Integrated Manufacturing System Modeling and Simulation Using Distributed Colored Timed Petri Net

Chung-Hsien Kuo * and Han-Pang Huang *

Robotics Laboratory, Department of Mechanical Engineering
National Taiwan University, Taipei, TAIWAN 10674, R.O.C.
TEL/FAX: (886)2-23633875
e-mail: hphuang@w3.me.ntu.edu.tw

*Professor and correspondence addressee *Graduated student

ABSTRACT

In this paper, an integrated and distributed modeling and simulation environment is developed for the next generation manufacturing systems. A distributed colored timed Petri net (DCTPN) is the modeling techniques for the proposed environment. Based on the DCTPN, the shop floor process and the manufacturing execution system (MES) can be integrated horizontally and vertically. In addition, a component-based three-tier application architecture is developed to achieve the distributed transparency, fault tolerance and loading balance. Finally, an order release server is integrated to the shop floor process DCTPN model to provide the wafer release orders based on different order releasing control policies.

INTRODUCTION

Next generation manufacturing systems are facing strict competition due to the customized production, shorten product life cycle and frequent process reengineering. Hence, the modeling and simulation are even more important then before next generation manufacturing environment. In order to fit the requirements of the next generation manufacturing environment, the distributed configuration, system integration and standard hardware/ software interfaces are proposed.

In this paper, a distributed component based colored timed Petri net is proposed to model the next generation manufacturing systems. A distributed colored timed Petri net (DCTPN) is developed based on the original Petri net (PN). The proposed DCTPN can model the colored-based manufacturing attributes and timed-based properties. In addition, the distributed and modularized models can be integrated into the desired configurations, and it is beneficial to the frequent process reengineering. Especially, a COM server place in a DCTPN is used to model the manufacturing execution system (MES). Based on the COM server places, the DCTPN models can integrate not only the shopfloor processes horizontally but also the MES vertically. In the proposed environment, common interface standards are defined among the models and MES servers. Finally, the distributed component interfaces are developed in terms of the Microsoft distributed component object model (DCOM). The proposed distributed architecture can achieve the distributed transparency, fault tolerance and loading balance for a manufacturing system. A distributed net constructor and simulation environment that are developed in terms of the Microsoft Visual C++ and Active template library (ATL) will be used to construct the next generation manufacturing systems. Finally, several IC fabrication applications are demonstrated to illustrate the capacity of the integrated DCTPN environment.

DISTRIBUTED COLORED TIMED PETRI NET

A distributed colored timed Petri net (DCTPN) is a kind of modified Petri nets, and it is developed based on the original Petri net (PN) [2, 12]. Petri net is practical to model the concurrent and asynchronous systems. They are important to the modern manufacturing systems. Therefore, a Petri net is suitable to model them. In fact, several literatures used the original PN or the modified PN to model the manufacturing systems. They are described as follows.

Kuo *et al.* [8] presented the modeling and simulation of an automotive assembly system using colored timed Petri net (CTPN). In this work, the product mixes behaviors can be well modeled. Kuo *et al.* [7] used an object-based CTPN to model an FMS. In this article, two manufacturing cells can be constructed modularly. In addition, the dispatching system is also considered. Lin *et al.* [11] used the colored timed Petri net (CTPN) to model a furnace in an IC Fab. In this work, the CTPN can also dynamically emulate a quasi environment for testing and predicting the problems in the future. Jeng *et al.* [6] presented the modeling, qualitative analysis and performance evaluation of the etching area in a Fab using Petri nets. In this research, the simulation can provide less than 5% error results as compared to the actual data. Srinivasan [14] proposed the modeling and performance analysis of a cluster tool using Petri net.

Zhou *et al.* [16] presented the construction of a flexible manufacturing cell using Petri net mathematical model. Jeng [5] proposed a Petri net synthesis theory for modeling flexible manufacturing systems. Abdallah *et al.* [1] proposed an efficient search algorithm for deadlock-free scheduling in an FMS using Petri net. Zhou *et al.* [17] used the Petri net to generate the optimal control policy

0-7803-5731-0/99/$10.00 ©1999 IEEE

for flexible manufacturing cells. Hu *et al.* [3] used the generalised stochastic Petri net (GSPN) to model and analyze an FMS for scheduling and decision support system (DSS). Such researches indicate that the Petri nets are suitable to model a manufacturing system. Hence, a PN is selected to model and simulate the proposed integrated environments.

