Proceedings of the 1999 IEEE
International Conference on Robotics & Automation
Detroit, Michigan • May 1999

Virtual Factory - A Novel Testbed for an Advanced Flexible Manufacturing System

Ming-Hung Lin , Li-Chen Fu and Teng-Jei Shih
Dept. of Computer Science and Information Engineering
National Taiwan University,Taipei,Taiwan,R.O.C
E-mail: lichen@ccms.ntu.edu.tw

Abstract

Nowadays, the market place is continuously changing and unpredictable, and hence an efficient prototyping environment is crucial. Here, we propose a virtual factory wherein an efficient prototyping testbed will be provided. This paper is one among very few that tries to represent the virtual factory in an analytic form so that many existing mathematical analysis can be applied. New pseudo resources can be added to form a new virtual environment, and control policy designed by engineers will be evaluated before being issued. An example of utilizing the prototyping testbed is given. The proposed testbed is compared with a traditional testbed and the results validate the intelligence and efficiency of the present prototyping testbed.

1 Introduction

The concept of virtual factory or virtual manufacturing is brought forth taking advantage of powerful computing capability of computers to achieve the goal of manufacturing in computer. There are three paradigms of virtual manufacturing (VM) [5] [6]. So far, Lee [7] developed a prototype of a VM system using an expert system shell and a database management system on internet. Nowadays, advent of advanced information technologies such as computer networking and 3D graphics drives more and more industrial companies to reform from traditional human-oriented semi-automation to information-oriented full-automation [2] [3] [4].

Today, manufacturing industries rely increasingly on distributed manufacturing enterprises organized by multi-enterprise partnerships [8] [9] [12]. To establish a VF, one must have a clear understanding of the manufacturing capabilities of all parties in the production network [10]. To date, Macedo [11] proposed an artificial intelligence based tool that helps to select the partners of a virtual factory. Bodner [13] developed modeling tools which support rapid development of detailed simulation models to assess system performance. Henning [14] presented the simulation functions as an interactive computer-based VF. Jain [15] proposed a VF framework for their systematic and efficient use. Finally, Saraswat [16] [17] proposed an approach to build a highly flexible computer controlled manufacturing facility, a suite of simulation tools which emulate all functions of the real factory.

To summarize from the previous work on the topic of VF, some aimed at developing the simulation models, some emphasized on virtual teaming, and some works on virtual prototyping. However, they are more conceptual than realistic for those results are either too coarse or somewhat fragmentary. Thus, when one comes to realize the capability "Manufacture in the Computer" for some real manufacturing, there are still in lack of many building blocks. Kazuaki [2] proposed some simple concept of the virtual factory, including the basic types of integrating the virtual world and the real manufacturing resources and activities, which however is far too abstract. In this paper, we propose a virtual factory wherein a brandnew efficient prototyping testbed will be provided.

The organization of the paper is as follows. In the next section, we will state the problem and give some discussion. In section 3, we try to represent the virtual factory in an analytic form. The complete developing environment for the prototyping testbed will be addressed, which will be based on the skeleton we proposed previously [20]. In section 4, an operation model is proposed, which will provide a general approach for the prototyping testbed. In section 5, the proposed prototyping testbed is implemented on an assembly cell, which is compared with a testbed proposed earlier [19] via simulations. In section 6, a brief conclusion is given.

2 Problem Statement, Analysis and Modeling

The prototyping testbed which we call virtual factory is a complete integrated developing environment

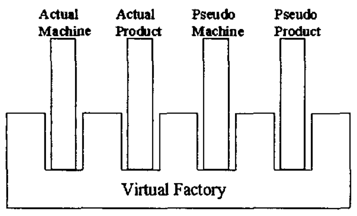

Figure 1: The prototyping testbed

as shown in Fig 1. Its mission is to resolve the following questions.

(1) How to provide an efficient prototyping skeleton for integrating and automating the manufacturing system software and hardware.

(2) How to give an operation model to respond to the requirement from the environment or manufacturing engineers subsequently.

