \quad (4)$$

where $\delta(i)$ is the due date of lot i and t is the current time.

3.3. Order Management Process

3.3.1. Order Management Agent

The order management agent provides an interface to customers. In a foundry fab, the order management agent has to process the incoming orders, estimate the due date, provide the real-time information of orders, and notify the exception of lot processing. In short, the order management agent should provide all the information of orders to customers. Since customers of the foundry fab are all enterprise-scale, the order management agent should provide a standard interface for customers. It can be viewed as a Business-to-Business communication. As a result, eXtended Markup Language (XML) and Partner Interface Process (PIP) in the RosettaNet are used in order management agent as the standards to communicate with customers.

3.4. Planning Process

3.4.1. Production Planning Agent

After receiving the orders, the production planning system should transfer the customer orders into manufacturing orders, i.e., the release schedule of lots. Time horizon in the production planning is a key attribute to decide the release schedule. If the time horizon is too short, the long-term order effects will not be considered. On the other hand, if the time horizon is too long, the planning system may be too complicated. Many lot release rules are summarized in [9]. Idle avoidance, Constant work-in-process (CONWIP), POISS, DETERMIN, Workload Regulating (WR) and PWR (Parametric WR) are frequently used heuristic rules. WR is used as the lot release rule in this paper.

3.4.2. Priority Setting Agent

The released lot is assigned a priority class by the system. In order to have a better performance in on-time delivery, the least slack policy is used to prioritize all lots daily in the fab before executing the tool capacity allocation procedure. The proportion of each priority class is fixed as 5% lots for SH, 15% for H, 30% for R, 45% for N, and 5% for S.

The estimated remaining cycle time and the slack value of each lot can be calculated from Eq. (3) and Eq. (4), respectively. If the slack s(i) > 0, it means that the lot commits its due date on time; otherwise, the lot will be delayed. The smaller the slack value of the lot has, the higher the priority class of the lot will be re-assigned. Since the proportion of each priority class is fixed, only partial lots can be raised their priority classes. Therefore, the slack value of a lot is served as the reference for adjusting its priorities.

The steps to re-assign the priority class of each lot are listed below.

Step 1: Calculate the slack values of all lots in the fab,

Step 2: Highlight some important lots by the managers,

Step 3: Sort the lots in terms of the larger slack value and importance,

Step 4: Re-assign the priority class according to the proportion, (SH, H, R, N, S) = (5%, 15%, 30%, 45%, 5%). The fixed percentage is determined from the shop floor supervisor.

3.4.3. Capacity Allocation Agent

The capacity of a tool is number of wafers that the tool can process in a day regardless the recipes. In general, the capacity of a tool depends on the PM schedule of the tool, the frequency of the setup change, recipes and the idle time of the tool. It is difficult to estimate the actual capacity of the tool due to the unpredicted interrupt and the idle time of the tool. The main purpose of this sub-section is to decide the amount of the tool capacities for each tool group. The detailed information for capacity allocation is described in [15]

The results of this procedure are the WIP information for one day later, reserved capacity of the tool for each tool group, the bottleneck tools, and the bottleneck tool groups. The bottleneck tools and bottleneck tool groups can be easily found from the capacity allocation algorithm. Let $N_B(m)$ = the number of blocked lots in the tool group m. It tells that $N_B(m)$ lots are blocked and can not continue their routes due to the lack of capacity of the tool group m. If $N_B(m)$ is too large, this tool group is called a bottleneck tool group.

The shop floor manager should take action to solve the bottleneck issue based on the information provided above. The action depends on the extra available resources, the experience of the managers, the PM schedule of the tools and the on-time delivery performance. Although it is not easy to make decision, the manager can adjust the capacities of the tools with the aid of the above results. Which tool should be added or which tool's PM schedule

can be changed in order to solve the bottleneck can rely upon the list of the bottleneck tools and tool groups.

After adjusting the capacities and the usage of the tools, re-run the procedures until the balance between the tool utilization and bottleneck is achieved.

3.5. The Message-Passing Architecture of Order Fulfillment Process

GMPP [3] is suitable for the message passing mechanism of the OFP. The agents are constructed on the event layer. The entire message passing in the OFP is similar to the one used in MES in Fig. 5. The difference between the MES model and the OFP is two folds. One is that in the MES model, the applications are CTPN designers. While in the OFP, applications are the agents with their tasks. The other is that the communication protocol used between applications is mostly NTU-NET, while COM+ events are used to act as the extension of direct message passing. The gateway, which provides the protocols for Application-to-Person and Person-to-Application, is used in both models to notify the users.

