Object-Oriented Approach of MCTPN for Modelling Flexible Manufacturing Systems

Chung-Hsien Kuo, Han-Pang Huang and Min-Chin Yeh
Robotics Laboratory, Department of Mechanical Engineering, National Taiwan University, Taipei, Taiwan 10674

Manufacturing industry is facing a stricter challenge than ever before owing to the rapid change in market requirements. Flexible manufacturing systems (FMSs) have a much greater capability than traditional fixed-type production systems for coping with the rapid change. In this paper, a modified coloured-timed Petri net (MCTPN) is developed to model the dynamic activities in an FMS. The MCTPN provides an object-oriented and modular method of modelling manufacturing activities. It includes colour, time, modular and communication attributes. The features of object-oriented modelling allow the FMS to be modelled with the properties of classes, objects, and container trees. Since the system activities can be encapsulated and modularised by the proposed MCTPN, the manufacturing systems can be easily constructed and investigated by the system developers. It makes the concept of software IC possible for modelling complex FMSs. Once all of the MCTPN objects are well defined, the developers need to consider only the interfaces and operations relating to the MCTPN objects. In order to demonstrate the capability of the proposed MCTPN, the FMS in the Manufacturing Automation Technology Research Center (MATRC) of the National Taiwan University will be stimulated and justified by using the proposed MCTPN along with the G2 expert system.

Keywords: Flexible manufacturing systems; Modelling; Modified coloured-timed Petri net

1. Introduction

An efficient flexible manufacturing system (FMS) that can cope with the requirements of rapid market change is highly desirable [1–6]. In order to achieve high productivity and quality, several modelling approaches have been used to analyse the efficiency of manufacturing systems, for example, queuing theory, finite-state machines, perturbation analysis, and Petri nets (PN) [7]. The activities in an FMS are embedded in multilevel structures [1,3,5,6,8]. More especially, they include production, process, cell and machine levels. In this paper, the modified coloured-timed Petri net (MCTPN) is developed for modelling complex FMSs. MCTPN is a modified PN that adds colour, time, modular and communication attributes to the traditional PN, and it is especially suitable for FMS modelling owing to its modular approach and object-oriented description.

Object-oriented modelling and design can describe problems in real-world concepts [9,10], and are now widely used in applications of software development. There are some successful applications in manufacturing system models. For example, Mize et al. [11] addressed three types of object in a manufacturing system, i.e. physical, informational, and control/decision objects. Chen [12] also proposed the methodology of integration of Petri nets and object-oriented technology. By using the object-oriented description and the modular approach, a complex FMS model can be constructed easily. The encapsulation, inheritance and polymorphism properties of objects are useful for constructing the system. The FMS model can be constructed along with the class, object and container trees, and thus the concept of software IC is made possible for FMS modelling. The developers need only consider the interfaces and operations relating to the defined MCTPN objects. In addition, the reusable property of the object-oriented technology will reduce the software development effort.

In the proposed MCTPN, macro transitions and communication places are used to construct a hierarchical and modular FMS model. The macro describes a series of process actions, such as operations in a flexible manufacturing cell (FMC). Independent modules are constructed first, then they are grouped into an FMS MCTPN module library. The communication signals integrate vertically from the machine level to the production level and horizontally among macro transitions on the same (machine, cell, process or production) levels. This approach makes the modelling of an FMS easier and clearer. Furthermore, it makes the model extension, maintenance and debugging more convenient.

Based on the proposed MCTPN, a real-time flexible manufacturing cell (FMC) controller and a real-time FMS simulator will be developed and applied to the FMS at MATRC (Manufacturing Automation Technology Research Center),
National Taiwan University. In this real-time FMS simulator, 13 dispatching rules are used which are based on the machine and workpiece properties.

2. Flexible Manufacturing System

The property of flexibility is important for flexible manufacturing systems, and an FMS combines both flow shop and job shop production methods. An FMS typically consists of a number of subproduction systems such as process actions, material handling devices, material storage, control units, inspection station and gauging stations. Production control messages including process information, product information and control commands are routed via a communication system, which consists of a computer, a control unit and a local area network (LAN).

