The Fifteenth International Symposium on Artificial Life and Robotics 2010 (AROB 15th '10),
B-Con Plaza, Beppu,Oita, Japan, February 4-6, 2010

Specification and real-time control of robotic manufacturing systems based on concurrent process modeling

G. Yasuda

Nagasaki Institute of Applied Science, Nagasaki 851-0193, JAPAN
(Tel : 81-95-838-5184; Fax : 81-95-830-2091)
(YASUDA_Genichi@NiAS.ac.jp)

Abstract: The paper deals with a systematic method of specification and real-time control for robotic manufacturing systems based on concurrent process modeling. The large scale and complex manufacturing systems have a hierarchical structure where a system is composed several lines with some stations and each station also has several machines and so on. In such a hierarchical structure, the controllers are geographically distributed according to their physical structure. So it is desirable to realize the hierarchical and distributed control. In this paper, the task specification of discrete event manufacturing processes is represented using a global Petri net. Then it is decomposed and distributed into the machine controllers, which are coordinated through communication between the coordinator and machine controllers so that the decomposed transitions fire at the same time. Implementation of a real-time distributed control system is described for an example robotic manufacturing system. The demonstrations show that the proposed system can be used as an effective tool for consistent modeling and control of large and complex manufacturing systems.

Keywords: Robotic manufacturing systems, specification, real-time control, concurrent process modeling, Petri nets.

I. INTRODUCTION

Recently, based on the rapid development of the computer technology, the factory automation systems have been becoming larger and more complex. The system architecture has shifted to distributed and parallel processing from centralized processing in order to reduce the development cost and to improve the reliability. In the field of factory automation systems, a demand for the automatic control has diversified and the control logic has become extremely complicated. This is because the combinative complexity of the control requirement, which comes from the non-deterministic features of event driven systems such as manufacturing control systems, is inevitable. To deal with the complexity, a new methodology on control system design based on the concept of event driven system is necessary. However, appropriate representation methods and analysis methods for control mechanism have not sufficiently been established.

Programming paradigm modeled by a network, such as Petri net, has been considered to be useful, because the network model can describe the execution order of parallel/sequential processes directly without ambiguity. The Petri net is excellent in expression and analysis of the dynamic behavior of event driven systems, because it can model the system that consists of simultaneous process elements that interfere to each other. The programming technique makes it possible to realize systematic and high-level description of system specification. Therefore, it has been applied to a variety of system developments such as real-time systems, production systems, communication systems, and so on.

However, in case of factory automation systems, the network model becomes complicated and it lacks for the readability and comprehensibility. Besides, only specification analysis stage has been supported, and the support for the control software coding stage is insufficient [1]. Therefore, the flexibility and expandability are not satisfactory in order to deal with the specification change of the system.

Due to its complexity, a large and complex manufacturing system is commonly structured into a hierarchy. A coarsely grained hierarchy of abstraction levels is made up of the followings: planning, scheduling, coordination, and local control. Each level operates on a certain time horizon, on a certain view of the manufacturing system. At the upper level, the time horizon is long and the global system is considered. In the hierarchy, each lower level is a disaggregation of the upper one (factory, shop, cell, station, machine, etc) and on the other hand more details are taken into account (products, parts, operations, steps, etc). Progressively, real-time constraints are introduced and at the bottom of the hierarchy (local control) they are very hard and all the emergency procedures are implemented at this level.

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. Conventional Petri net based control systems were implemented based on an overall system model. 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.

The hierarchical and distributed control for large and complex discrete event 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. In this paper, the author presents a methodology for hierarchical and distributed control of large and complex robotic manufacturing systems using extended Petri nets, to construct the control system where the cooperation of each controller is implemented so that the behavior of the overall system is not deteriorated and the task specification is completely satisfied.

II. PETRI NET BASED SPECIFICATION OF MANUFACTURING SYSTEMS

Discrete event systems such as robotic manufacturing systems have the properties of asynchronism, ordering, concurrency and conflict, so that unsafeness and deadlock will be apt to occur in the systems. The Petri net is one of the effective means to represent such systems. For applying it to the design, analysis, and control of the systems the guarantee of safeness and the notation of input/output of signals from/to machines should be required. A kind of graph deduced from the Petri net was proposed so as to satisfy the above requirements [2].

