19

Design and Implementation of Hierarchical and Distributed Control for Robotic Manufacturing Systems using Petri Nets

Gen'ichi Yasuda
Nagasaki Institute of Applied Science
Japan

1. Introduction

To realize control systems for flexible manufacturing systems, it is necessary to provide effective tools for describing process specifications and developing control algorithm in a clear and consistent manner. Conventionally, process modeling at the conceptual level is based on flowcharts, time diagrams, state machine diagrams, etc., and the control algorithms at the execution level are generally developed with relay ladder diagrams or procedural languages for programmable logic controllers. However, in the area of real-time control of discrete event manufacturing systems the main problems that the system designer has to deal with are concurrency, synchronization, and resource sharing problems. For this class of problems, Petri nets have intrinsic favorable qualities and it is very easy to model sequences, choices between alternatives, rendezvous and concurrent activities by means of Petri nets. The network model can describe the execution order of sequential and parallel tasks directly without ambiguity. Moreover, the formalism allowing a validation of the main properties of the Petri net control structure (liveness, boundedness, etc.) guarantee that the control system will not fall immediately in a deadlocked situation. In the field of flexible manufacturing systems, the last aspect is essential because the sequences of control are complex and change very often. Furthermore, a real-time implementation of the Petri net specification by software called a token player can avoid implementation errors, because the specification is directly executed by the token player and the implementation of these control sequences preserves the properties of the model. In this approach, the Petri net model is stored in a database and the token player updates the state of the database according to the operation rules of the model. For control purposes, this solution is very well suited to the need of flexibility. When the control sequences change only the database must be changed.

Motivated by the potential that Petri nets have for modeling parallel and concurrent processes, some techniques derived from Petri nets have been successfully introduced as an effective tool for describing control specifications and realizing the control in a uniform manner (Komoda, et al. 1984), (Murata, et al. 1986). However, in the field of flexible manufacturing systems, the network model becomes complicated and it lacks for the readability and comprehensibility. Therefore, the flexibility and expandability are not

satisfactory in order to deal with the specification change of the manufacturing system. Depending on the size and complexity of the system, these models can become very difficult to understand and treat. Despite the advantages offered by Petri nets, the synthesis, correction, updating, etc. of the system model and programming of the controllers are not simple tasks. Furthermore, the increasing use of robots and other automated machines has generated control software written in different levels and it is impossible for one person to understand all the control specifications in a large and complex manufacturing system.

The overall structure of the working area in a large and complex manufacturing system consists of one or more lines, each line consists of one or more stations, and each station (shop or cell) consists of one or more machines such as robots and intelligent machine tools. Inside of a cell, machines execute cooperation tasks such as machining, assembling and storing. Inside of a shop, cells cooperate mutually and execute more complicated tasks. Furthermore each machine consists of several motion elements. A task executed by a robot or an intelligent machine tool can be seen as some connection of more detailed subtasks. For example, transferring an object from a start position to a goal position is a sequence of the following subtasks; moving the hand to the start position, grasping the object, moving to the goal position, and putting it on the specified place. Thus the manufacturing system handles complicated tasks by dividing a task hierarchically in this structure, which is expected to be effective in managing cooperation tasks executed by great many machines or robots (Hatvany, 1985). Since in the large and complex systems, the controllers are geographically distributed according to their physical (hardware) structure, it is desirable to realize the hierarchical and distributed control. Distributed implementation can bring about the simplicity, easy modification, and relatively efficient execution of the Petri net based control scheme because the size of each Petri net model is not so large. Conventional Petri net based control systems were implemented based on an overall system model. The hierarchical and distributed control for large and complex manufacturing systems has not been implemented so far. If it can be realized by Petri nets, the modeling, simulation and control of large and complex discrete event manufacturing systems can be consistently realized by Petri nets (Lee, 2006), (Queiroz, 2000).