FMSs combine the advantages of the high-flexibility job shop production and the high-efficiency flow shop production. Thus, FMSs can be designed to produce multiple types of components in small batches. The FMS under study will be decomposed into four levels hierarchically as shown in Fig. 1.

1. Production level. This is the highest level in an FMS. It handles the operations of the system, and it consists of a master production schedule (MPS), system resource management, database, scheduling of maintenance and system performance measurement.

2. Process level. This coordinates and controls all cells in the FMS, e.g. real-time scheduling (dispatching) rules.

3. Cell level. This is the level of the manufacturing cells below the process level.

4. Machine level. This may be machining, inspection or material handling equipment, e.g. conveyors, machining centre, lathe machine, milling machine, inspection centre, robot and automated guided vehicle (AGV).

3. Modified Coloured-Timed Petri Net (MCTPN)

The MCTPN extends the framework of the original PN by adding colour, time and modular attributes to the net. The

![Hierarchical structure of the FMS.](image_url)
modules, and use the catch-down place to receive (or catch) tokens from the lower-level modules. Similarly, the lower-level modules use the pitch-up place to send tokens to the higher-level modules, and use the catch-up place to catch tokens from the higher-level modules [2-6,14].

By using a hierarchical and modular design, the FMS can be modelled modularly. The primary considerations are to make the model easy to extend, maintain and debug in the future.

A hierarchical MCTPN is a ten-tuple, \( MCTPN = (P~, T~, P_0, T_0, P_c, T_c, A, B, F, C) \), where \( P_0 \) is a set of immediate places; \( T_0 \) is a set of immediate transitions; \( P_c \) is a set of communication places; \( T_c \) 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 colour set of transitions and places.

These elements are further elaborated as follows:

1. Place: \( P = \{ p_1, p_2, p_3, \ldots, p_n \} \) is a finite set of places, \( n \geq 1 \), including immediate places, timed places and communication places. Notice that the immediate places describe the conditions or properties (without time factor) of resources; the timed places describe the time properties of resources; the communication places describe the communication packages in the hierarchical MCTPN.

2. Transition: \( T = \{ t_1, t_2, t_3, \ldots, t_m \} \) is a finite set of transitions, \( m \geq 1 \), including immediate transitions, timed transitions and macro transitions. Notice that the immediate transitions describe the behaviours or events (without time factor) of resources; the timed transitions describe the time properties of resources.

3. Colour: \( C(p) \) and \( C(t) \) represent the colour sets of places in \( P \) and transitions in \( T \):

\[
C(p) = \{ a_{11}, a_{12}, \ldots, a_{1u} \}, \quad u_1 = |C(p)| \quad (i = 1, 2, \ldots, n)
\]
\[
C(t) = \{ b_{j1}, b_{j2}, \ldots, b_{jm} \}, \quad v_j = |C(t)| \quad (j = 1, 2, \ldots, m)
\]

where \( n \) and \( m \) are non-negative integer; \( a \) and \( b \) are colours of places and transitions, and \( | \cdot | \) denotes the cardinality.

4. Input, output, inhibitor and interrupt functions:

\[
I(p, t)(a, b) : C(p) \times C(t) \rightarrow \mathbb{N}, \text{ is an input function. It describes the mapping relation from the transition } t \text{ with colour } b \text{ to the place } p \text{ with colour } a, \text{ where } \mathbb{N} \text{ is an non-negative integer. Similarly,}
\]
\[
O(p, t)(a, b) : C(p) \times C(t) \rightarrow \mathbb{N}, \text{ is an output function}
\]
\[
Inh(p, t)(a, b) : C(p) \times C(t) \rightarrow \mathbb{N}, \text{ is an inhibitor function}
\]
\[
Int(p, t)(a, b) : C(p) \times C(t) \rightarrow \mathbb{N}, \text{ is an interrupt function}
\]

5. Marking: \( \mu(p) : C(p) \rightarrow \mathbb{N}; \forall p \in P \), has \( n \) elements. \( \mu(p) \) is an \( (n \times 1) \) vector defined as:

\[
\mu(p) = \sum_{i=1}^{u_1} n_i a_{ih},
\]

where \( n_{ih} \) is the number of colours \( a_{ih} \) at this instant; \( u_i \) is the total number of colours in place \( p_i \). The term \( \mu(p)(a_{ih}) \) denotes the number of colours \( a_{ih} \) in place \( p_i \). The term \( \mu_0 \) is the initial marking.

