Proceedings of the 2001 IEEE
International Conference on Robotics & Automation
Seoul, Korea · May 21-26, 2001

Model-based Control for Reconfigurable Manufacturing Systems

Kazushi Ohashi[*1] and Kang G. Shin[*2]

*1: Industrial Electronics & Systems Laboratory
Mitsubishi Electric Corp.
8-1-1 Tsukaguchi-Honmachi
Amagasaki, Hyogo 661-8661, Japan
ohashi@fas.sdl.melco.co.jp

*2: Real-Time Computing Laboratory
Eelectrical Engineering and Computer Science
The University of Michigan
Ann Arbor, MI 48109-2122, U.S.A.
kgshin@eecs.umich.edu

Abstract: The manufacturing industry cannot stay competitive and survive in today's market without agile adaptation to rapidly changing customers' demands. This in turn requires manufacturing systems to be reconfigurable for timely introduction of new products in the market. Unfortunately, at present, the system designers cannot systematically and completely manage their design data, because manufacturing systems have gradually become too large and too complicated to manage. In order to reconfigure and reuse H/W and S/W components in manufacturing systems, and improve the engineering environment of system control design, we propose a model-based control design using state transition diagrams and a general graph description, while taking reconfiguration and reuse of design data into account. We demonstrate the utility of the proposed approach using a real application.

1. Introduction

User demands for products tend to change rapidly. To respond to such demands, manufacturers must modify the design and functions of products, and change their production schedule to be competitive. This in turn requires to quickly reconfigure their manufacturing systems like a cell or line. It is, however, difficult and time-consuming to change system control programs due mainly to their ad hoc, and sometimes proprietary, nature. Especially, many parts of the System Control Design (SCD) data are inter-related to one another, thus requiring significant time and effort to identify and determine which parts to be modified. As a result, it is very costly to modify the control programs, or sometimes costlier than re-developing them from scratch.

Since graphs are very useful to represent data relations, many researchers attempted to model manufacturing systems using various graph descriptions, such as state transition diagrams (STDs) [2,3,4,6,7] and IDEF [5]. For accurate system descriptions, however, these models usually consist of an excessive number of states when they are used to model realistic (large) systems. To cope with this difficulty, some of these models adopt a hierarchical structure, but they have not taken into account system reconfiguration and reuse of H/W, S/W and SCD data, while its unit of reconfiguration is coarse in others. The authors of [3] and [4] presented a model based on CIM-OSA, which is composed of event flow and resource behaviors at a control specification level, and described with an object diagram and a Petri-Net. This model is decomposed according to manufacturing system functionalities. Researchers have taken into account reconfiguration of a manufacturing system and reuse of components of SCD data. But the unit of replaceable parts is a resource object. [5] provided a model consisting of message flows based on IDEF3 and a resource configuration graph. In this model sequence control flows and synchronization between resources in the message flow can be understood easily, but it didn't consider data reconfiguration. [6] presented a model consisting of a manufacturing system's structure graph and production routes based on Colored Petri-Nets. This model adopts templates and allows use of variables. It includes the concept of time to estimate runtime, but it is difficult to identify synchronization parts between resources when system size is large. [7] proposed a cooperation model between resource behaviors and a system control model using hierarchical Petri-Nets. The authors demonstrated the use of the analysis tools to verify the properties of their models, and analyzed them both quantitatively and qualitatively. Their concept is similar to ours in terms of a cyclic description of resource behaviors, synchronization descriptions and their compositions. However, in their model it is difficult to detect the boundaries between resource behaviors in the final composed model.

We previously attempted to model manufacturing systems with a cyclic STD model [2]. Each equipment behavior is described with a cyclic STD, and the model of a target system is configured to connect synchronized transitions between STDs of resources. This modeling method is very simple, but the definitions of system sequence control are included implicitly, and hence, it is difficult to modify part of the control. Moreover, the number of redundant descriptions explodes when the

The work reported in this paper was done during K. Ohashi's visit to RTCL and supported in part by DARPA under Grant F33615-00-1706 administered by the US Airforce Wright Laboratory.

number of combinations between resources increases in workpiece path. Neither did it give any information about the redundancy in behavior descriptions.

In this paper we propose a model-based control design method in which one can build a control system easily and efficiently using a reconfigurable SCD data model. We (1) make the SCD data model reconfigurable from the designers' standpoints and (2) reduce redundant state descriptions. This model is described by combining three sub-models corresponding to process flow, workpiece path, and resource behavior. Each of them is a cyclic STD or a graph description, and is hierarchically-structured.