In a manufacturing system such as workstation, robots often must interact with automated machines, other robots or operators (Hoermann, 1989), (Merabet, 1986), (Jones, et al. 1989). These external processes are executing in parallel and asynchronously. It is not possible to predict exactly when events of interest to the robot program may occur. The signal lines are supported by most robot systems to coordinate multiple robots and machines, but this is a very limited form of communication between processes. Sophisticated tasks require efficient means for coordination and for sharing the state of the system between processes. The programming system should provide a mechanism for specifying the behavior of systems more complex than a single robot. Existing robot programming systems are based on the view of a robot system as a single robot weakly linked to other machines. Many machines may be cooperating during a task. The interactions between them may be highly dynamic. No existing robot programming system adequately deals with all of these interactions. No existing computer language is adequate to deal with this kind of parallelism and real-time constraints (Silva, 1990).

In this chapter, the author presents a methodology by extended Petri nets for hierarchical and distributed control of large and complex robotic manufacturing systems, to construct the control system where the cooperation of each controller is implemented within a

coordinator mechanism so that the behavior of the overall system is not deteriorated and the task specification is completely satisfied. Based on the hierarchical and distributed structure of the system, the Petri net based specification procedure is a top-down approach from the conceptual level to the detailed level of the discrete event manufacturing systems. The macro representation of the system is broken down to generate the detailed Petri nets at the machine control level. Then the Petri nets are decomposed and assigned to the machine controllers. The proposed procedure is demonstrated through an example of robotic manufacturing cell.

2. Modeling of Robotic Manufacturing Systems using Petri Nets

A manufacturing process is characterized by the flow of workpieces or parts, which pass in ordered form through subsystems and receive appropriate operations. From the viewpoint of discrete event process control, an overall manufacturing process can be decomposed into a set of distinct activities (or events) and conditions mutually interrelated in a complex form. An activity is a single operation of a manufacturing process executed by a subsystem. A condition is a state in the process such as machine operation mode. To represent discrete event manufacturing systems a modeling technique was derived from Petri nets. Considering not only the modeling of the systems but also the actual manufacturing system control, the guarantee of safeness and the additional capability of input/output signals from/to the machines are required (Masuda, et al. 1981). From the viewpoint of real-time control, controlling a process consists in driving its devices from a state to another one. In discrete event system control domain, three steps are necessary for each evolution of the process: first, the control system sends a request to the process actuators, second, the process evolves according to the request and, at the end of the evolution, returns an execution report to the control system, third, the control system updates the states of the control models according to the report; then it is ready to send another request. This outline points out that, each time the control system sends a request to the process, it must wait for the associated report. Thus the function of the description method based on common Petri net technique is enhanced it may not move the tokens until the operation, shown by each place, is finished, even if the marking satisfies the ignition condition.
The extended Petri net consists of the following six elements: (1)Place (2)Transition (3)Directed arc (4)Token (5)Gate arc (6)Output signal arc. A transition is enabled if and only if it satisfies all the following conditions:

(1) It does not have any output place filled with a token.
(2) It does not have any empty input place.
(3) It does not have any internal permissive arc signaling 0.
(4) It does not have any internal inhibitive arc signaling 1.

An enabled transition may fire when it does not have any external permissive arc signaling 0 nor any external inhibitive arc signaling 1. The firing of a transition removes tokens from all its input places and put a token in each output place connected to it. The assignment of tokens into the places of a Petri net is called marking and it represents the system state. In any initial marking, there must not exist more than one token in a place. According to these rules, the number of tokens in a place never exceeds one, thus, the Petri net is essentially a safe graph.

If a place has two or more input transitions or output transitions, these transitions may be in conflict for firing. When two or more transitions are firable only one transition should fire using some arbitration rule. By the representation of the activity contents and control strategies in detail, features of discrete event manufacturing systems such as ordering, parallelism, asynchronism, concurrency and conflict can be concretely described through the extended Petri net.

3. Design of Hierarchical and Distributed Control

The overall procedure for the design and implementation of hierarchical and distributed control is summarized as shown in Fig. 1. A global, conceptual Petri net model is first chosen which describes the aggregate manufacturing process. At the conceptual level each task specification is represented as a place of the Petri net, where the activity of each equipment is also represented as a place. Based on the hierarchical approach, the Petri net is translated into detailed subnets by stepwise refinements from the highest system control level to the lowest machine control level (Suzuki, 1983). At each step of detailed specification, some parts of the Petri net, places, are substituted by a subnet in a manner, which maintains the structural properties. Then, the detailed Petri net is decomposed into subnets, which are executed by each machine controller.