4. Simulation and Result

The simulations based on the data collected from a real foundry fab are shown in this section. The simulation result includes cycle time estimation, tool group move, tool move and bottleneck identification.

4.1. Tool group moves

If the tool group move of the tool group m is $Move_{TG}(m)$, it means that there are $Move_{TG}(m)$ wafers flowed through the tool group m, i.e., the throughput of the tool group. The tool group move is defined as when a lot completes an operation, the lot contributes moves to this tool group. The moves that added to this tool group are equal to the number of the wafer of the lot. For example, if a lot with 24 wafers has been processed by the tool group m, then 24 tool group moves are added to the tool group m. Thus, we have

$$Move_{TG}(m) = \sum_{i \in \tilde{L}(m)}^{N} N_L(i) \tag{5}$$

where N is the number of tools in $\tilde{L}(m)$; $N_L(i)$ is the quantity of $L(i)$; $\tilde{L}(m)$ is the set of lots which are processed in the $TG(m)$; L is the set of all lots. Note that $\tilde{L}(m)$ is the subset of L.

The estimated total tool group moves in "1998-12-31" are 54597 wafers, as shown in Fig. 6. The move of the tool group 33 is as high as 2850 wafer moves.

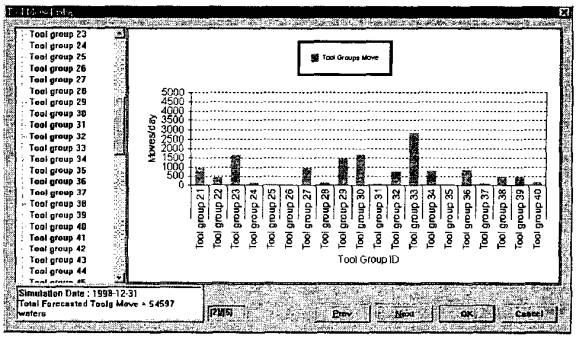

Fig. 6 Estimated tool group moves

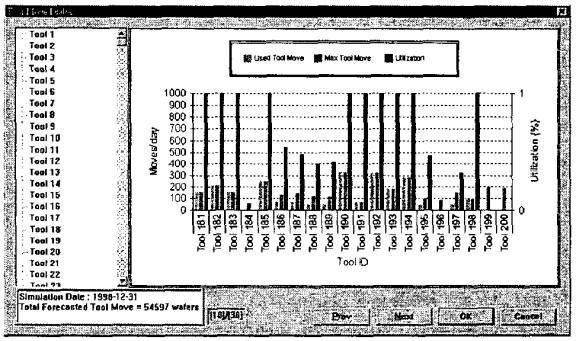

Fig. 7 Estimated tool moves

4.2. Tool moves

Similar to the tool group moves, the estimated tool moves in the tool n, $Move_T(n)$, is the number of wafers that are processed in the tool n. We have

$$Move_T(n) = \sum_{i \in \tilde{L}(n)}^{N} N_L(i) \tag{6}$$

where $\tilde{L}(n)$ is the set of lots which are processed in $T(n)$.

Fig. 7 shows not only the allocated capacity but also the utilization of the tools. The utilization of the tool m is defined as

$$e_{T(n)}(t) = \frac{AC_{T(n)}(t)}{UC_{T(n)}(t)} \tag{7}$$

where $AC_{T(n)}(t)$ is the allocated capacity of the tool n at day t.

4.3. Bottleneck analysis

Only the tool groups, which block the lots, are listed in Fig. 8. It can be found that the tool groups 3 and 7 are critical to the system. Over 10 lots are blocked in these two tool groups and they are called the bottleneck tool groups.

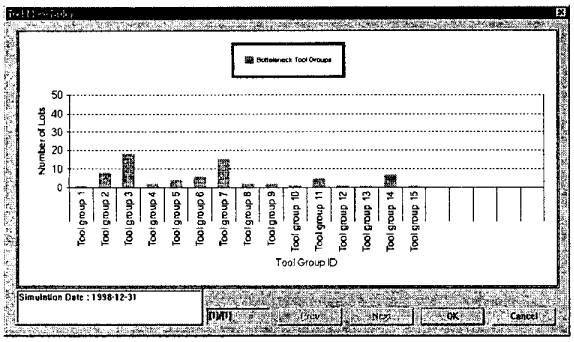

Fig. 8 Bottleneck tool groups

In order to resolve the bottleneck tool groups 3 and 7, three and two new tools, for example, are added to the tool groups 3 and 7, respectively. The result shows that the bottleneck tool group 3 is no more the bottleneck tool group, while the number of the blocked lots of the tool group 7 is reduced from 15 lots to 6. Note that it is only a reference for the manager. In general, the bottleneck tool groups can be handled by adding new tools or adjusting the PM schedules of the existing tools according to the actual data. The manager can identify the high-utilization tools and decide whether the PM schedule of the tool can be adjusted or not.