The paper is organized as follows. Section 2 describes the architecture of reconfigurable manufacturing systems. Section 3 introduces the notation in graph descriptions, and Section 4 presents a real application example. Section 5 describes data structures, and the paper concludes with Section 6.

2. Architecture

We will focus on how to change the type, number and location of resources, and process flow in SCD as easily as possible for designers who do not completely manage the SCD data. Such a robust model requires the following basic functionalities:

(1) Data definition from designer's viewpoints,

(2) Hierarchical and componential model structures.

The main objects in system control design are:

(1.1) Manufacturing (M-) process flow,

(1.2) Selection of resources,

(1.3) Workpiece path/flow between resources,

(1.4) Resource behavior (e.g., sequence of parts of a motion or process control program),

(1.5) Collaboration and synchronization (C&S) between resources,

(1.6) Layout (resource and workpiece base locations).

The model is composed of the following functionalities:

(2.1) Hierarchal skeleton structure of SCD data,

(2.2) Parameterized descriptions of motion or process behavior,

(2.3) Variable and parameter list.

Fig. 1 shows the proposed SCD model which consists of three sub-models and combinations thereof:

- M-process flow (for (1.1))
- Workpiece path (for (1.2), (1.3) and (1.5))
- Resource behavior (for (1.4) and (1.5))

M-process flow and resource behavior are described with STD and workpiece path using a graph description. The behaviors of each resource are described with a cyclic STD and can be understood easily. Parts of data in these models can be linked to each other at a low level (e.g., a place, transition or node, which is an element of each graph description). These parts together form a SCD model, but these graph descriptions only show the data structure of SCD. Actual process or motion data are linked to places or nodes. Moreover, process and motion data are decomposed into parameterized descriptions, and variable and parameter list. This way, redundant descriptions are reduced in decomposing the SCD data into components at lower levels than a resource, and sharing the same parts. As a result, the local data can be reconfigured easily.

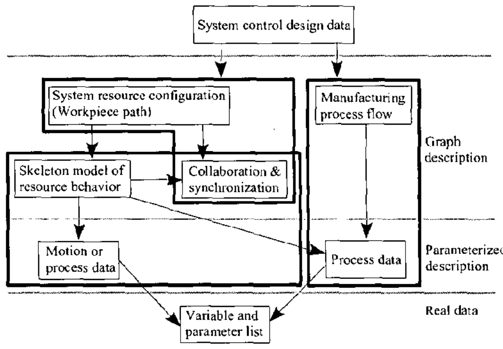

System control design data
System resource configuration (Workpiece path)
Manufacturing process flow
Skeleton model of resource behavior
Collaboration & synchronization
Motion or process data
Process data
Variable and parameter list
Graph description
Parameterized description
Real data

Fig. 1 The proposed model configuration

3. Notation and Modeling

3.1 M-process flow and resource behavior

M-process flow and resource behavior are described with STDs. A STD is constructed with a set of places, transitions and arrows. Graph models are constructed by connecting places and transitions with arrows. A place represents a part of manufacturing process and can have a hierarchical structure, and designers define and expand SCD data from an upper level to a lower level. Also, a place itself has data objects as shown in Fig. 2 where the line ended with a circle means a reference to another data object.

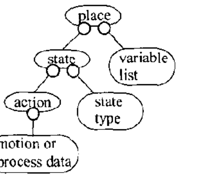


Fig. 2 Data structure of a place

A place only represents a node in graph descriptions. A combination of state, variable and parameter list represents the actual context of place, and finally identifies the place. "Variable list" means a list of variables defined by the user and can be defined locally, and "parameter list" means a list of resource variables obtained from the profiled data of resources. Also, a state is composed of action and state type data. Action data is motion or process execution data like a program and is parameterized. State type includes the data related to execution type of resources (type of M-process, execution code type of action, and so on). Likewise, SCD configuration and its actual detailed context arc separatcd. State is reused at other places in the same SCD or other SCDs. M-process flow and resource behavior are represented below as a flow of such places.

M-proccss flow

The M-process flow shows a sequential flow of M-processes for each workpiece as shown in Fig. 3.