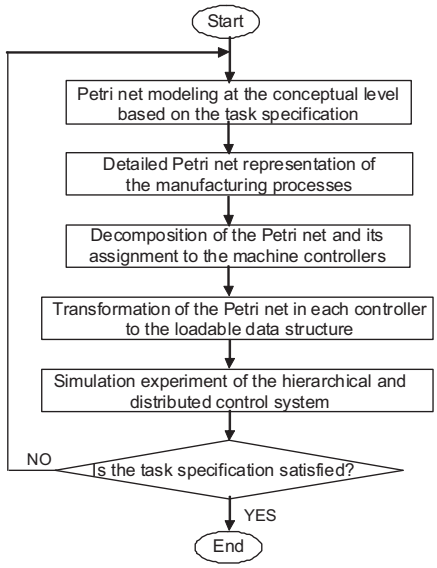


Fig. 1. Flow chart of Petri net based implementation of hierarchical and distributed control

In the decomposition procedure, a transition may be divided and distributed into different machine controllers as shown in Fig.2. The machine controllers should be coordinated so that these transitions fire in union.

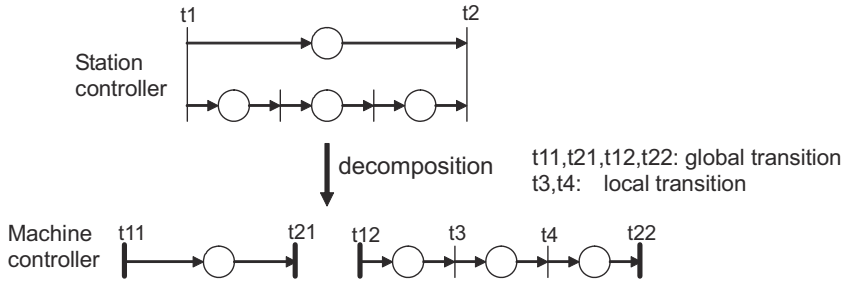


Fig. 2. Decomposition of transition

Decomposed transitions must function in union, that is, the aggregate behavior of decomposed subnets should be the same as that of the original Petri net. Decomposed transitions are called global transitions, and other transitions are called local transitions. By the Petri net model, the state of the discrete event system is represented as the marking of tokens, and firing of any transition brings about change to the next state. So the firing condition and marking before decomposition should be the same as those after decomposition. The firability condition and external gate condition of a transition before decomposition are described as follows:

$$t_j(k) = \bigcap_{m=1}^{M} p_{j,m}^{I}(k) \wedge \bigcap_{n=1}^{N} \overline{p_{j,n}^{O}(k)} \wedge \bigcap_{q=1}^{Q} g_{j,q}^{IP}(k) \wedge \bigcap_{r=1}^{R} \overline{g_{j,r}^{II}(k)} \tag{1}$$

$$g_j^{E}(k) = \bigcap_{u=1}^{U} g_{j,u}^{EP}(k) \wedge \bigcap_{v=1}^{V} \overline{g_{j,v}^{EI}(k)} \tag{2}$$

where,

M : input place set of transition j

$p_{j,m}^{I}(k)$: state of input place m of transition j at time sequence k

N : output place set of transition j

$p_{j,n}^{O}(k)$: state of output place n of transition j at time sequence k

Q : internal permissive gate signal set of transition j

$g_{j,q}^{IP}(k)$: internal permissive gate signal variable q of transition j at time sequence k

R : internal inhibitive gate signal set of transition j

$g_{j,r}^{II}(k)$: internal inhibitive gate signal variable r of transition j at time sequence k

U : external permissive gate signal set of transition j

$g^{EP}_{j,u}(k)$: external permissive gate signal variable u of transition j at time sequence k

V : external inhibitive gate signal set of transition j

$g^{EI}_{j,v}(k)$: external inhibitive gate signal variable v of transition j at time sequence k