2.1 Pseudo Domain and Physical Domain

Consider a continuous system in a discrete-event world. Let $X_D(t_k)$ and $X_C(t_k)$ be the discrete and continuous state variable at time t_k, respectively, and denote the time of occurrence of event k as t_k. In the interval $[t_{k-1}, t_k)$, we assume $X_D(t_k)$ remain unchanged and $X_C(t_k)$ vary continuously with speed vector $v(t)$ at time t_k^-, where time t_k^- is denoted as the time right before but arbitrarily close to t_k. For corresponding state variables, their update equations are the following,

$$X_D(t_k) = G[X_D(t_{k-1})]$$

and

$$X_C(t_k) = X_C(t_{k-1}) + \int_{t_{k-1}}^{t_k} v(u)du$$

, where G is a nonlinear function. Consider the time t in the interval $[t_{k-1}, t_k)$, Denote the state variable at time t as $X(t)$. It is possible to deduce from the state variable at time t_{k-1} and the history of the speed vector v from t_{k-1} to t the reachable state variable at time t, namely, $X(t)$, as follows:

$$X(t) = G[X_D(t_{k-1})] + \int_{t_{k-1}}^{t} v(u)du \qquad (1)$$

.

From the viewpoint of the virtual factory, the environment consists of pseudo domain (D_p) and physical domain (D_s). A pseudo domain D_p is a triplet $< PX(t^P), PQ(t^P), PT >$, where $PX(t^P), PQ(t^P), PT$ are pseudo resource-state variables, pseudo system-state variables and pseudo time-state variables, respectively, i.e., $t^P \in PT$, $PX(t^P) \subset X(t)$, $PQ(t^P) \subset X(t)$. Pseudo resource-state variables may characterize the changes in the pseudo resources or record pseudo data, e.g., the motion trajectory of a pseudo robot that is run by computer, or the pseudo-manufacturing data generated by computer via some software. In this sense, the resources here are composed of hardware and software processes. On the other hand, the pseudo system-state variable represents the state of the system and is updated solely by computer simulation.

Similarly, the physical domain D_s is also a triplet $< SX(t^S), SQ(t^S), ST >$, where $SX(t^S), SQ(t^S), ST$ are physical resource-state variables, physical system-state variables and physical time-state variables, respectively, i.e., $t^S \in ST$, $SX(t^S) \subset X(t)$, $SQ(t^S) \subset X(t)$. The difference of the physical domain from the pseudo domain lies in that the former reflects the reality but the latter only reveals what to be expected if some pseudo resources are going to reality. In general, physical resource-state variables includes physical hardware and software counterparts, e.g., hardware like robot, buffer, loader and software like vision package. Physical system-state variable represents the copy of the state of the overall manufacturing system. Because pseudo time-state variables PT is the time horizon used by a computer, it is in general to be faster than the physical time-state variables ST.

Based on the previous definition, we now denote a set of virtual resource-state variables as VX, given by $VX = PX(t^P) \cup SX(t^S)$. By the same token, let VQ be a set of virtual system-state variable such that $VQ = PQ(t^P) \cup SQ(t^S)$. Consider a transformation function T, where $T : PX \rightarrow SX$, and let $x(t^P) = T[y(t^S)]$, $x \in PX(t^P), y \in SX(t^S)$. If there exists an inverse transformation function T^{-1} such that $y(t^S) = T^{-1}[x(t^P)]$, then there is an *isomorphic relation* between x and y, e.g., *actual robot PUMA* and *pseudo robot PUMA*.

2.2 Virtual Automaton

We define the virtual factory as VF, $VF = [VA, LB, SP]$, where VA, LB, SP are the virtual automaton, librarian broker, and specifications, respectively. A virtual automaton VA is a 9-tuple set $VA =< Q, I, \rho, q_0, W, O, \psi, \xi, \tau >$, where

a. Q is a finite set of state, $Q \subseteq VQ$.

b. I is a finite set of input, $I \subseteq VX$.

c. ρ is a set of the next-state function. Let σ be a set of the next-state function such that $\sigma : PQ(t^P) \times SQ(t^S) \times PX(t^P) \times SX(t^S) \rightarrow PQ(t^P) \times SQ(t^S)$, with $\rho = \rho_p \cup \rho_s \cup \rho_m$, where $\rho_p : PQ(t^P) \times \emptyset \times PX(t^P) \times \emptyset \rightarrow PQ(t^P) \times \emptyset$, $\rho_s : \emptyset \times SQ(t^S) \times \emptyset \times SX(t^S) \rightarrow \emptyset \times SQ(t^S)$, and $\rho_m \subset \sigma - (\rho_p \cup \rho_s)$.

d. $q_0 \in Q$ is called the initial state, and $W \subseteq Q$ is called the set of final states.

e. O is a finite set of output, $O \subseteq PX(t^P)$; and ψ is output function $\psi : Q \times I \to O$.

f. ξ is an outport function such that $\xi : PX \to SX$.

g. τ is an inport function such that $\tau : SX \to PX$.