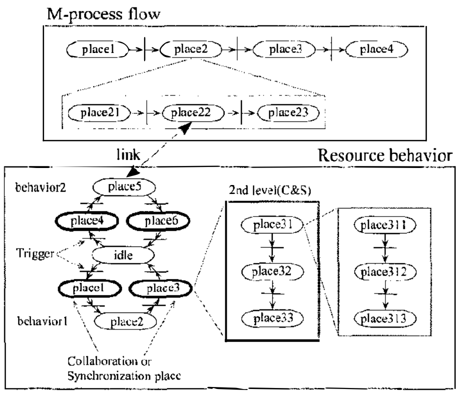


Fig 3 M-process flow and resource behavior

Each M-process is described by a place and can bc defined hierarchically. All terminal places, which are elementary M-processes, can be linked to any place of any resource behavior. But on this phase of M-process flow definition, actual resources are not decided exactly until it is linked to resource behaviors, and state and variable lists are defined. Then the M-process flow itself is independent of resources. Users can reuse the M-process flow for a system that has the same M-process flow by modifying the context of places.

Resource behavior

The resource behavior represents a sequence of resource actions and is shown in Fig. 3. The resource behavior is defined based on the cyclic STD of the upper-most level. The cyclic STD is a conceptual behavior described with basic actions like "Loaded", "Unloaded" and "Processing." Some resources can use the same cyclic STD at this level, and we can provide a certain basic cyclic STD as a template. A resource can own multiple STDs each of which corresponds to a motion or process control program. Cyclic STDs, which are not independent within a resource, share a start place, which is called the "Idle" place. Upon completion, every resource behavior always returns to the "Idle" place. A place can be described hierarchically in the same way as an M-process flow. The difference between M-process flow and resource behavior is that all cyclic STDs have more than onc collaboration and synchronization (C&S) place, which is represented by a thick-lined oval, at the top level of a resource behavior. A C&S place represents collaboration or synchronization with any C&S places of other resources (e.g., "Loaded" and "Unloaded" place) as shown in Fig. 6. There always exists a C&S place next to the "Idle" placc, becausc one behavior is selected by one of transitions between the "Idle" place and the activated place. If multiple actions need to be executed in parallel within a resource, it needs to own multiple behaviors, which do not share the "Idle" place.

Lower-level places of a C&S place are defined here or in the connection node of workpiece path. Details of collaboration and synchronization are defined at the second level of resource behavior as shown in Fig. 6.

3.2 Workpiece path

A workpiece path is specified with a graph description, and uses 4 types of nodes and 2 types of arrows as shown in Fig. 4.

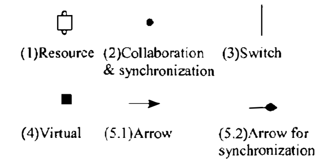

(1)Resource
(2)Collaboration & synchronization
(3)Switch
(4)Virtual
(5.1)Arrow
(5.2)Arrow for synchronization

Fig. 4 Nodes in workpiece path

A resource node represents a real or conceptual resource and each small circle corresponds to the C&S place of a resource behavior. When the node represents a conceptual resource, it can be expanded hierarchically. Then, small circles mean C&S places of a complex behavior to the outside world. In case of an actual resource, it corresponds to a resource behavior.

A C&S node is linked to the C&S places of collaborating or synchronizing resource behaviors. The normal arrow of (5.1) in Fig. 4 is used for collaboration to link nodes, and the diamond-shape arrow of (5.2) is used for synchronization. Designers can define C&S conditions bctween resource behaviors at a lower level through this node as shown in Fig. 6.

A switch node represents the branch or join of resource nodes, and is linked to the designer-defined conditions.

A virtual node is used in the "detailed" connection node and represents real input and output places of approximate resource nodes locally. Usually, a connection between resources is described conceptually at the upper-most level, and all actual connection nodes don't appear at the upper level of workpicce path. But when a conceptual resource node is expanded, the number of connection nodes is not restricted to one (e.g., when multiple resources are used in one process as shown in Fig. 5). As an example, a workpiece path among resource nodes and the lower level of connection node CO2 are given in Fig. 5. Two resource nodes are sensors, and three virtual nodes are C&S places of the equipment in the expanded node CO2. Also "PM11.1. Loaded" means <Resource name>.<Behavior number>. <Place name> in this figure.

The relation between workpiece path and resource behavior is shown in Fig. 6. Details of synchronization can be defined through a connection node.