The addition or removal of a token of a place is described as follows:

$$p^{I}_{j,m}(k+1) = p^{I}_{j,m}(k) \wedge \overline{(t_j(k) \wedge g^{E}_{j}(k))} \tag{3}$$

$$p^{O}_{j,n}(k+1) = p^{O}_{j,n}(k) \vee (t_j(k) \wedge g^{E}_{j}(k)) \tag{4}$$

The firability condition of a transition after decomposition is described as follows:

$$t_{jsub}(k) = \bigcap_{m=1}^{Msub} p^{I}_{jsub,m}(k) \wedge \overline{\bigcap_{n=1}^{Nsub} p^{O}_{jsub,n}(k)} \wedge \bigcap_{q=1}^{Qsub} g^{IP}_{jsub,q}(k) \wedge \overline{\bigcap_{r=1}^{Rsub} g^{II}_{jsub,r}(k)} \tag{5}$$

$$g^{E}_{jsub}(k) = \bigcap_{u=1}^{Usub} g^{EP}_{jsub,u}(k) \wedge \overline{\bigcap_{v=1}^{Vsub} g^{EI}_{jsub,v}(k)} \tag{6}$$

From eq.(1) and eq.(5),

$$t_j(k) = \bigcap_{sub=1}^{S} t_{jsub}(k) \tag{7}$$

From eq.(2) and eq.(6),

$$g^{E}_{j}(k) = \bigcap_{sub=1}^{S} g^{E}_{jsub}(k) \tag{8}$$

where,

S : total number of subnets

M_{sub} : input place set of transition $jsub$ of subnet sub

$p^{I}_{jsub,m}(k)$: state of input place m of transition $jsub$ of subnet sub at time sequence k

N_{sub} : output place set of transition $jsub$ of subnet sub

$p^{O}_{jsub,n}(k)$: state of output place n of transition $jsub$ of subnet sub at time sequence k

The addition or removal of a token of a place connected to a decomposed transition is described as follows:

$$p^{I}_{jsub,m}(k+1) = p^{I}_{jsub,m}(k) \wedge \overline{(t_j(k) \wedge g^{E}_{j}(k))} \tag{9}$$

$$p^{O}_{jsub,n}(k+1) = p^{O}_{jsub,n}(k) \vee (t_j(k) \wedge g^{E}_{j}(k)) \tag{10}$$

Consequently it is proved that the firability condition of the original transition is equal to AND operation of firability conditions of decomposed transitions. If and only if all of the decomposed transitions are firable, then the global transitions are firable. The coordinator program has been introduced to coordinate the decomposed subnets so that the aggregate behavior of decomposed subnets is the same as that of the original Petri net.
In case that a transition in conflict with other transitions is decomposed, these transitions should be coordinated by the station controller (Fig. 3).

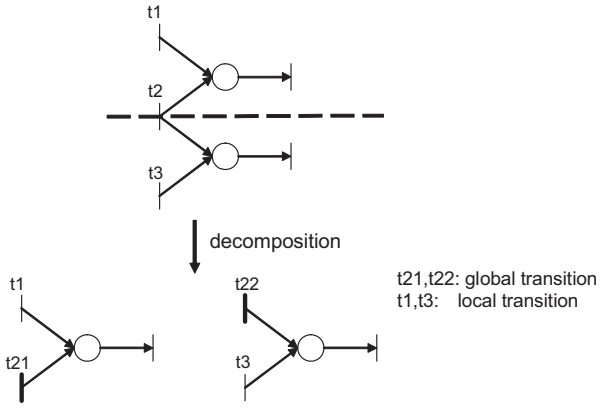


Fig. 3. Decomposition of transition in conflict

The Petri net based control structure with introduction of coordinator is shown in Fig. 4. The control software is distributed into the station controller and machine controllers. The station controller is composed of the Petri net based controller and the coordinator. The conceptual Petri net model is allocated to the Petri net based controller for management of the overall system. The detailed Petri net models are allocated to the Petri net based controllers in the machine controllers. Each machine controller directly monitors and controls the sensors and actuators of its machine.
The control of the overall system is achieved by coordinating these Petri net based controllers. System coordination is performed through communication between the coordinator in the station controller and the Petri net based controllers in the machine controllers as the following steps.