Definition 1 *A virtual factory VF is called a* **Monitoring Machine,** *if we let $Q = Q - PQ(t^P)$, $I = I - PX(t^P)$, and $\rho = \rho_s$.*

Definition 2 *A virtual factory VF is called a* **Simulation Machine,** *if we let $Q = Q - SQ(t^s)$, $I = T[SX(t^s)]$, and $\rho = \rho_p$.*

Definition 3 *A virtual factory VF is called an* **Advanced Mixing Machine,** *if we let $PT \neq ST$.*

Definition 4 *A virtual factory VF is called an* **Advanced Testing Machine,** *if we let $\forall sx \in SX$, sx is at idle state.*

Let P_M be the performance function to be evaluated in the manufacturing system such that $P_M : CD \to R$, where CD is a set of control or design polices, including scheduling strategies, planning rules, layout planning, R is real number, and M is a performance criteria, e.g., throughput, completion time. In general, we denote the cost function for use of the variable VX as Cv, i.e., $Cv : VX \to R$, where R is real number. Consider two policies ζ and χ, where ζ is designed incorporating the virtual factory whereas χ is generated based on the traditional testbed, but both policies are designed to solve the same problem κ. Let VX^{χ} be the set of variables that is used to develop χ, and VX^{ζ} be the set of variables that is used to develop ζ.

Definition 5 *The* **gain of virtual factory** *relative to a traditional testbed B for problem κ can be defined as the following equation,*

$$G_{vf}(\kappa, B) = \frac{\sum_M P_M(\zeta) \sum_i Cv(VX_i^{\chi})}{\sum_M P_M(\chi) \sum_i Cv(VX_i^{\zeta})} \quad (2)$$

subject to

$$\forall vx_i \in VX, 0 \leq Cv(vx_i)$$

, and if there is an isomorphic relation z, $z(SX_i) = PX_i$ such that

$$Cv(PX_i) \leq Cv(SX_i)$$

. In general, librarian broker LB is defined as an information manager, that explores and extracts the information from various resources, e.g., statistic data generator, equations of queuing network, simulation generator, and historic or temporal data. The specifications SP represent the provided services from virtual factory.

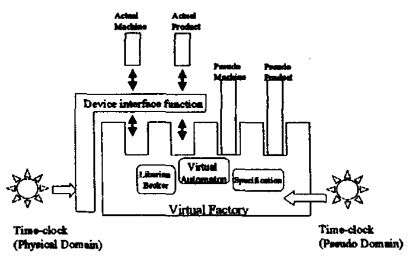

Figure 2: The block diagram of the operation model for a virtual factory

3 General Approach for System Operation

The block diagram of the operation model for a virtual factory is given in Fig 2. The device interface function DI provides data communication and transformation between the physical devices and the VF. Note that the physical devices consist of both hardware and software. There are two time-clock generators, TCG_a and TCG_b, where The TCG_a provides the time clock for refreshing the elements in the pseudo domain D_p, whereas the TCG_a provides the time clock for refreshing the elements in the physical domain D_s.

In general, a manufacturing system MS can be described as a 6-tuple set $MS =< MQ, I, \rho, \Gamma, O, \psi >$, where

a. MQ is a finite set of state, and I is a finite set of input.

b. Γ is a finite set of control policies or scheduling rules.

c. ρ is a set of the next-state function, $\rho : MQ \times I \times \Gamma \to MQ \times \Gamma$.

d. O is a finite set of output, and ψ is output function $\psi : MQ \times I \times \Gamma \to O$.

We assume that each rule or control policy to be evaluated can be represented as $R = (T_0, p_1, A_1, \cdots, p_n, A_n, T_n)$, where p_i is the ith procedure of R, T_{i-1} is an approximate starting time required by p_i, T_i is an approximate completion time, and $A_i \subset VX$ is the set of resources to be required for p_i. On the other hand, the initial set of resource variables, denoted as H, is set $H = \emptyset$. The local variable i is initialized such that $i = 0$. The general approach of model operation can be summarized in the following steps:

(1) Run the device interface function DI such that $MQ, I \in MS$, $SX = DI[I], SQ = DI[MQ]$.

(2) Check the state of each physical resource-state variable $SX(t^S)$, where $T_{i-1} \leq t^S \leq T_i$. If $SX(t^S) \in A_i$ and $SX(t^S)$ is at an idle state, then $H = H \cup \{SX(t^S)\}$.