6. Conditions of enabling and firing: The transition \( t_j \) is enabled with respect to the colour \( b_{jk} \) if

\[
\mu(p)(a_{ih}) \geq I(p, t)(a_{ih}, b_{jk}), \quad \forall p_i \in P, \quad a_{ih} \in C(p_i)
\]
\[
\mu(p)(a_{ih}) = 0, \quad \forall \text{Inh}(p, t)(a_{ih}, b_{jk}), \quad \forall p_i \in P, \quad a_{ih} \in C(p_i)
\]
\[
\mu(p)(a_{ih}) = 0, \quad \forall \text{Int}(p, t)(a_{ih}, b_{jk}), \quad \forall p_i \in P, \quad a_{ih} \in C(p_i)
\]

After the transition \( t_j \) is fired, there are two possible outcomes.

For an immediate transition the new marking \( \mu' \) becomes:

\[
\mu'(p)(a_{ih}) = \mu(p)(a_{ih}) + O(p, t)(a_{ih}, b_{jk}) - I(p, t)(a_{ih}, b_{jk})
\]

For a timed transition, the temporary marking \( \mu'' \) is:

\[
\mu''(p)(a_{ih}) = \mu(p)(a_{ih}) - I(p, t)(a_{ih}, b_{jk})
\]

The transition waits for the firing time, then releases the token(s) to the corresponding place(s). Then the new marking \( \mu'' \) becomes:

\[
\mu''(p)(a_{ih}) = \mu'(p)(a_{ih}) + O(p, t)(a_{ih}, b_{jk})
\]

7. Time function: It is simply the time attribute in the places and transitions. For the transition, \( f(t(b_{jk})) : T \rightarrow \mathbb{N} \), is the time required by the timed transition \( t \) associated with the colour \( b_{jk} \) to complete the firing. For the place, \( f(p(a_{ih})) : P \rightarrow \mathbb{N} \), is the time delay required in the timed place \( p \) associated with the colour \( a_{ih} \) to release.

8. Reachability set: It consists of all reachable markings from the initial marking.

9. Liveness, deadlock: For any reachable marking, if there exists a transition in this MCTPN which is enabled, then the MCTPN is live; otherwise, this MCTPN is in a deadlock.

The icon definition of MCTPN is shown in Fig. 2.

### 4. Object-Oriented MCTPN Description

The object-oriented technique is becoming more popular for system modelling and analysis, especially for software application development. The object-oriented approach describes the system as a collection of objects, and these objects can interact with one another using both data structure and behaviour. Thus, object-oriented technology attempts to connect the data
and behaviour strongly. In the object-oriented area, the aspects are identity, classification, polymorphism, and inheritance. In the design of a system, the methodology binds the model in the application domain, then implements this system in detail. The methodology consists of analysis, system design, object design, and implementation stages [11].

In this study, the object-oriented MCTPN models are designed in terms of the stages of object-oriented methodology. Class, message and object are elements of object-oriented technology. In the object-oriented approach [16] object and message passing represent the resource object and the relationship between the resource objects. Class and inheritance can reduce the effort required for system development. The method represents the operation of the objects. Thus, MCTPN objects may be described by using the object-oriented description.

Aggregation, generalisation and inheritance play important roles in the design of an object-oriented MCTPN FMS model. Aggregation can be described as the relationship of “part-whole” or “a-part-of”. In other words, aggregation describes the relationship between an entire assembly object and its component objects. Generalisation and inheritance can be used to share the similarities among classes while preserving their differences. In other words, generalisation describes the relationship between a class and its refined version. The class being refined is the superclass, and each refined version is a subclass. Since the instance of a subclass is also an instance of the superclass, generalisation may be described as the relationship of “is-a” [10]. In the object-oriented MCTPN FMS models under study, the applications of aggregation and generalisation are described as follows:

1. The system view: In the FMS under study, FMC1 and FMC2 are two manufacturing cells. The manufacturing cell, storage cell, inspection cell and AGV are parts of the FMS. Thus, aggregation association can represent the system object architecture of FMSs (see Fig. 3).