(1) When each machine controller receives the start signal from the coordinator, it tests the firability of all transitions in its own Petri net, and sends the information on the global transitions and the end signal to the coordinator.
(2) The coordinator tests the firability of the global transitions, arbitrates conflicts among global and local transitions, and sends the names of firing global transitions and the end signal to the machine controllers.
(3) Each machine controller arbitrates conflicts among local transitions using the information from the coordinator, generates a new marking, and sends the end signal to the coordinator.
(4) When the coordinator receives the end signal from all the machine controllers, it sends the output command to the machine controllers.
(5) Each machine controller outputs the control signals to its actuators.

Multilevel hierarchical and distributed control for large and complex manufacturing systems can be constructed such that the control system structure corresponds to the hierarchical and distributed structure of the general manufacturing system. The overall system is consistently controlled, such that a coordinator in a layer coordinates one-level lower Petri net based controllers and is coordinated by the one-level upper coordinator.

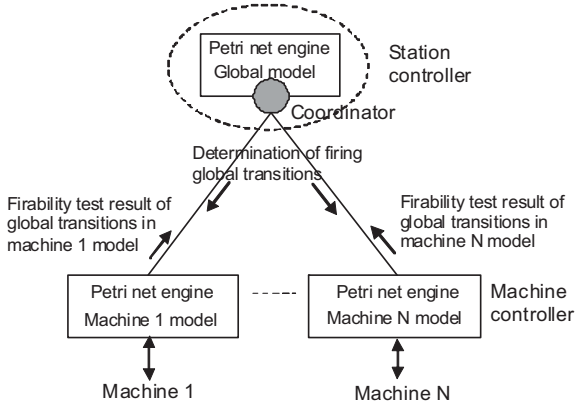


Fig. 4. Petri net based control structure with coordinator

4. Implementation of Example Control System

The basic procedures of modeling and decomposition of robotic manufacturing systems are shown through a simple example. The manufacturing system has two robots, a machining center, and two conveyors, where one is for carrying in and the other is for carrying out. The example manufacturing system is shown in Fig. 5. The main execution of the system is indicated as the following task specifications:

(1) A workpiece is carried in by the conveyor CV1.
(2) The robot R1 loads the workpiece to the machining center MC.
(3) The machining center MC processes the workpiece.

(4) The robot R2 unloads the workpiece from the machining center and places it on the conveyor CV2.
(5) The workpiece is carried out by the conveyor CV2.

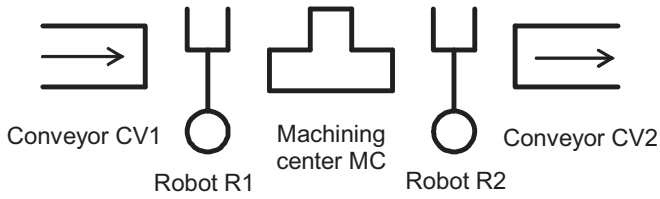


Fig. 5. Example of robotic manufacturing system

A global, conceptual Petri net model is first chosen which describes the aggregate manufacturing process. The places which represent the subtasks indicated as task specifications are connected by arcs via transitions in the specified order corresponding to the flow of subtasks and a workpiece. The places representing the existence of machines are also added to connect transitions which correspond to the beginning and ending of their subtasks. Thus at the conceptual level the manufacturing process is represented as shown in Fig. 6. In this step, if necessary, control conditions such as the capacity of the system between the respective subtasks must be connected to regulate the execution of the Petri net. Next, each place representing a subtask at the conceptual level is translated into a detailed subnet. Fig. 7 shows the detailed Petri net representation of loading, processing and unloading in Fig. 6.
For the manufacturing system, various control structures can be considered as shown in Figs. 8 and 9. In the centralized control system, the station controller directly controls all the machines using detailed Petri net (Fig. 8). For an example structure of hierarchical and distributed control composed of one station controller and three machine controllers (Fig. 9(b)), the Petri net executed in each machine controller is shown in Fig. 10.