5. Conclusion

This paper integrates the authors' previous works realize the modeling of the OFP. The OFP contains four subsystems, order management process, planning process, manufacturing process, and event monitoring process. The applications in each sub-process are constructed as agents in the distributed environment. The agents communicate with each other over the GMPP for better efficiency. The information such as tool group move, bottleneck analysis can be obtained in this paper. However, communication efficiency and the advantages of the proposed architecture in the distributed environment has not been compared.

References

[1] Walter Brenner, Rudiger Zarnekow, Hartmut Wittig, *Intelligent Software Agents: Foundations and Applications*, Berlin: Springer-Verlag, 1998.

[2] Kei Shiu Ho, Hong Va Leong, "An extended CORBA Event Service with Support for Load Balancing and Fault-Tolerance," *Proceedings of International Symposium on Distributed Objects and Applications*, pp. 49-58, 2000.

[3] H.P. Huang, C.Y. Yu, "Development of a Generic Message-Passing Platform for distributed environment," *Proceedings of the 2001 IEEE International Conference on Intelligent Robots and Systems Tuson,* Nov., 2001.

[4] R. Kolluru, S. Smith, P. Meredith, R. Loganantharaj, T. Chambers, G. Seetharaman, T. D'Souza, "A Framework for the Development of Agile Manufacturing Enterprise," *Proceedings of the 2000 IEEE International Conference on Robotics and Automation San Francisco*, pp.1132-1137, 2000.

[5] C.H. Kuo, H.P. Huang and M.C. Yeh, "Object-Oriented Approach of MCTPN for Modelling Flexible Manufacturing Systems," *International Journal of Advanced Manufacturing Technology*, vol.14, pp.737-749, 1998.

[6] C.H. Kuo, "Development of Distributed Component Based Manufacturing System Framework," *Dissertation*, Department of Mechanical Engineering, National Taiwan University, 1999.

[7] C.H. Kuo, "Performance Evaluation of a Controlled OHT System in a Fab Intrabay Using Modular-Based Colored Timed Petri Net," 5th International Conference on Mechatronics Technology, 2001.

[8] L.R. Lin and H.P. Huang, "Real-Time Networking for the Implementation of CIM," *Proceedings of International Conference on Automation Technology*, Taiwan, vol.1, pp.21-28, 1996.

[9] M.H. Lin, L.C Fu, "Modeling, Simulation and Performance Evaluation of an IC Wafer Fabrication System: A Generalized Stochastic Colored Timed Petri Net Approach," *Internal Journal of Production Research*, vol.38, no.14, pp.3305-3341, Sep., 2000.

[10] F.R. Lin, G.W. Tan, Michael J. Shaw, "Modeling Supply-Chain Networks by a Multi-agent System," *1998 IEEE Proc. 31st Annual Hawaii International Conference on System Science*, pp.105-114, 1998.

[11] Riyaz Silora, Michael J. Shaw, "Coordination Mechanisms for Multi-agent Manufacturing Systems: Applications to Integrated Manufacturing Scheduling," *IEEE Transactions on Engineering Management*, vol.44, no.2, pp.175-187, 1997.

[12] Y.H. Su, R.S. Guo, S.C Chang, "Virtual Fab: Enabling Framework and Dynamic Manufacturing Service Provisioning Mechanism," submitted paper to IEEE Transactions on Engineering Management in January 2001.

[13] G.S.H. Tan, K.L. Hui, "Applying Intelligent Agent Technology as the Platform for Simulatoin," *Proceedings 31st Annual Simulation Symposium 1998*, pp.180-187, 1998.

[14] C.Y. Yu and H. P. Huang, "Fab Model Based on Distributed Neural Network," *National Conference on Automation Technology*, ChiaYi, pp.271-277, 1999.

[15] C.Y. Yu, H.P. Huang, "Priority-Based Tool Capacity Allocation in the Foundry Fab," *Proceedings of the 2001 IEEE International Conference on Robotics and Automation Korea*, May, pp. 1839-1844, 2001.