2. The object-oriented MCTPN view: In the object-oriented MCTPN under study, the MCTPN objects are constructed by using the inheritance association. Within the object hierarchy, subclasses inherit or override attributes from its superclass, and the branches from a class are its subclasses. Thus the classes of places and transitions can be classified to be the superclasses in MCTPN nets (see Fig. 4).

An MCTPN model can represent the dynamic behaviour of the object-oriented model. The colour sets of an MCTPN

![Fig. 3. Aggregation and generalisation of an FMS.](image)

![Fig. 4. The MCTPN object hierarchy.](image)

definition indicate the attributes of places and transitions objects. For example, a timed place may have attributes such as boundedness, time, job, batch, buffer and machine attributes. The messages are transmitted by the arcs connecting the places and the transitions. The transitions are mapped into the operations in the MCTPN nets.

Modular design can simplify a complex FMS model. A macro transition can describe a series of actions for places and transitions. It is a modular representation. Container [16] is an important property in the object-oriented approach, and it allows objects to be embedded within it (for example, Microsoft Word). In the object-oriented MCTPN approach, the modular MCTPN can be described, that is a macro transition can be described as a container. For example, there are several place objects and transition objects in a container, and such a container is a modular design. The container tree of this FMS is described in Fig. 5.

When constructing an object-oriented MCTPN, the MCTPN objects can be described as:

1. Transition objects describe the event or decision.
2. Place objects describe the resource or condition.
3. Arcs describe the message passing; they are the operations between objects.

![Fig. 5. The FMS MCTPN container tree.](image)
4. Macro transitions may be described in terms of containers. An object-oriented MCTPN has the following properties:

1. MCTPN class hierarchy and inheritance.

2. Generalisation of MCTPN instance objects from MCTPN classes.

3. Data abstraction.


5. Polymorphism.

6. Dynamic binding.

Thus, the encapsulation, inheritance and polymorphism properties of object-oriented technology help designers to model a complex FMS conveniently. In addition, the reusable property of object-oriented technology reduces software development time. In this study, object-oriented technology and modular design can make system modelling and the simulator coding much easier and faster.

Since the properties of the object-oriented modelling (OOM) are used to model an FMS with the class, object and container trees, software IC will be possible when modelling the FMS.
Particularly, the developers need only to consider the interfaces and operations relating to the defined MCTPN objects.

In practice, the MCTPN classes must be constructed first. All objects of immediate places, timed places, communication places, immediate transitions, timed transitions and macro transitions are created from the instances of the corresponding MCTPN classes. In this manner, a complete MCTPN net can be transformed into an object-oriented program. Thus, an object-oriented description MCTPN model and an object-oriented programming can be realised.

5. Modelling an FMS by Using MCTPN

Before modelling the FMS, the designers have to construct the MCTPN classes, then group them into the class libraries. In this paper, the structure of the FMS is decomposed into four hierarchical levels. The MCTPN model is built using modular design. Each module of the MCTPN model is based on macro transitions. The material and information flows in the FMS are integrated by the communication places. The following concepts are used in the modelling of an FMS:

1. An immediate place describes the resources of the FMS, such as material, machine, message, control command and buffer.
2. A timed place is used to model a place that is associated with a time-delay, for example, a conveyor belt on which components stay for a certain time.
3. A communication place describes a place that provides communication between macro transitions, e.g. between a cell layer and a machine layer.
4. An immediate transition models the events occurring in the FMS, for example, a machine completes one job. On the other hand, it can also model decision making, such as selecting the next product type for a machine under a specific dispatching rule.
5. A timed transition can describe a process that needs time to complete.