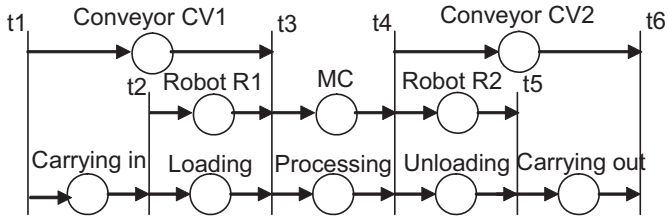
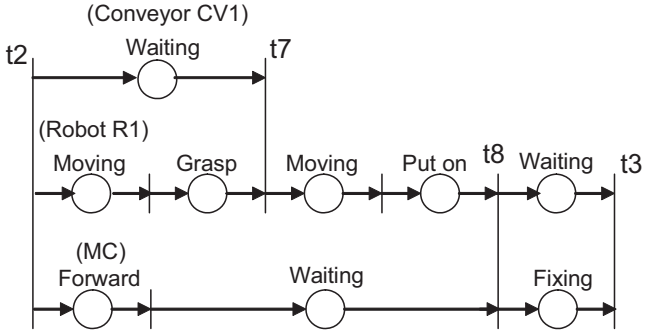
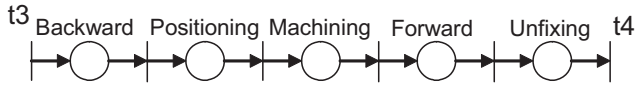


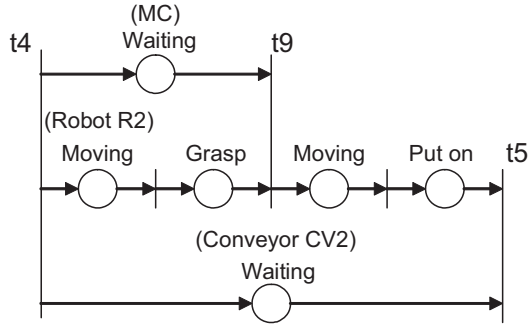
Fig. 6. Petri net representation of the example system at the conceptual level



(a) Loading



(b) Processing



(c) Unloading

Fig. 7. Detailed Petri net representation of subtask operations

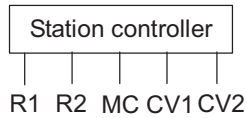
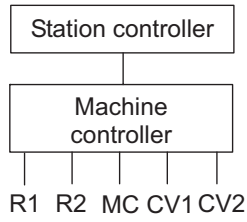
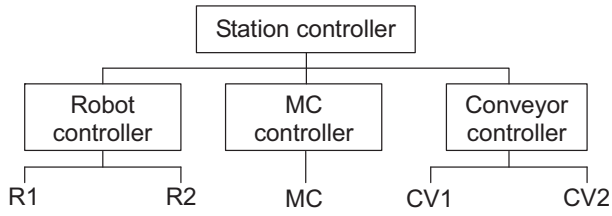


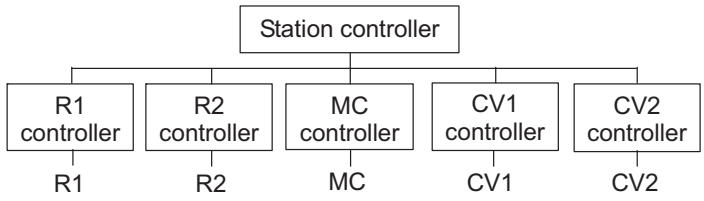
Fig. 8. Example structure of centralized control system



(a)

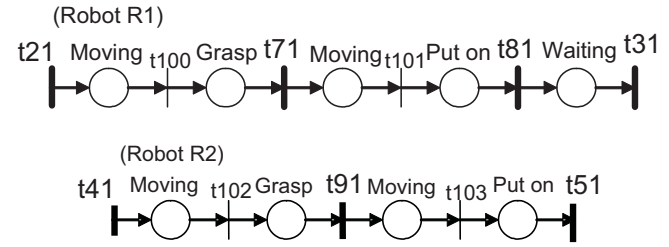


(b)