An original Petri net is a four-tuple structure, $PN = (P, T, I, O)$, where P is a set of places, T is a set of transitions, I is an input function and O is an output function. The relationships between the places and the transitions are established by the input and output functions and connected graphically by the arcs. Based on this structure, the marking that indicates the number of token in all places can be used to represent the PN dynamics.

Since the complex characteristics of high product mix, complex product routines and distributed manufacturing of the next generation manufacturing environment, the original PN may be not suitable to model such a complex manufacturing system. The reasons are:

1. Resource attributes: The operation of the manufacturing system depends on resource attributes. For example, the investigation of a product-mix behavior cannot ignore the resource attributes.
2. Operation time: Time properties are important in the manufacturing systems such as the processing time of the equipment and the transportation time of an automatic material handling system (AMHS).
3. Manufacturing execution system (MES) integration capacity: Most of the related research of Petri net in manufacturing systems is focused on the process analysis, such as evaluating equipment utilization, work in process (WIP) and throughput. However, the shopfloor processes are strongly correlated with the manufacturing execution system (MES), such as the order release management server. Hence, the modeling of the MES in the manufacturing system can not be ignored.
4. Distributed modular design and communication issues: A complex system can be divided into several hierarchies and modules to simplify the system model. Each module can describe a system or a sub-system, while a system or a sub-system can contain several sub-systems. A distributed architecture can reduce the communication traffic and improve the communication efficiency when comparing with the centralized one. In addition, the distributed architecture is robust and fault-tolerant. Especially, the distributed manufacturing environment needs a distributed modeling technique.

DCTPN DEFINITIONS

Based on the previous reasons, a DCTPN is developed by adding the color attributes [4, 9], time property, modular description, communication connectivity and MES servers. A distributed hierarchical DCTPN can be defined as an eleven-tuple structure, $DCTPN = (P_t, T_t, P_0, T_0, P_c, P_d, T_m, A, B, F, C)$, where P_t is a set of timed places; T_t is a set of timed transitions; P_0 is a set of immediate places; T_0 is a set of immediate transitions; P_c is a set of communication places; P_d is a set of COM server places; T_m is a set of macro transitions; A is a set of directed arcs; B is a set of inhibitor arcs; F is a set of interrupt arcs; C is the color set of transitions and places. These elements are further elaborated as follows:

1. Place: P={p1,p2,p3,...,pn} is a finite set of places, $n \geq 1$, including immediate, timed, COM (component object model) server, domain-pitch, domain-catch, catch-up, catch-down, pitch-up and pitch-down places.
   (a) An immediate place can be used to describe the state, condition or property (without time factor) of resource.
   (b) A timed place can be used to model the resource that needs an elapsed time to be available. For example, a conveyer can be modeled by a timed place in a factory.
   (c) A COM server place can be used to model the MES server that is developed based on the DCTPN and DCOM interfaces. When a token enters a COM server place, the corresponding remote MES server can be executed to serve the corresponding COM server place.
   (d) Communication places provide the communication among different DCTPN nets. It includes the domain-pitch, domain-catch, catch-up, catch-down, pitch-up and pitch-down places. Domain-pitch and domain catch places are used to communicate between different DCTPN nets at the same hierarchy level, and they are established in terms of the pair relationship. A domain-pitch place can send the tokens to one and only one corresponding domain-catch place. A pitch-up place is used for the lower level DCTPN nets. It sends the tokens to one and only one corresponding catch-down place in the higher level net, and they are established in terms of the pair relationship. A pitch-down place is used for the higher level DCTPN nets. It sends the tokens to one and only one corresponding catch-up place in the lower level net, and again they are established in terms of the pair relationship.
   (e) Input (output) transitions are the transitions that are input (output) related to the place in terms of arcs.
2. Transition: T={t1,t2,t3,...,tm} is a finite set of transitions, $m \geq 1$, including immediate transitions, timed transitions and macro transitions.
   (a) An immediate transition can describe the event (without time factor) of resource or the decision-making.
   (b) A timed transition can describe the process that needs elapsed time. In the design of the DCTPN, the time transition removes the specified color set of tokens from the corresponding input places when fired. After appropriate time elapses, the tokens will be released to the corresponding output places. In addition, the timed transition can handle only one token at each firing time.
   (c) Input (output) places are the places that are input (output) related to the transition in terms of arcs.
   (d) A macro transition is the refinement of the DCTPN, and it can be expanded as a series of transitions, places and arcs. The interconnection between different nets is achieved by communication places. A macro transition can be established only when the input place of this macro transition is a pitch-down place, and the output place of this macro transition is a catch-down place. Essentially, the macro transition implements the modular design of the Petri nets [15].
3. Color: $C(p)$ and $C(t)$ represent the color sets of places in P and transitions in T;