6. A macro transition models the aggregation of many processes, such as the cells or machines.

The following procedures are used in the construction of the model:

1. Analyse the FMS, use the hierarchical construction, and define the cells and machines in each level.
2. Define the system requirements, resources, decision making and events, then describe them using appropriate places and transitions.
3. Use macro transitions to model the FMS, cells and machines. Modify them in different levels if necessary.
4. Define the processes in each level in terms of material flow or machines.
6. Define the interface and communication between different levels and macro transitions.
7. Use MCTPN to model the flowcharts in step 5.
8. Justify the model.

This approach will be implemented on an object-oriented real-time expert system, G2 [17]. The G2 environment provides an
6. Dispatching Rules

So far, the approaches described above can only construct the FMS MCTPN model. However, such an MCTPN model cannot run alone. In this paper, dispatching rules are also proposed. An efficient dispatching rule affects the system performance directly. Thus, this MCTPN model can be integrated with some efficient dispatching rules to operate this simulator. In such a way, a complete real-time FMS simulator is achievable. In this paper, 13 real-time dispatching rules are used. In the MCTPN model, the dispatching rules are added to transition decisions.

The dispatching rules are developed in terms of the workpiece characteristics and machine characteristics. Thirteen dispatching rules [3,6] are developed as follows:

1. Dispatching rules depending on workpiece characteristics:
   (a) “FCFS”: first come first serve.
   (b) “SPT”: shortest processing time.
   (c) “WSPT”: weighted shortest processing time.
   (d) “SRPT”: shortest remaining processing time.
   (e) “LRPT”: longest remaining processing time.
   (f) “FORT”: fewest operation remaining time.
   (g) “LORP”: largest operation remaining time.
   (h) “EDD”: earliest due date.

2. Dispatching rules depending on machine characteristics:
   (a) “SOTA”: shortest operation time with alternative considered.
   (b) “ESTA”: earliest starting time with alternative considered.
   (c) “LITA”: longest idle time with alternative considered.
   (d) “EFTA”: earliest finishing time with alternative considered.
   (e) “LITA” + “ESTA”: combination of “LITA” and “ESTA”.

Inference engine to reason about the objects in the knowledge base. The object-oriented description MCTPN model can be constructed easily in such an object-oriented real-time expert system.
7. Application Examples

The proposed MCTPN will be used to design an FMC controller and an FMS simulator. The real-time cell controller for the FMC, based on the MCTPN, is shown in Fig. 6. This FMC contains a cell controller, a D&M4 CNC milling, a pallet, a CNC lathe (a simulation stand), a storage unit, a conveyor, a mini robot and some workpieces. The cell controller can control this FMC by using some appropriate dispatching rules. The real-time controller is constructed using the MCTPN model. A graphical user interface (GUI) is also developed for monitoring and simulation purposes. The GUI and cell controller are implemented on a personal computer.

A real-time simulator for the FMS is developed to illustrate how to create and implement the MCTPN model. Thirteen dispatching rules are used to operate this FMS real-time simulator. This FMS plant is an experimental automated factory in MATRC, National Taiwan University (see Fig. 7). It includes a transportation cell, a storage cell, two FMCs and an inspection cell. In this simulator, the graphical user interface plays an important part. Figure 8 shows the simulator layout. The cells are described as follows:

1. FMC1: is the first manufacturing cell in this FMS, and consists of a CNC milling machine, a CNC lathe machine, a robot and two buffers. Figure 9 shows the FMC1 layout, and Fig. 10 shows the FMC1 layout of the real-time simulator.

2. FMC2: is the second manufacturing cell in the FMS, and consists of a machining centre and an automated pallet changer and two buffers. Figure 11 shows the FMC2, and Fig. 12 shows the FMC2 layout of the real-time simulator.

3. Transportation cell: includes an AGV and three parking stations. The AGV is shown in Fig. 13.

4. Storage cell: consists of an AS/RS that is a three-by-six unit warehouse, a cart and two buffers. Figure 13 shows the AS/RS layout, and Fig. 14 shows the AS/RS layout of the real-time simulator.

5. Inspection cell: includes a coordinate measuring machine (CMM) and two buffers. Figure 15 shows the inspection cell's layout.