(3) Tune the time clock TCG_a, check the state of each pseudo resource-state variable $PX(t^P)$, where $T_{i-1} \leq DI[t^P] \leq T_i$. If $PX(t^P) \in A_i$, then $H = H \cup \{PX(t^P)\}$.

(4) For each $px \in PX(t^P) \cap H$ and $sx \in SX(t^S) \cap H$, if there is an *isomorphic relation* between px and sx, then calculate the gain of virtual factory $G_{vf}(\kappa, B)$ for $H - \{px\}$ and $H - \{px\}$ individually, and to select the one which maximize the gain.

(5) Increase i, and do step (2)(3)(4) until the $i = n$

Notice that step (1) is just to map each physical device (e.g., a robot) to its counterpart in computer, and always keep the changes of Q and SQ synchronous. Although the proposed virtual factory is tied closely to the existent manufacturing system, it is our objective not to interfere the running of the manufacturing system. In Step (2), based on this consideration, SX can be used only when it is idle.

4 Experiment on An Example

4.1 Robotic Flexible Assembly System

In a robotic flexible assembly cell (RFAC), there are different types of equipment which cooperate with one another to assemble the parts sent into the system. Our experimental environment is a two-robot assembly cell , that is dedicated to assembling various types of mechanical parts serially sent in through a conveyor belt. The cell is composed of several pieces of hardware. The equipment structure is shown in Fig. 3. There are two products produced in our assembly cell.

Each product has four parts, respectively. The first product is assembled with only vertical insertion operations. The second product includes more complex operations. To assemble the second part with the base part for the second product, the robot needs to do vertical insertion and then a rotation to fasten the part with the base part. Sixteen parts can be placed randomly in the pallet of the loader. The loader will load the part onto the conveyor belt one at a time upon request. In the following section, the implementation of virtual factory at the flexible assembly cell is described. The modeling is explained first and the interaction between the components are discussed later.

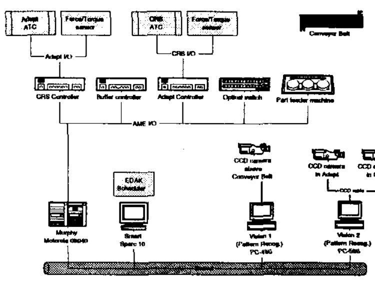

Figure 3: The equipment structure in an assembly cell

4.2 Virtual Factory Modeling in an Experimental RFAC

We model our virtual factory as multiple processes and libraries that work together.

- **Physical Domain** : The robotic assembly cell has two robots, respectively named as *Adept* and *CRS*. The robot *Adept* has four assembly sites and the one *CRS* has two. The other types of equipment are explained as in the previous subsection.
- **Pseudo Domain**:
  (1) The robotic assembly cell has four pseudo robots, respectively named as *Pseudo Adept*, *Pseudo CRS*, *Pseudo PUMA*, and *Pseudo A-arm*. The robot *Pseudo PUMA* has three assembly sites whereas the one *Pseudo A-arm* has two. Since *Pseudo Adept* is an emulator of *Adept* , it also means that there is an isomorphic relation between them. Similarly, *Pseudo CRS* is an emulator of *CRS*. Here, software modules represent the four pseudo robots.
  (2) In addition to the two products produced in our assembly system, there are two new pseudo products which will be produced virtually. These two new ones are similar to the two old ones, but the color of them are different. Each product also has four parts as mentioned earlier.
- **Virtual Automaton**:
  (1) The six robots are included in the set of virtual resource-state variables VX, namely, *Pseudo Adept*, *Pseudo CRS*, *Pseudo PUMA*, *Pseudo A-arm*, *Adept*, and *CRS*.
  (2) The variable VX also includes the parts of two products produced in our real assembly

cell and the parts of two new pseudo products which will be produced virtually.

(3) The set of next-state function ρ in the virtual automaton are partitioned into two parts, one behaves just like the real assembly cell, whereas the other is the set of relations between the pseudo resource-states and the pseudo system-states.

- **Librarian Broker**: We have three databases, namely, mathematics database, manufacturing database, and prediction database. They are in charge of storing the data needed for optimal feedback process, system operation process and prediction diagnosis process respectively. All information can be extracted through the librarian broker.

- **Specifications**: There are three processes, which are optimal feedback process, multi-objective negotiation process, and prediction diagnosis process. Each process can be perform independently.

- **Device Interface Function**: In this experiment, we use a commercial software package, *Wonderware InTouch*, which can view and interact with the execution of an entire operation through graphical representations of the production processes. We use it to perform communication and transformation between the real assembly cell and the virtual factory.