$C(p_i) = \{a_{i1}, a_{i2}, \ldots, a_{iu_i}\}$, $u_i = |C(p_i)|$; $i = 1,2,\ldots,n$

$C(t_j) = \{b_{j1}, b_{j2}, \ldots, b_{jv_j}\}$, $v_j = |C(t_j)|$; $j = 1,2,\ldots,m$

where n and m are non-negative integer; a and b are colors of places and transitions, and $|\bullet|$ denotes the cardinality.

4. Input, output, inhibitor and interrupt functions:
   $I(p,t)(a,b) : C(p)\times C(t) \rightarrow N$, is an input function. It describes the mapping relation from the transition t with color b to the place p with color a, where N is a non-negative integer. Similarly,
   $O(p,t)(a,b)$: $C(p)\times C(t) \rightarrow N$, is an output function.
   $Inh(p,t)(a,b) : C(p)\times C(t) \rightarrow N$, is an inhibitor function.
   $Int(p,t)(a,b) : C(p)\times C(t) \rightarrow N$, is an interrupt function.
5. Arcs:
   The input and output functions are represented in terms of the arcs graphically. The arcs in DCTPN are composed of directed, interrupt and inhibitor arcs. An arc multiplier is added to the arc in terms of fixed or attribute based integer variable. An arc multiplier can simplify the multiple arcs between a place-transition pair to one arc with an arc multiplier. In an IC fabrication manufacturing system, the relationships between a pod and pieces of wafers in a pod can be modeled in such an attributed-based arc multiplier since the number of pieces of wafers in a pod may be not fixed. Contrarily, a fixed arc multiplier is not useful here.
   The directed arc connects between a place and a transition, or vice versa. The place connected with an inhibitor arc is called an inhibitor place. When an inhibitor place contains the same color tokens as the output transition with the number of arc multiplier, then the output transition is inhibited to fire. Similarly, the place connected with an interrupt arc is called an interrupt place. When an interrupt place contains the same color tokens as the output transition with the number of arc multiplier, then the firing in the output transition is interrupted and further inhibited. For the immediate transition case, the interrupt and the inhibitor arcs behave the same. For the timed transition case, if the timed transition is firing and the interrupt place has the same color tokens with the number of multiplier, then it interrupts the firing of this transition and returns the color tokens to the input place(s).
6. Marking: $\mu(p):C(p) \rightarrow N$; $\forall p \in P$, has n elements. $\mu(p_i)$ is an $(n \times 1)$ vector defined as:

$$\mu(p_i) = \sum_{h=1}^{u_i} n_{ih} a_{ih}$$

   where n_{ih} is the number of tokens of colors a_{ih} at this instant; μ_i is the total number of colors in place p_i . $\mu(p_i)(a_{ih})$ denotes the number of tokens of colors a_{ih} in place p_i . μ_0 is the initial marking.
7. Conditions of enabling and firing: The transition t_j is enabled with respect to the color b_{jk} if

$$\mu(p_i)(a_{ih}) \geq I(p_i,t_j)(a_{ih},b_{jk}),\ \forall p_i \in P, a_{ih} \in C(p_i)$$

$$\mu(p_r)(a_{rg}) = 0,\ \forall Inh(p_r,t_j)(a_{rg},b_{jk}),\ \forall p_r \in P, a_{rg} \in C(p_r)$$

$$\mu(p_l)(a_{lf}) = 0,\ \forall Int(p_l,t_j)(a_{lf},b_{jk}),\ \forall p_l \in P, a_{lf} \in C(p_l)$$

   After the transition t_j is fired, two outcomes could appear.
   For an immediate transition the new marking μ' becomes:

$$\mu'(p_i)(a_{ih}) = \mu(p_i)(a_{ih}) + O(p_i,t_j)(a_{ih},b_{jk}) - I(p_i,t_j)(a_{ih},b_{jk})$$