Fig. 17. MCTPN for FMS.
A graphical human machine interface (GHMI) [2] is introduced to simulate this real-time FMS simulator. The operator can set up the following conditions:

1. Set up the number of workpieces in each batch, the maximum number of batches is eighteen.
2. Set up the routeing steps and processing times for every batch.
3. Set up the due date of every batch.
4. Set up the weighting of each workpiece.
5. Set up the performance (or reliability) of all machines and cells.
6. Set up the dispatching rule that can operate in this real-time FMS simulator.
7. Set up the run mode with the options of one shift only or 3 shifts (continuous production).

In addition to the GHMI, some functions are also provided in this real-time FMS simulator. These functions are the real-time monitoring of FMS activities, real-time monitoring of MCTPN activities and on-line performance queries. The managers and operators can examine this system for process monitoring, and the process engineers can evaluate this system for both process monitoring and MCTPN emulation.

The following steps are used to model an FMS by using MCTPN.

1. Define macro transitions in the system level: they are AGV-macro-transition, AS/RS-macro-transition, FMC-macro-transition and inspection-macro-transition.
2. Define the macro transition in the level of FMC-macro-transition: they are FMC1-macro-transition and FMC2-macro-transition.
3. Define the macro transition in the level of FMC1-macro-transition: they are CNC-macro-transition and robot-macro-transition.
4. Define the decision transition in the system level.
   (a) Materials input system.
   (b) Materials output system.
   (c) Materials begin processing from AS/RS.
   (d) Materials begin inspecting from AS/RS.
   (e) Materials begin processing from the other cell.
   (f) Materials begin inspecting from the other cell.
   (g) Material back to AS/RS.
5. Define the decision transition in the cell level of FMC1.
   (a) Robot uploads the job.
   (b) Robot transfers the job.
   (c) Robot downloads the job.
6. In the machine level, analyse the necessary activities of a machine, and list all the required commands. The developers can describe them using appropriate places and transitions, and model the processing time by using the timed transitions.
7. Finally, define the communication places in the system, cell and machine levels.

For the proposed FMS, 13 dispatching rules are provided. These dispatching rules are provided in terms of the job properties and machine properties.
The complete FMS model can be constructed by using MCTPN. In the system level, there is an MCTPN net for FMS (see Fig. 17). In this net, transitions are used to describe the events, decisions, and macros in the FMS layer. For example, the FMS_T1 decides the destination cell, destination buffer, and the extracted workpiece in the AS/RS; the FMS_T14 describes the event of workpiece return to the AS/RS; the FMS_T11 describes the behaviour of the inspection cell. Places are used to describe the commands, resource status, and communication packages. For example, the FMS_P2 describes the command of “input” or “output”; the FMS_P22 communicates with the inspection cell; the FMS_P7 describes the status of AGV.

In the cell level, there are MCTPN nets for FMC1, FMC2, AGV, AS/RS and inspection cells. Here, an example of an FMC1 MCTPN net is shown in Fig. 18. On the lefthand side of this net, there are two macro transitions (robot and CNC). Each macro transition describes the behaviour of the corresponding machine. The communication places can communicate with the higher and lower layers. For example, FMC_P1 can receive FMS messages; FMC_P6 can send messages to the CNC. On the righthand side of this net, there are several nets. These nets contain the communication between the higher and lower layers.

In the machine level, there are MCTPN nets for CNC machines and the robot. The CNC MCTPN net is shown in Fig. 19. In this net, the timed transition is used to describe the process of action. For example, CNC_T7 describes the process of machining; CNC_T14 describes the process of maintenance.

When the MCTPN nets for the FMS have been completely constructed, the designer can write a program to simulate the FMS by using these MCTPN nets and the constructed MCTPN class libraries. In addition, the modular design and object-oriented programming (OOP) can simplify the coding work. In this FMS under study, a real-time object-oriented expert system G2 is used to establish this approach and simulate this FMS. The simulation flowchart is shown in Fig. 20.

The functions of this real-time simulator are described as follows:

1. Man–machine interface: including workpiece data setting, the workpieces of each batch, the routing and processing...
Table 1. Batch data setting.