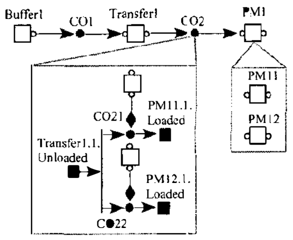


Fig. 5 An example of workpiece flow

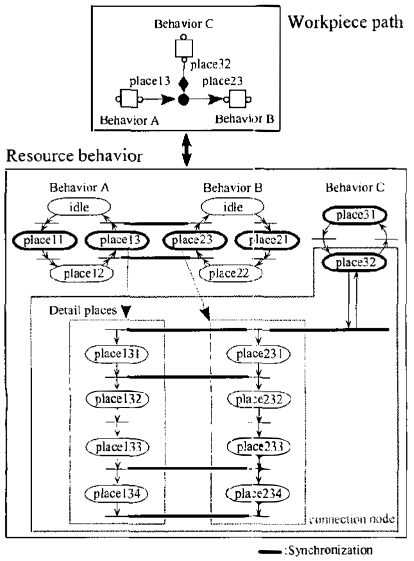


Fig. 6 Relation between workpiece path and resource behavior

4. An Example Application

The above model is applied to the SCD of a Plasma Display Panel (PDP) transfer cell as an example. This cell consists of elements of panels, three cassettes storing panels, two types of robots (GTR and RH-L3), two positioning tables (PT) (one of which is PT1 for input and the other is PT2 for output), and process machine (PM) evaporating MgO on panels. The graphical scene of this cell is given in Fig. 7 and its layout in Fig. 8. One cassette stores 15 panels. The types of work are to transfer panels from a cassette to PM, perform a chemical process on PM, and restore the panels to a cassette. PM draws in a carrier and 2 panels can be placed on the carrier each time. The workpiece path is cyclic as shown in Fig. 8. On PTs, the workpiece position is corrected. But position control is omitted to simplify the presentation. Each robot, PTs and PM have sensors, which check the existence of panels. The actual definition of each sub-model is explained below using the cell, and details are given using GTR and PT1.

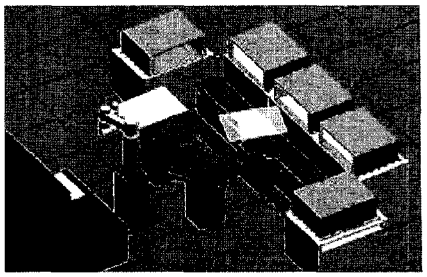


Fig. 7 PDP transfer cell (CG screen)

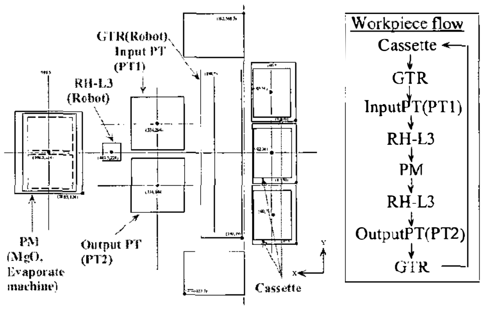


Fig. 8 Cell layout

4.1 M-process flow and resource behavior

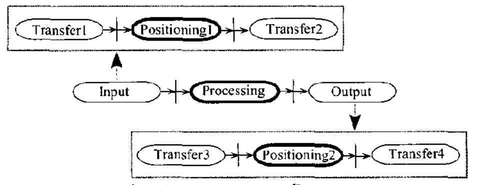


Fig. 9 M-process flow

This cell's process consists of transfer processes and a chemical process, and a process flow is described as shown in Fig. 9. The "Input" place shows how to transfer from buffer to PM, and the "Output" place represents a reverse-process for it. Each basic resource behavior can be described as shown in Fig. 10 where thick-lined ovals represent C&S places. The resource behavior is described with <Resource name>.<Behavior number> and * represents the corresponding resource or behavior number. "Wait*" or "Passive*" places denote the action of previous state being finished. An example of a place list of resource behaviors is shown in List 1. States can be shared by these multiple places.