Various experiments are performed under different cases for the proposed model.

Case 1. *Virtual factory as a Monitoring Machine.* If all of the resources are limited to real physical resources, it means that we only use the *Adept* and *CRS*. The main functionality of virtual factory is monitoring to see whether the given task is executed properly in the system. In the experiment, part loading machine loads part into the real assembly cell randomly, and the robot will either assemble it or store it on buffer. We try to balance the number of parts for different types, but the order of the part is completely random. When each robot's assembly sites are full, any finished product will be removed.

Case 2. *Virtual factory as a Simulation Machine.* The experiment is performed on the pseudo robots, i.e., to use the pseudo robots : *Pseudo Adept* and *Pseudo CRS*. We replace the physical equipment with its emulator. In the experiment, part-loading machine loads part into the virtual factory according to some distribution processes given from the librarian broker.

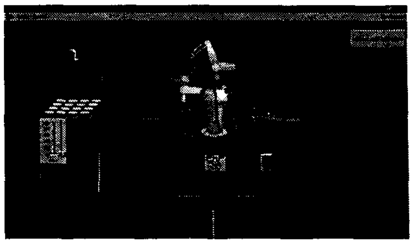

Figure 4: Graphical representation of an implemented virtual factory

VX	Pseudo Puma	Pseudo A-arm	Pseudo Adept
Cv	1.7	2.3	1.8
VX	Adept	CRS	Pseudo CRS
Cv	40.8	49.6	1.1
VX	Puma	A-arm	
Cv	70.7	60.3	

Table 1: Cost function for each variable

Case 3. *Virtual factory as an Advanced Mixing Machine.* It is a complicated experiment. In the experiment, *Pseudo PUMA*, *Pseudo A-arm*, *Adept* and *CRS* are running together. They form a four-robotic virtual assembly cell. While the part-loading machine loads part into the real assembly cell randomly, the pseudo parts of the two new pseudo products have also been loaded into the virtual factory according to some distribution processes.

Case 4. *Virtual factory as an Advanced Testing Machine.* Similar to the *Simulation Machine*, here the difference is to add to pseudo robots, *Pseudo PUMA* and *Pseudo A-arm*, and parts of two new pseudo products into the experiment.

4.3 Experimental Results

The graphical representation of the virtual factory for our experimental RFAC is shown in Fig. 4. It is implemented via a commercial software package *Wonderware InTouch* . Let the problem κ be the assembly problem described previously. The parameters that we will examine are the ***gain of virtual factory*** $G_{vf}(\kappa)$ given in equation (2). We assume the cost function Cv for each variable VX can be represented in Table 1, where the unit is \$/hour. We denote the number of finished products per second as our throughput. Let's first consider the assembly scheduling designed within a traditional testbed EMFAK (Event-driven model of flexible automation kernel [19]) which is clearly a greedy method. There the robots can assem-

	Case 3	Case4
Cost of VX (greedy)	221.4	133.9
Cost of VX (IG)	94.4	6.9
$P_{Throughput}(greedy)$	0.38	0.39
$P_{Throughput}(IG)$	0.31	0.16
Gain	1.91	7.96

Table 2: Experiment results for case 3 and case 4

ble the parts or subassemblies whenever they satisfy the geometric constraints. Contrastingly, the schedule designed in use of virtual factory is an algorithm with integer programming(IG). Because the functions of simulation and monitoring are the same as those for EMFAK, we have no gain in it. The results of gain we examined for case 3,4 explained in the previous sub-section are shown in Table 2. In Table 2, the greedy method is better than integer programming when the parts are placed randomly in the pallet of the loader. However, we will know that the value of gain becomes larger when more pseudo robots instead of real ones are used. We can also use the virtual factory to develop powerful schedules not limited to capacity of resources.

5 Concluding Remarks

The proposed operation model of the virtual factory is capable of coordinating with hybrid methods and targeted to build an open system such that it can be self-configured dynamically to respond to a changing market place. New pseudo resources can be added to form a new virtual environment, control policy designed by engineers will be evaluated before issued. This paper is one among very few that tries to represent the virtual factory in an analytic form so that many existing mathematical analyses can be applied, and the results demonstrate the intelligence and efficiency of the proposed prototyping testbed.

References

[1] John McGehee, John Hebley and Jack Mahaffey, "The MMST Computer-integrated Manufacturing System Framework," *IEEE Transaction on Semiconductor Manufacturing*, vol. 7, no. 2, pp. 107–116, 1994.