   For a timed transition, the temporary marking μ'' is:

$$\mu''(p_i)(a_{ih}) = \mu(p_i)(a_{ih}) - I(p_i,t_j)(a_{ih},b_{jk})$$

   After the delay in t_j , the new marking μ''' becomes:

$$\mu'''(p_i)(a_{ih}) = \mu''(p_i)(a_{ih}) + O(p_i,t_j)(a_{ih},b_{jk})$$

   In a DCTPN, when a token in an immediate place is available, the condition of the output transition determines the fate of the token. If the output transition is an immediate transition, then the token is released from the input place to the output place(s). On the other hand, if the output transition is a timed transition, then the token will not be released until a prescribed time delay is reached. Tokens in a timed place are processed in a similar way.
8. Time function: It is simply the time attribute in the places and transitions. For the transition, $f(t(b_{jk})) : T \rightarrow N$, is the time required by the timed transition t associated with the color b_{jk} to complete the firing. For the place, $f(p(a_{ih})) : P \rightarrow N$, is the stochastic time delay required in the timed place p associated with the color a_{ih} to release.
9. Reachability set: It consists of all reachable markings from the initial marking. Reachability is a basic property of the DCTPN. A marking μ_n is said to be reachable from the initial marking μ_0 if there exists a sequence of firings that transforms μ_0 to μ_n. The firing sequence is denoted by,
   $\sigma = t_1(b_{1k})\ t_2(b_{2k}) \ldots\ t_n(b_{nk})$, where b_{ik} is the color set of the transition b_n
10. Boundedness: A DCTPN is said to be k-bounded or simply bounded if the tokens in each place does not exceed a finite number k for any reachable marking μ_n from bounded initial marking μ_0. If k =1, then a DCTPN is said to be 1-safe or 1-bounded.
11. Liveness: A DCTPN is said to be live if no matter what marking has been reached from initial marking μ_0 , it is possible to fire any transition in the net by progressing through some future firing sequences. Consequently, a live DCTPN can guarantee the deadlock-free of the net.
12. Marking: In the DCTPN, colored tokens are the status of system resources. The token distribution in all places is the colored marking of the net. Colored markings can represent the dynamics of the DCTPN models. From the colored marking, the status of the system can be obtained at any instance.

DCTPN MODELING AND APPLICATION ARCHITECTURE

In this paper, the DCOM-based component server is used to develop the distributed modeling and simulation environment. DCOM [13] is developed by Microsoft, and it extends the component object model (COM) to support distributed communication among

components on different sites through a LAN, a WAN, or even the Internet. Hence, the component servers can be registered and allocated in different computers to achieve distributed computing. In addition, a component-based three-tier architecture is used here to achieve the distributed transparency, loading balance and performance balance, as shown in fig. 2. In this architecture, the data service tier is configured in terms of the database server, such as Microsoft SQL server; the second tier is constructed in terms of the DCOM-based component servers; the client application tier is consist of the client applications. In the proposed environment, the data server stores the data of the DCTPN models, initial marking and simulation results. The client application can be the DCTPN net constructor and simulation emulator. Especially, the client application uses the DCTPN inference engine to emulate the DCTPN simulation, and uses the order release server to generate the production order. Besides, the component servers can read/ write data from/ to the database in the data server tier. Finally, the client applications communicate with each other using the NTU-NET [10]. Notices that NTU-NET is a real-time communication protocol. It is developed based on TCP/IP. Specially, it divides the communication packages into five priorities; namely, alarm, condition, message, mail and file. In addition, the NTU-NET provides the security, fault tolerance and efficiency of communications.

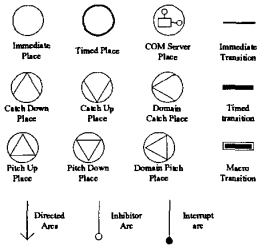


Fig. 1 DCTPN icon definition

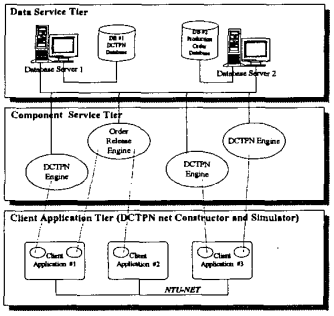


Fig. 2 Three-tier component-based architecture

Based on this three-tier component-based architecture, the distributed transparency, loading balance and fault tolerance can be achieved since the distributed DCOM-based component server can be replaced, removed and added by the client applications in terms of the DCOM IDL (interface description language). The methods to achieve distributed transparency, loading balance and fault tolerance are described as follows.