The extended Petri net consists of the following six elements: place, transition, directed arc, token, gate arc, output signal arc. A place represents a condition of a system element or action. A transition represents an event of the system. A directed arc connects a place to a transition, and its direction shows the input and output relation between them. Places and transitions are alternately connected using directed arcs. The number of directed arcs connected with places or transitions is not restricted. A token is placed in a place to indicate that the condition corresponding to the place is holding.

A gate arc connects a transition with a signal source, and depending on the signal, it either permits or inhibits the occurrence of the event which corresponds to the connected transition. Gate arcs are classified as permissive or inhibitive, and internal or external. An output signal arc sends the signal from a place to an external machine. 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.

The proposed procedure of modeling and decomposition of robotic manufacturing systems are shown as follows. 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 as shown, where the activity of each equipment is also represented as a place.

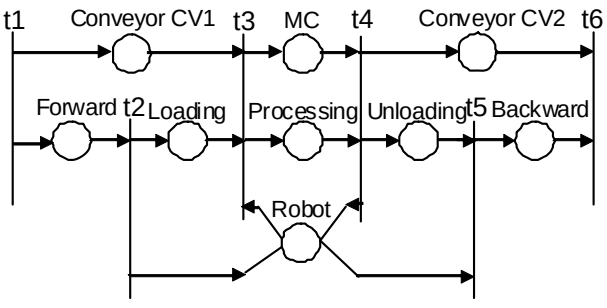

Fig.1 Petri net representation of example system at the conceptual level

A typical example system at the subsystem coordination level in the factory automation system concerns a robot loading and unloading a machining tool with input and output conveyors, and its Petri net representation at the conceptual level is shown in Fig.1.

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. At each step of detailed specification, some parts of the Petri net, transitions or places, are substituted by a subnet in a manner, which maintains the structural properties. The detailed Petri net representation of the loading operation in the example system is shown in Fig.2. Loading a workpiece to the machining center necessitates the cooperative or synchronized activities among the conveyor CV1, the machining center, and the robot. Similarly, unloading a workpiece from the machining center, necessitates the cooperative or synchronized activities among the conveyor CV2, the machining center, and the robot.

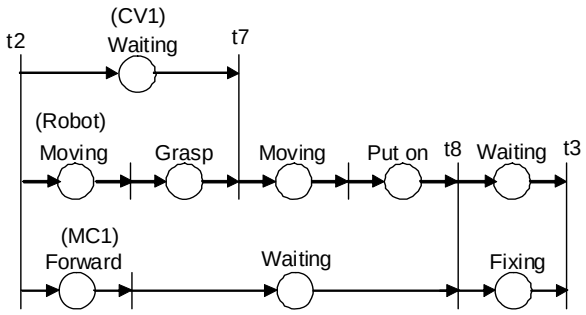

Fig.2 Detailed Petri net representation of loading operation

Unit operations in the detailed Petri net representation concern the real-time control of machines, devices, etc. They interact directly with the sensors and actuators. When a token enters a place that represents a unit operation, the hardware controller defined by the controller code is informed to execute a determined subtask with a determined data that are both defined by the place parameters. From these places, output signal arcs are connected to the machines, and external gate arcs from the machines are connected to the transitions of the Petri net when needed, for example, to synchronize and coordinate operations. The general form of Petri net representation at the local control level is shown in Fig.3.

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, several steps are necessary for each evolution of the process: first, the control system sends a request to the process actuators, second the external machine sends an acknowledge, third the process evolves according to the request and, at the end of the evolution, returns an execution report to the control system, and 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 token should not be moved until the operation, shown by each place, is finished, even if the marking satisfies the ignition condition.