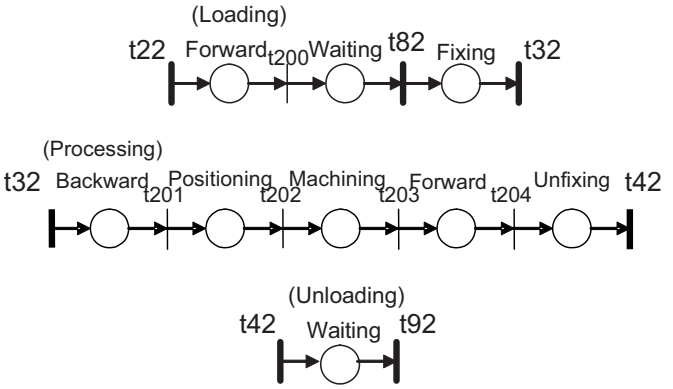


(c)

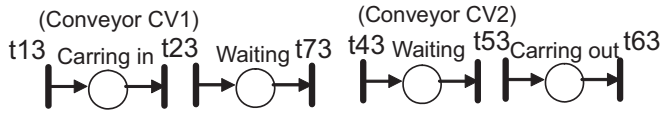
Fig. 9. Example structures of hierarchical and distributed control system



(a) Robot controller



(b) MC controller



(c) Conveyor controller

Fig. 10. Petri net representation of machine controllers
(▌: global transition, ▏: local transition)

For the example system, the hierarchical and distributed control system has been realized using a set of PCs. Each machine controller is implemented on a dedicated PC. The robot controller executes robot motion control through the transmission of command. The station controller is implemented on another PC. Communications among the controllers are performed using serial communication interfaces. The machine controllers control two conveyors or robots, so control software on each PC is written using multithreaded programming.
The names of global transitions and their conflict relations are loaded into the coordinator in the station controller. The connection structure of a decomposed Petri net model and conflict relations among local transitions are loaded into the Petri net based controller in a machine controller. In the connection structure, a transition of a Petri net model is defined using the names of its input places and output places; for example, t1-1=b1-1, -b1-11, where the transition no.1 (t1-1) of Robot controller (subsystem no.1) is connected to the input place no.1 and the output place no.11. For the distributed control system shown in Fig. 9(b), information inputted to the loader is as follows.

(Robot controller)
t1-21=-b1-3 t1-100=b1-3,-b1-4 t1-71=b1-4, -b1-5 t1-101=b1-5, -b1-6
t1-81=b1-6, -b1-7 t1-31=b1-7 t1-41=-b1-13 t1-102=b1-13,-b1-14
t1-91=b1-14, -b1-15 t1-103=b1-15, -b1-16 t1-51=b1-16

(MC controller)
t2-22=-b2-8 t2-200=b2-8, -b2-9 t2-82=b2-9, -b2-10 t2-32=b2-10, -b2-12
t2-42=b2-11, -b2-17 t2-92=b2-17

(Conveyor controller)
t3-13=-b3-1 t3-23=b3-1, -b3-2 t3-73=b3-2 t3-43=-b3-12
t3-53=b3-12, -b3-19 t3-63=b3-19

Using the names of transitions in the subsystems, global transitions are defined; for example, G2: t0-2, t1-21, t2-22, t3-23 indicates that the global transition G2 is composed of the transition no.2 of Station controller (subsystem no.0) , the transition no.21 of Robot controller, the transition no.22 of MC controller (subsystem no.2), and the transition no.23 of Conveyor controller (subsystem no.3). Then, the coordinator information for the example distributed control system is as follows.

G1: t0-1, t3-13 G2: t0-2, t1-21, t2-22, t3-23 G3: t1-71, t3-73
G4: t1-81, t2-82 G5: t0-3, t1-31, t2-32 G6: t0-4, t1-41, t2-42, t3-43
G7: t1-91, t2-92 G8: t0-5, t1-51, t3-53 G9: t0-6, t3-63

By executing the coordinator and Petri net based controllers algorithms based on loaded information, simulation experiments have been performed. Experimental results show that the decomposed transitions fire at the same time as the original transition of the detailed Petri net of the whole system task. Firing transitions and marking of tokens can be directly observed on the display at each time sequence using the Petri net simulator.