1. Initially, the client application can arbitrarily select any available component server to achieve the distributed transparency.
2. When a component server is failed, the client application can select other available component server of different site to continue the previous tasks. Hence, the distributed fault tolerance can be achieved.
3. When a component server is overloaded, the client application can release the original component server, and change the tasks to other available component server of different site. Consequently, the distributed loading balance can be achieved.

The applications in this paper are developed in terms of the Microsoft Visual C++, Active Template Library (ATL) and SQL server. They are constructed and developed by following the three-tier component-based architecture hinted in Fig. 2. These applications form an integrated DCTPN environment, and this environment provides distributed modeling and simulation of the DCTPN nets. The application architecture is shown in Fig. 3. The functions of each application are described as follows.

1. CTPNDesign: It is a Windows-based DCTPN model constructor and real-time simulator.
2. CTPNWeb: It is a browser-based DCTPN real-time simulator. Notice that the CTPNDesign and CTPNWeb can directly communicate with each other in terms of the NTU-NET.
3. OrderEng: It generates a new production release order in terms of the release control policies, and it is executed by the CTPNDesign and CTPNWeb.
4. CTPNEng: It is a DCOM server application. The DCTPN inference rules (firing rule) are coded in this application. In this application architecture, the CTPNEng is executed by the CTPNDesign and CTPNWeb for the simulation issues.
5. SQL server: It stores the data of DCTPN models, production route table, production order table, simulation parameters and results. This database support the data of CTPNEng, OrderEng, CTPNDesign and CTPNWeb.

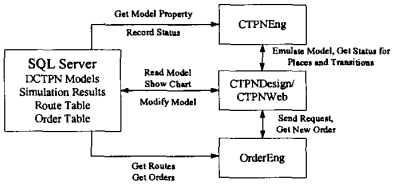


Fig. 3 Application architecture of DCTPN integrated environment

In this paper, the DCTPN models are constructed in terms of the user-friendly Windows-based CTPNDesign net constructor, as shown in Fig. 4. The CTPNDesign supports all DCTPN elements defined in this paper. Since the CTPNDesign is an object-oriented modeling application, the DCTPN elements are all objects. The user can select the desired DCTPN elements from the toolbar when the design mode is selected. Then the selected DCTPN elements, including places, transitions and arcs, can be directly put into the view. In this manner, a DCTPN model can be constructed in terms of the CTPNDesign. Then the parameters of each DCTPN

element (object) can be assigned in terms of dialogs. The parameters of the DCTPN elements are described as follows.

1. Model parameters: Color sets, DCTPN net ID, DCTPN net hierarchy level, DCTPN net communication port and DCTPN initial marking ID.
2. Places: Place type (immediate place, timed place, and so on), place name, place priority, place color sets, boundedness, conflict transitions resolution rule, remote communication host ID, remote communication port, remote communication priority, corresponding remote place ID, time parameter, COM-server type and MES server control policies.
3. Transitions: Transition type, transition name, transition priority, transition color sets, token releasing rule, queue length place ID, remote net ID, remote net hierarchy level, firing rule, token releasing rule and time parameter
4. Arcs: Arc type (directed, inhibitor and interrupt), arc multiplier type (fixed or reference color set) and arc multiplier value.

Due to the object-oriented programming, the relationship of the DCTPN objects can be established. Once a place and a transition are connected by an arc, the input/ output place (transition) of the transition (place) can be automatic established. On the other hand, the token release rule of the places and the conflict transitions resolution rules of the DCTPN are constructed in the proposed DCTPN environment. The conflict transitions resolution rules are the firing count, firing time, queue length, priority and referring name of conflict transitions. Besides, the token release rule of the places are first come first serve (FCFS), priority, last come first serve, commit due date, processing time, weighted processing time, remaining processing time, operation remaining count and release time of the color set (attributes) of tokens.