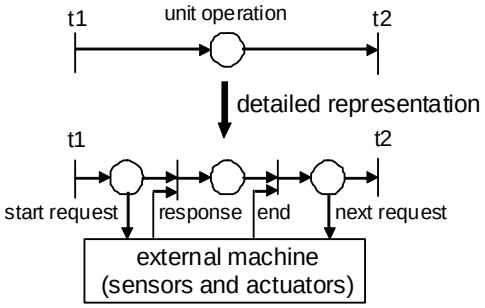

Fig.3 Detailed Petri net representation at the local control level

III. DECOMPOSITION AND COORDINATION

It is natural to implement a hierarchical and distributed control system, where one controller is allocated to each control layer or block. For the manufacturing system, an example structure of hierarchical and distributed control is composed of one station controller and three machine controllers. The detailed Petri net is decomposed into subnets, which are executed by each machine controller.

In the decomposition procedure, a transition may be divided and distributed into different machine controllers. Decomposed transitions are called global transitions (t2, t7, t8, t3 in Fig.2) 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. From the logical formulation of firability condition and marking before

and after decomposition, it is proved that the firability condition of the original transition is equal to AND operation of firability conditions of decomposed transitions [3]. In case that a transition in conflict with other transitions is decomposed, these transitions should be coordinated by the station controller.

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. The control of the overall system is achieved by coordinating these Petri net based controllers such that decomposed transitions fire at the same time and the task specification is completely satisfied. System coordination is performed through communication between the coordinator in the station controller and the Petri net based controllers in the machine controllers.

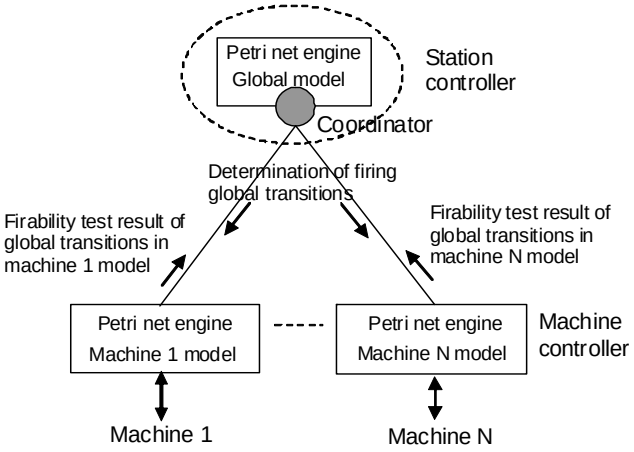

Fig.4 Petri net based control structure with coordinator

IV. PETRI NET MODEL BASED CONTROL EXPERIMENTS

From the fact that the control of a system represented by a Petri net can be realized directly by executing the corresponding Petri net as the control algorithm, a Petri net based controller was developed with a microprocessor. Also software was developed on a common PC to support synthesis, simulation, and debugging of Petri nets and to load the program into the controller. 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. Petri net simulation is performed by scanning these structural data in a tabular form. Using them, a control system was implemented for an experimental system, and their effectiveness was confirmed by executing tasks specified.

V. CONCLUSIONS

A methodology to construct hierarchical and distributed control systems, which correspond to the structure of manufacturing systems, has been presented. The controllers are arranged according to the hierarchical and distributed nature of the manufacturing system. The control software does not use the overall detailed system model, and the decomposed Petri net model in each machine controller is not so large and easily manageable.

Multilevel hierarchical and distributed control for large and complex manufacturing systems can be also 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.

The Petri net model includes the control algorithm; control is executed in order that the behavior of the Petri net model is in correspondence with that of the real system. Thus modeling, simulation and control of large and complex manufacturing systems can be performed consistently using Petri nets.

REFERENCES

[1] Desrochers, A. D. and Al-Jaar, R. Y., (1995) Applications of Petri Nets in Manufacturing Systems: Modeling, Control and Performance Analysis, IEEE Press

[2] Hasegawa, K., Takahashi, K., and Miyagi, P. E., (1988) Application of the Mark Flow Graph to represent discrete event production systems and system control, Trans. of SICE, 24, 69-75

[3] Yasuda, G., (2009) Implementation of distributed cooperative control for industrial robot systems using Petri nets, Preprints of the 9th IFAC Symposium on Robot Control (SYROCO '09), 433-438