<table>
<thead>
<tr>
<th>Batch number</th>
<th>Process 1 (time)</th>
<th>Process 2 (time)</th>
<th>Process 3 (time)</th>
<th>Process 4 (time)</th>
<th>Process 5 (time)</th>
<th>Total job number in one batch</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>la(50)</td>
<td>lb(45)</td>
<td>3(20)</td>
<td>2(55)</td>
<td>3(20)</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2(70)</td>
<td>3(20)</td>
<td>la(45)</td>
<td>1b(50)</td>
<td>3(20)</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1b(40)</td>
<td>lb(35)</td>
<td>3(20)</td>
<td>2(80)</td>
<td>3(20)</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>2(65)</td>
<td>3(20)</td>
<td>1b(55)</td>
<td>1a(40)</td>
<td>3(20)</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>2(55)</td>
<td>3(20)</td>
<td>1b(55)</td>
<td>3(20)</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>3(20)</td>
<td>2(70)</td>
<td>3(20)</td>
<td>1a(45)</td>
<td>3(20)</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>3(20)</td>
<td>2(70)</td>
<td>lb(45)</td>
<td>1a(55)</td>
<td>3(20)</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>2(75)</td>
<td>3(20)</td>
<td>lb(35)</td>
<td>2(75)</td>
<td>3(20)</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>1a(45)</td>
<td>lb(45)</td>
<td>3(20)</td>
<td>2(75)</td>
<td>3(20)</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>1b(50)</td>
<td>3(20)</td>
<td>3(20)</td>
<td>1a(45)</td>
<td>3(20)</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>2(65)</td>
<td>3(20)</td>
<td>3(20)</td>
<td>1b(55)</td>
<td>3(20)</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>1b(45)</td>
<td>3(20)</td>
<td>3(20)</td>
<td>2(80)</td>
<td>3(20)</td>
<td>1</td>
</tr>
<tr>
<td>13</td>
<td>1a(45)</td>
<td>2(70)</td>
<td>3(20)</td>
<td>1b(45)</td>
<td>3(20)</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>1b(40)</td>
<td>3(20)</td>
<td>3(20)</td>
<td>1a(45)</td>
<td>3(20)</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>3(20)</td>
<td>3(20)</td>
<td>la(45)</td>
<td>3(20)</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>2(65)</td>
<td>3(20)</td>
<td>3(20)</td>
<td>2(75)</td>
<td>3(20)</td>
<td>1</td>
</tr>
<tr>
<td>17</td>
<td>2(70)</td>
<td>3(20)</td>
<td>lb(45)</td>
<td>3(20)</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>18</td>
<td>3(20)</td>
<td>2(70)</td>
<td>3(20)</td>
<td>la(45)</td>
<td>3(20)</td>
<td>1</td>
</tr>
</tbody>
</table>

where:
- Process index la: FMC1 lathe
- Process index lb: FMC1 milling
- Process index 2: FIVIC2 (MC)
- Process index 3: Inspection centre
- Time: simulation time unit (second)
- : None operation

Table 2. Simulation results under different dispatching rules.