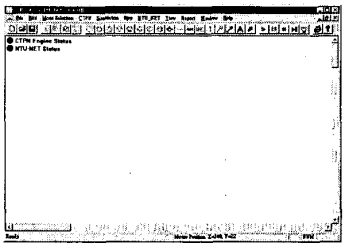


Fig. 4 Application architecture of DCTPN integrated environment

DCTPN MODELING AND SIMULATION

In this section, the modeling of DCTPN is introduced. First, the use of COM-server place is illustrated. A COM-server place can be used to model the MES servers. Fig. 5 shows a DCTPN net, and this DCTPN net also shows the integration of the MES server. In this net, the initial marking is a token in P_01. This will cause the new token created in P_04 periodically (depend on the elapsed time value of T_01). Such created new token will cause a new token in a COM-server place, P_02. The entry token of P_02 will execute the remote component server assigned in Fig. 6, and it will follow the control policies to operate the corresponding MES server. In Fig. 6, an "order release server" is assigned, and it follows the control policies of "Wafer_Quantity->Lot_Quantity->Commit_Due->Lot_Priority" (highest priority to lowest priority) in an IC Fab. The releasing lot order is shown in Fig. 7. In order to examine the control policies of the "order release server", the control policies is changed to the "Lot_Priority->Commit_Due->Wafer_Quantity->Lot_Quantity". The different releasing lot order is shown in Fig. 8. In Fig. 7 and Fig. 8, The symbols of "IntAttValue0", "IntAttValue1", "IntAttValue2", "IntAttValue3", "CharAttValue0", "CharAttValue1", "CharAttValue2" and "BoolAttValue0" represent the wafer quantity in a pod, lot priority, release time, lot commit due, lot ID, product ID, route ID and operation time (minutes), respectively. Apparently, the wafer release order is different from the different order release policies. Therefore, a MES server can be properly integrated to the shop floor process models.

In addition, the lot tracking can be done when simulation starts. Fig. 9 shows the status of P_05. By the way, the status of each transition and place can all be queried in this environment in the simulation time.

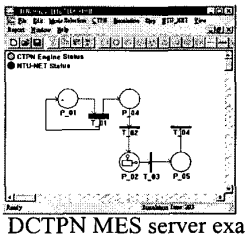


Fig. 5 DCTPN MES server example

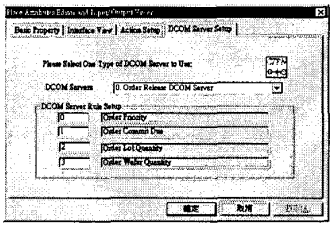


Fig. 6 COM-server place setup

Finally, the proposed environment can be simulated in a distributed configuration. In addition to the Windows-based application, the DCTPN can also be simulated in terms of the browser-based application. The browser-based application is convenient because it can be executed through the Internet explorer (IE) through the Internet. Fig. 10 shows two DCTPN nets, and these nets are executed through the browser. Notices that they can be executed on different computer and the pitch-down place P_02 at the left-hand-side net sends the token to the catch-up place P_01 at the right-hand-side net. Similarly, the pitch-up place P_03 at the right-hand-side net sends the token to the catch-down place P_03 at the left-hand-side net. The right-hand-side net is an expansion of the macro transition T_04 at the left-hand-side net.

CONCLUSION

This paper proposed a new extension of Petri nets. The proposed DCTPN emphasizes the integration capacity of the manufacturing systems. The designed

COM-server place follows the Microsoft DCOM IDL interfaces. Consequently, the MES servers can be developed based on the DCTPN and DCOM interfaces. They can be executed by the tokens of the COM-server places in the DCTPN nets directly. The benefits are that the DCTPN nets can handle more computation and inference tasks, and the models can include the MES properly. Hence, this work is more capable to the real-production system. Based on the integrated DCTPN environment, the DCTPN simulation can use the MES database through the COM-server place. Finally, the distributed modeling, distributed simulation, three-tier component-based application architecture and browser-based application make the integrated DCTPN environment more efficient, robust and convenient.

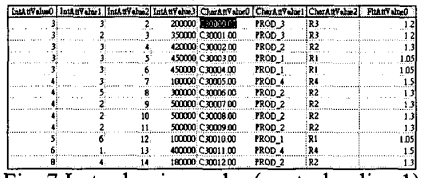


Fig. 7 Lot releasing order (control policy 1)

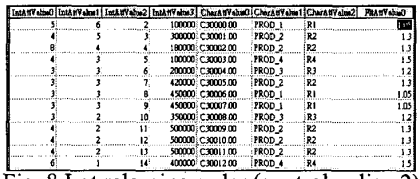


Fig. 8 Lot releasing order (control policy 2)

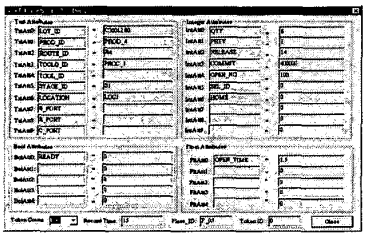
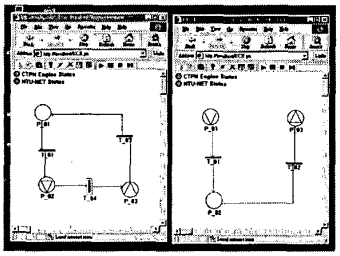


Fig. 9 Place status query in the simulation time



ACKNOWLEDGEMENT

This work was partially supported by the National Science Council, Taiwan, R.O.C., under Grants NSC87-2622-E-002-011 and NSC88-2218-E-002-003.