[2] Kazuaki Iwata,Masahiko Onosato, Koji Teramoto, Suguru Osaki, "Virtual Manufacturing Systems as Advanced Information Infrastructure for Integration Manufacturing Resources and Activities," *Annals of the CIRP* , vol. 46, no. 1, pp. 335–338, 1997.

[3] Muller Peter, Svan Engelen, Terlouw Pieter, de Vreede Gert-Jan, "Multimedia and co-operative work in new product development: A participatory approach to virtual prototyping," *Proceedings of the 1996 IEEE Conference on Emerging Technologies and Factory Automation* , pp. 777–782, 1996.

[4] Eccleston Michael, " Virtual prototyping," *Manufacturing Engineer* , vol. 75, no. 3, pp. 129–132, 1996.

[5] "Virtual Manufacturing User Workshop Final Report," *Virtual Manufacturing User Workshop*, Dayton, Ohio, 12-13, July, 1994.

[6] "Virtual Manufacturing Technical Workshop Final Report," *Virtual Manufacturing Technical Workshop*, Dayton, Ohio, 25-26, October, 1994.

[7] K.I.Lee, "Virtual Manufacturing Systems- a Test-Bed of Engineering Activities," *Annals of the CIRP* , vol. 46, no. 1, pp. 347–350, 1997.

[8] Geoffrey C. Orsak and Delores M. Etter, "Connecting the engineer to the 21st Century Through Virtual Teaming," *IEEE Transaction on Semiconductor Manufacturing*, vol. 7, no. 2, pp. 107–116, 1994.

[9] Di Stefano, Fazzino, F. Lo Bello, L. Mirabella, , "Virtual Lab: A Java application for distance learning," *Proceedings of the 1997 IEEE 6th International Conference on Emerging Technologies and Factory Automation,* , pp. 93–98, 1997.

[10] Vairaktarakis George, Hosseini Jamshid, Syam Siddhartha, , "Forming partnerships in a virtual factory," *Proceedings of the 1996 27th Annual Meeting of the Decision Sciences Institute,* , pp. 1394–1396, 1996.

[11] Macedo Julio, Hosseini Jamshid, Syam Siddhartha, "Intelligent reference models for designing virtual factories," *Proceedings of the 1996 IEEE International Engineering Management Conference Vancouver,* , pp. 346–350, 1996.

[12] Bailey Diane , Dessouky Maged , Verma Sushil, Adiga Sadashiv, Bekey George, Kazlauskas Ed , "Virtual factory for manufacturing education," *Proceedings of the 1997 6th Annual Industrial Engineering Research Conference,* , pp. 879–884, 1997.

[13] Bodner Douglas and Reveliotis Spiridon, "Virtual factories: An object-oriented simulation-based framework for real-time FMS control," *Proceedings of the 1997 IEEE 6th International Conference on Emerging Technologies and Factory Automation*, pp. 208–213, 1997.

[14] Henning Kathleen and Reveliotis Spiridon, "Simulation: the virtual factory," *Proceedings of the 1995 38th APICA International Conference and Exhibition Orlando*, pp. 12–14, 1995.

[15] Jain Sanjay, "Virtual factory framework: a key enabler for agile manufacturing," *Proceedings of the 1995 INRIA/IEEE Symposium on Emerging Technologies and Factory Automation* , pp. 247–258, 1995.

[16] Saraswat Krishna, "Adaptable IC manufacturing systems for the 21st century," *Proceedings of the 1993 Symposium D on Integrated Processing for Micro and Optoelectronics of the 1993 E-MRS Spring Meeting Conference Strasbourg* , pp. 131–137, 1994.

[17] Saraswat Krishna, "Programmable factory for IC manufacturing for the 21st century, " *Proceedings of the IEEE/SEMI International Semiconductor Manufacturing Science Symposium San Francisco* , pp. 2–6, 1993.

[18] Robert Orfali, Dan Harkey, and Jeri Edwards, *The Essential Distributed Objects Survival Guide.* John Wiley,Inc., 1987.

[19] H.-S. Huang, L.-C. Fu, and J. Y. jen Hsu, "Rapid setup of system control in a flexible automated production systems," *Proceedings IEEE International Conference on Robotics and Automation*, pp. 1517–1522, 1997.

[20] Ming-Hung Lin and L.-C. Fu, "Systematic creation and application for virtual factory with object oriented concept," *Proceedings IEEE International Conference on Robotics and Automation*, pp. 304–309, 1998.