<table>
<thead>
<tr>
<th>Rule</th>
<th>Set up time (s)</th>
<th>Complete time (s)</th>
<th>Processing time (s)</th>
<th>Bottle neck index</th>
<th>Job complete sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>634</td>
<td>6159</td>
<td>5525</td>
<td>none</td>
<td>1,2,7,6,5,3,4,9,10,11,12,18,8,13,15,14,17,16</td>
</tr>
<tr>
<td>2</td>
<td>49</td>
<td>5758</td>
<td>5709</td>
<td>1</td>
<td>2,1,6,14,3,4,9,15,13,12,17,18,11,10,7,5,8,16</td>
</tr>
<tr>
<td>3</td>
<td>139</td>
<td>5878</td>
<td>5739</td>
<td>1</td>
<td>2,6,5,1,15,4,10,12,18,8,7,6,3,14,13,17,9,11</td>
</tr>
<tr>
<td>4</td>
<td>109</td>
<td>5761</td>
<td>5652</td>
<td>1</td>
<td>2,1,7,5,15,6,17,4,14,18,10,13,9,11,3,12,8,16</td>
</tr>
<tr>
<td>5</td>
<td>34</td>
<td>6043</td>
<td>6009</td>
<td>3</td>
<td>2,1,12,5,10,14,13,9,14,16,7,16,8,3,15,7,18,11</td>
</tr>
<tr>
<td>6</td>
<td>34</td>
<td>5885</td>
<td>5851</td>
<td>1</td>
<td>1,2,7,5,6,15,4,10,9,12,18,8,3,11,13,14,16,17</td>
</tr>
<tr>
<td>7</td>
<td>52</td>
<td>5526</td>
<td>5474</td>
<td>none</td>
<td>1,2,4,3,9,10,12,5,11,18,13,14,17,6,7,16,15,8</td>
</tr>
<tr>
<td>8</td>
<td>148</td>
<td>5683</td>
<td>5535</td>
<td>none</td>
<td>2,1,15,18,11,4,10,17,8,12,7,5,6,13,16,14,9</td>
</tr>
<tr>
<td>9</td>
<td>157</td>
<td>5782</td>
<td>5625</td>
<td>none</td>
<td>1,2,7,6,5,3,4,9,10,11,12,18,8,3,15,14,17,16</td>
</tr>
<tr>
<td>10</td>
<td>61</td>
<td>5549</td>
<td>5488</td>
<td>none</td>
<td>2,1,5,6,4,3,10,9,11,12,18,8,13,15,14,17,16</td>
</tr>
<tr>
<td>11</td>
<td>68</td>
<td>5728</td>
<td>5660</td>
<td>1</td>
<td>2,1,6,5,4,9,11,13,10,14,15,17,12,18,7,3,8,16</td>
</tr>
<tr>
<td>12</td>
<td>111</td>
<td>5644</td>
<td>5533</td>
<td>none</td>
<td>2,1,6,5,3,4,9,10,11,12,18,8,13,14,16,15,17</td>
</tr>
<tr>
<td>13</td>
<td>226</td>
<td>5743</td>
<td>5517</td>
<td>none</td>
<td>2,1,6,5,3,4,9,11,13,10,14,15,12,17,7,18,8,16</td>
</tr>
</tbody>
</table>

Where rule 1 to rule 13 are the dispatching rules in this study.
C.-H. Kuo et al.

Fig. 21. Workpiece status.

Fig. 22. Gantt chart for the workpiece status (cell layer).

Fig. 23. Gantt chart for the workpiece status (machine layer in FMC1).

Fig. 24. Gantt chart for all workpieces.
in the FMS. The status of every workpiece can be seen from this chart. For example, one can see which cell or machine is processing a specified workpiece. The vertical axis indicates the corresponding workpieces, each shown with different colours. Notice that the symbols for the workpieces (FMS-JOB\textsubscript{i}, FMC-JOB\textsubscript{i}, job\textsubscript{i}, i = 1 to 18) in Figs 22, 23, and 24 are different, although they describe the same workpieces (workpiece\textsubscript{1} to workpiece\textsubscript{18}). The routing definitions of all the workpieces are listed in Table 1.

8. Conclusions

In this paper, an MCTPN is proposed for constructing the dynamic activities of FMSs. The proposed MCTPN can be used to design both the FMC controller and FMS simulator. The objective of this paper is to illustrate the procedures of modelling, simulation and real-time control of the FMC and FMS. In the proposed MCTPN, object-oriented description and modular design play important roles. Such an approach can reduce the development time of the system. Any complex system can be modelled using the MCTPN in this manner, and the models are clear and easy to extend, modify and debug. In addition to the MCTPN modelling of the production activities in the FMS, the failure of machine and maintenance of equipment are also considered. The proposed MCTPN can also be used to model statistical process control (SPC), machine fault, maintenance of equipment and fault diagnosis [4]. Reliability analysis of the FMS based on MCTPN is currently being developed.

Acknowledgments

This work was partially supported by the National Science Council, Taiwan, ROC, under Grants NSC85-2212-E-002-073, and NSC85-2622-E-002-018. The authors are grateful to the reviewers for their careful reading of the manuscript and their helpful suggestions.

References