REFERENCES

[1] I.B. Adballah, H. ElMaraghy and T. ElMekkawy, "An Efficient Search Algorithm for Deadlock-Free Scheduling in FMS Using Petri Nets," *IEEE Intl. Conf. on Robotics and Automation*, Leuven Belgium, Vol. 2, pp. 1793-1798, May, 1998.

[2] A.A. Desrochers and R.Y. AI-Jaar, *Applications of Petri Net in Manufacturing Systems - Modeling, Control, and Performance Analysis* , N.Y.:IEEE Press, 1995.

[3] G.H. Hu, Y.S. Wong and H.T. Loh, "An FMS Scheduling and Control Decision Support System Based on Generalised Stochastic Petri Net," *Intl. Journal of Advanced Manufacturing Technology*, Vol. 10, pp. 52-58, 1995.

[4] P. Huber, K. Jensen and R.M. Shapiro, "Hierarchies in Coloured Petri Nets," *Advanced in Petri Nets 1990, Lecture Notes in Computer Science*, Springer-Verlag, pp. 313-341, 1990.

[5] M.D. Jeng, "A Petri Net Synthesis Theory for Modeling Flexible Manufacturing Systems, " *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 27, No. 2, pp. 139-183, 1997.

[6] M.D. Jeng, X. Xie and S.W. Chou, "Modeling, Qualitative Analysis, and Performance Evaluation of the Etching Area in a IC Wafer Fabrication System Using Petri Net," *IEEE Trans. on Semiconductor Manufacturing*, Vol. 11, No. 3, pp. 358-373, 1998.

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

[8] C.H. Kuo, H.P. Huang, K.C. Wei and S.S.H. Tang, "System Modeling and Real-Time Simulator for Highly Model-Mixed Assembly Systems," *Intl. Journal of ASME on Manufacturing Science and Engineering*, Vol. 121, pp. 1-8, 1999.

[9] C.H. Kuo and H.P. Huang, "Colored Timed Petri Net Based Statistical Process Control and Fault Diagnosis to Flexible Manufacturing Systems," *IEEE Intl. Conf. on Robotics and Automation*, New Mexico, Vol. 4, pp. 2741-2746, April, 1997.

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

[11] S.Y. Lin and H.P. Huang, "Modeling and Emulation of a Furance in IC Fab Based on Colored-Timed Petri Net," *IEEE Trans. on Semiconductor Manufacturing*, Vol. 11, No. 3, pp. 410-420, 1998.

[12] T. Murata, "Petri Nets: Properties, Analysis and Applications," *Proc. of the IEEE*, Vol. 77, No. 4, April, pp. 541-580, 1989.

[13] D. Rogerson, *Inside COM* , Microsoft Press, 1996.

[14] R.S. Srinivasan, "Modeling and Performance Analysis of Cluster Tools Using Petri Net," *IEEE Trans. on Semiconductor Manufacturing*, Vol. 11, No. 3, pp. 394-403, 1998.

[15] C.J. Tsai and L.C. Fu, "Modular Approach for Petri-Nets Modeling of Flexible Manufacturing Systems Adaptable to Various Task-Flow Requirement," *Proc. IEEE Int. Conf. on Robotics and Automation*, France, pp. 1043-1048, 1992.

[16] M.C. Zhou, K. McDermott, P.A. Patel and T. Tang, "Construction of Petri Net Based Mathematical Models of an FMS Cell, " *IEEE Intl. Conf. on Systems, Man, and Cybernetics*, pp. 367-372, 1991.

[17] Q. Zhou, M. Wang and S.P. Dutta, "Generation of Optimal Control Policy for Flexible Manufacturing Cells: A Petri Net Approach," *Intl. Journal of Advanced Manufacturing Technology*, Vol. 10, pp. 59-65, 1995.