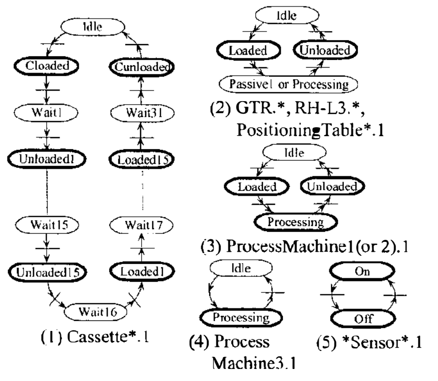


Fig. 10 Resource behaviors

List 1 Place list of GTR and PT1

	Behavior	Place		
		Level1	Level2	Level3
GTR	To PM (GTR.1)	Idle		
		Loaded (ToBuffer)	Move1	Move11(1-3)
				Move12(1-15)
			Approach1	
			Absorb1	
			Leave1	
		Passive1		
		Unloaded (To PT1)	Move2	
			Approach2	
			Release1	
			Leave2	
			Wait1	
Positioning Table1	ToPM (Positioning Table1.1)	Idle		
		Loaded	Wait1	
			Rotate0	
		Positioning		
		Unloaded	Wait2	
			Rotate90	

4.2 Workpiece path

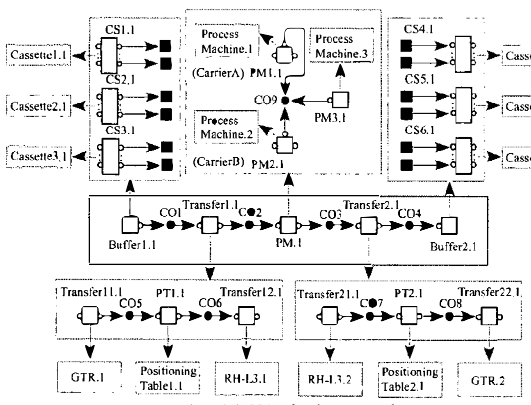
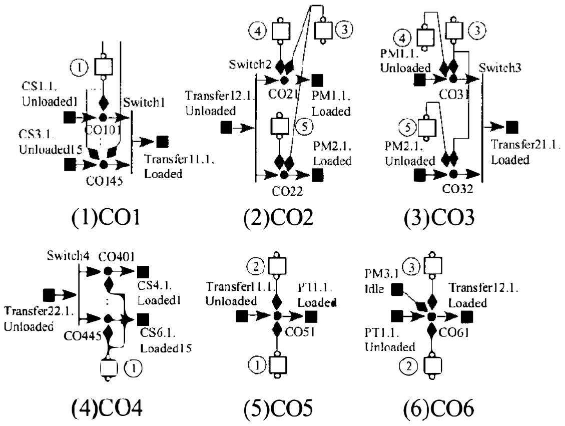


Fig. 11 Workpiece path

Workpiece path can be described hierarchically as shown in Fig. 11 where CO* is a C&S node's name. C&S node details are described as shown in Fig. 12 (CO7, 8 and 9 are omitted.). In an equipment resource node, the left-hand side circle represents the "Loaded" place and the right-hand side circle does the "Unloaded" place. In a sensor resource node, the upper-side circle represents the "Off" place and the bottom-side circle does the "On" place. Conditions stored in switch nodes are described as follows for the case of CO1 (Place1 and Place2 are variables):

Type, Priority

CS1.1.Unloaded1	// Priority 1
CS1.1.Unloaded2	// Priority 2
:	

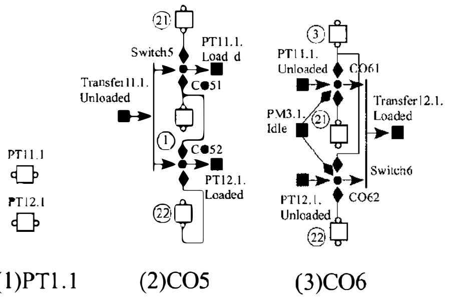


1. GTRSensor.1 2. PT1Sensor.1 3. RH-L3Sensor.1
4. PMSensor1.1 5. PMSensor2.1 6. PT2Sensor.1

Fig. 12 Detailed C&S nodes

List 2 An example of synchronization data on CO51

Transfer11	PT1	GTRSensor	PT1Sensor
Move2	Wait1		
Approach2			
Release1			
Leave2		<-N:Off	<-N:On
Wait1	Rotate0		



21: PT11.1.Sensor 22: PT12.1.Sensor

Fig. 13 In a case of changing number of input PT

An example of synchronization condition data included in C&S nodes (CO5) is shown in List 2. Here the lines above or below a place name represent transitions, and thick lines denote collaboration. In sensor places, the arrow is synchronization, and <-N means synchronization at the next transition of a pointed place.

If designers want to change the number of Input PTs to two (PT11 and PT12), they need to modify SCD data by changing resource node PT1 and C&S nodes, CO5 and CO6, as shown in Fig. 13. Then, the states used in PT1 can be reused in PT11 and PT12. State data is omitted here, but one can easily infer that the parameterized data on actions can be reused without increasing the amount of data too much.