5. Conclusions

A methodology to construct hierarchical and distributed control systems, which correspond to the hardware structure of manufacturing systems, has been presented. The overall control structure composed of one station controller and several machine controllers has been implemented using a communication network of PCs for the example robotic manufacturing system. The conceptual Petri net model of task specification is allocated to the Petri net based controller in the station controller for management of the overall system. The detailed Petri net models are allocated to the Petri net based controllers in the machine controllers. By introduction of the coordinator, the Petri net based controllers are arranged according to the hierarchical and distributed nature of the manufacturing system. Experimental results show that the decomposed transitions fire at the same time as the original transition of the detailed Petri net of the whole system task. The Petri net model in each Petri net based machine controller is not so large and easily manageable. Thus, modeling, simulation and control of large and complex manufacturing systems can be performed consistently using Petri nets.

6. References

Hatvany, J. (1985). Intelligence and cooperation in heterarchical manufacturing systems. Robotics and Computer Integrated Manufacturing, Vol. 2, No. 2, 101-104

Hoermann, A. (1989). A Petri net based control architecture for a multi-robot system, Proceedings of IEEE International Symposium on Intelligent Control, 493-498

Jones, A. & Saleh, A. (1989). A decentralized control architecture for computer integrated manufacturing systems, Proceedings of IEEE International Symposium on Intelligent Control, 44-49

Komoda, N.; Kera, K. & Kubo, K. (1984). An autonomous, decentralized control system for factory automation. IEEE Computer, Vol. 17, No. 12, 73-83

Lee, E. J., Togueni, A. & Dangoumau, N. (2006). A Petri net based decentralized synthesis approach for the control of flexible manufacturing systems, Proceedings of the IMACS Multiconference Computational Engineering in Systems Applications

Masuda, R. & Hasegawa, K. (1981). Mark flow graph and its application to complex sequential control system, Proceedings of the 13th Hawaii International Conference on System Science, 194-203

Merabet, A. (1986). Synchronization of operations in a flexible manufacturing cell: the Petri net approach. Journal of Manufacturing Systems, Vol. 5, No. 3, 161-169

Murata, T.; Komoda, N. & Matsumoto, K. (1986). A Petri net based controller for flexible and maintainable sequence control and its applications in factory automation. IEEE Transactions on Industrial Electronics, Vol. IE-33, No.1, 1-8

Queiroz, M. H. & Cury, J. E. R. (2000). Modular supervisory control of large scale discrete event systems, Proceedings of the Fifth Workshop on Discrete Event Systems

Silva, M. (1990). Petri nets and flexible manufacturing, In: Advances in Petri Nets 1989, Lecture Notes in Computer Science, Vol. 424, Rozenberg, G., (Ed.), 374-417, Springer-Verlag, Berlin

Petri Nets Applications

Edited by Pawel Pawlewski

ISBN 978-953-307-047-6
Hard cover, 752 pages
Publisher InTech
Published online 01, February, 2010
Published in print edition February, 2010

Petri Nets are graphical and mathematical tool used in many different science domains. Their characteristic features are the intuitive graphical modeling language and advanced formal analysis method. The concurrence of performed actions is the natural phenomenon due to which Petri Nets are perceived as mathematical tool for modeling concurrent systems. The nets whose model was extended with the time model can be applied in modeling real-time systems. Petri Nets were introduced in the doctoral dissertation by K.A. Petri, titled "„Kommunikation mit Automaten" and published in 1962 by University of Bonn. During more than 40 years of development of this theory, many different classes were formed and the scope of applications was extended. Depending on particular needs, the net definition was changed and adjusted to the considered problem. The unusual "flexibility" of this theory makes it possible to introduce all these modifications. Owing to varied currently known net classes, it is relatively easy to find a proper class for the specific application. The present monograph shows the whole spectrum of Petri Nets applications, from classic applications (to which the theory is specially dedicated) like computer science and control systems, through fault diagnosis, manufacturing, power systems, traffic systems, transport and down to Web applications. At the same time, the publication describes the diversity of investigations performed with use of Petri Nets in science centers all over the world.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821