5. Data Structure

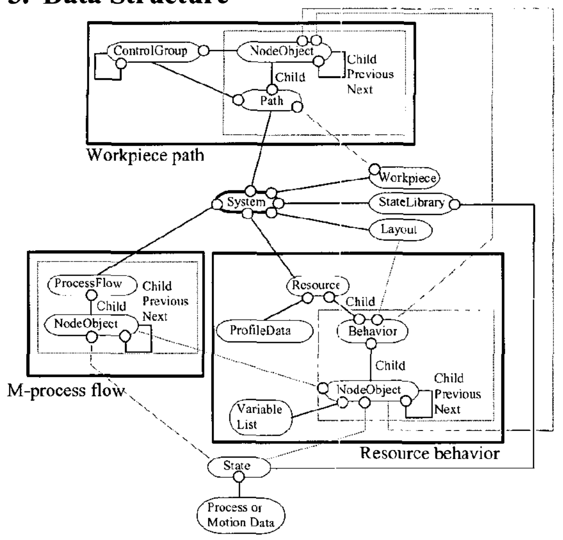


Fig. 14 Data structure of SCD model

The data structure to realize the proposed model is shown in Fig. 14. Ovals represent data objects based on object-oriented methods. The line ended with a circle denotes the reference of object data. Especially, gray lines represent the relations between sub-models and main data objects. Previous, next and child in the figure mean to own data lists (e.g., hierarchy). Places and nodes of sub-models are linked and an SCD model is constructed by their combination. We only give the meaning of some special data classes as follows.

• System:	Top object in SCD data
• ProfileData:	Resource parameter list
• ControlGroup:	Administration of resource groups
• Behavior:	Cyclic STD of resource
• NodeObject:	Super class of every node (place, transition, resource node, etc.)

6. Conclusion

We proposed a modeling method for supporting the reconfiguration of a manufacturing system. Specifically, we presented a system control design model and described how to define the data as the first step. The model is then composed of 3 sub-models based on workpiece path, manufacturing process flow, and resource behavior. These sub-models are described with general graph descriptions and state transition diagrams, and decomposed into multiple data levels, such as skeleton data configuration, parameterized description, and variable and parameter lists. By separating control structure from actual data and components of the structure based on the designer's viewpoints, this model enables the data to be modified only locally and facilitates efficient reconfiguration of manufacturing systems. Also part of skeleton of control configurations, and process, motion and synchronization data can be reused. A Plasma Display Panel transfer cell is modeled as an example.

As future work, we are planning to

(1) verify the state-level system behavior by simulation using the model, and refine the model,
(2) investigate how to reconfigure the data and make execution data for the model., and
(3) expand the model to be able to deal with timing constraints, complex distributed control systems and description of exceptions.

Reference

[1] J. L. Peterson., "PETRI NET THEORY AND THE MODELING OF SYSTEMS", Prentice-Hall, 1981.

[2] K. Furusawa, T. Yoshikawa, and K. Ohashi, "Development of an Integrated Workcell Design and System Engineering (II) -Applying Line-Model to Real Systems-", *Proceedings of the 41th Annual Conference of ISCIE*, pp. 337-338, 1997 (Japanese)

[3] R. P. Monfared, and R. H. Weston., "The re-engineering and reconfiguration of manufacturing cell control systems and reuse of their components", *Proc. Instn. Mech. Engrs.*, vol. 211, pp. 495-508, 1997

[4] J. M. Edwards and M. Wilson, "A top down and bottom up approach to manufacturing enterprise engineering using the function view", *Int. J. Computer Integrated Manufacturing*, vol. 11, no. 4, pp. 364-376, 1998

[5] H. Cho and I. Lee, "Integrated framework of IDEF modeling methods for structured design of shop floor control systems", *Int. J. Computer integrated manufacturing*, vol. 12, no. 2, pp.113-128, 1999

[6] A. Zimmermann, S. Bode, and G. Hommel., "Performance and Dependability Evaluation of Manufacturing Systems Using Petri Nets", "Manufacturing and Petri Nets" of *17th Int. Conf. on Application and Theory of Petri Nets*, pp.235-250, 1996

[7] M. Heiner, P. Deussen., and J. Spranger., "A Case Study in Developing Control Software of Manufacturing Systems with Hierarchical Petri Net", "Manufacturing and Petri Nets" of *17th Int. Conf. on Application and Theory of Petri Nets*, pp.177-196, 1996