Chapter 3
Modeling Discrete Event Systems Using Petri Nets

M. Narciso and M.A. Piera

3.1 Introduction

The main characteristic of the behavior of a Discrete Event System (DES) is that the System's state variables do not change value until an event happens, in other words, the event generates an instant change in the state of the system. Time advances from event to event and events happen chronologically but not necessarily at regular time intervals. The complexity in these types of systems lies in the fact that any decision can block, freeze, delay, enable/disable future events.

Consider, by way of example, the process of airplane arrivals and departures at an airport that only has one runway. It is easy to see how the runway will be blocked until an airplane has finished landing and how this can delay scheduled take-offs with airplanes being forced to wait at the head of runway. In this same example we observe that we cannot accurately predict how the state of the system will evolve (simply think of the evolution over time of the numbers of airplanes waiting at the head of the runway), as the time required for runway use depends on conditions that are hard to represent, such as how much experience the pilot has, wind conditions, the exact weight of the airplane at the instant of take off, among other things, and that can have an influence on deviations from the nominal expected time for the take-off operation. In addition to these uncertainties, which can be modeled as stochastic activities that influence the time an event may last (start or end of an activity), there are also physical and logical constraints that may delay the occurrence of an event, known as concurrence, synchronization and parallelism, for a variety of activities that have an effect on the system's evolution.

In this sense, the stochastic, dynamic and asynchronous nature of discrete event systems demands a modeling formalism that considers all the aforementioned characteristics that influence the system's evolution and allows us to represent both its structure and its behavior as a function of the possible settings. This should all favor the maintenance of the model, eliminating or adding new events in

I.F. De La Mota et al., *Robust Modelling and Simulation*,
DOI 10.1007/978-3-319-53321-6_3

accordance with any changes to the system, or to the operational context, that could ultimately have an influence on said model.

Therefore, one requirement of the DES modeling formalism is that it must allow us to represent all the events that influence the system and the causal relationships between them so as to represent its entire behavior.

3.2 Petri Nets

Petri nets (PN) are a modeling formalism that enables us to naturally represent a DES. Models of discrete event systems are essentially based on the concepts of events and activities. An event corresponds to a change in the value of the system's state variables and an activity encapsulates what happens between two events. Although a PN is not the only formalism that will enable us to represent events and activities, it facilitates our formal representation of parallelism and synchronization (Silva and Valette 1989; Zaremba and Prasad 1994; Zimmerman 1995; Zimmermann et al. 1996; Zhou and Venkatesh 1999; Petri Nets World 2015).

Other aspects that contribute to potentiating PNs in both the modeling and the quantitative analysis of DES, as well as for the verification, validation and analysis of the simulation results, are (Guasch et al. 2003):

- They allow us to study structural aspects of the system, such as jammed situations or the achievability of certain states.
- They make it possible to immediately determine all those events that might occur when the system is in a certain state and all the events that could be unleashed by the occurrence of a particular event.
- They allow us to formalize a system at different levels of abstraction, in accord with the modeling objectives.
- They enable the description of a complex system using the bottom-up methodology: development of the complete system model based on the PN (submodels) of the subsystems that have already been developed and verified.
- They constitute a graphic modeling formalism with very few syntactic rules.
- They enable us to find the possible paths to achieving a final state starting from an initial state, and to know the cost of each one of the paths.
- They make it possible to obtain the set of possible states that can be achieved starting from an initial state.

3.2.1 Description of Petri Nets

A PN is a particular case of the directed, weighted and bipartite graph that uses the following elements of representation:

- **Place nodes:** are graphically represented by circles or ellipses and can be used both to describe a system's queues (warehouses, *buffers*, *stocks*, etc.) and to describe conditions on the state in which the elements or resources that make up the system are to be found. Figure 3.1 gives the graphic representation of a PN with 5 place nodes, known as P1, P2, P3, P4 and P5.
- **Transition nodes:** are represented by rectangles and can be used for modeling the events that appear in the dynamics of a system. The PN of Fig. 3.1 contains a single transition node known as T1.
- **Directed arcs:** are represented by arrows and make it possible to connect a place node with a transition node, or a transition node with a place node, but never two nodes of the same type. The PN in Fig. 3.1 contains five directed arcs, three of which connect place nodes P1, P2 and P3 to transition T1, and two of which connect transition node T1 to place nodes P4 and P5.
- **Weights**[1]: the arcs that connect the place nodes with the transition nodes usually have an associated weight, which allows us to describe, for example, the conditions required for the event represented by the transition node to be able to occur. Similarly, the arcs that connect the transition nodes with the place nodes usually have an associated weight, which makes it possible to describe the changes to the state of the system as a consequence of the occurrence of the event represented by the transition node. The arcs represented in the PN of Fig. 3.1 have weights of value 1, 2 and 3.
- **Tokens**[2]: are graphically represented as points inside a place node and allow us to model, for example, the number of elements (pieces, resources, personal banking, etc.) in a place node, or the state of a condition (true or false) that indicates the fulfillment or not of said state of a condition. In the PN in Fig. 3.1, place node P1 has 4 tokens, place nodes P2 and P4 have 1 token each, place node P3 has 3 tokens and place node P5 has 2 tokens.
- **Marking:** a marking represents any arbitrary distribution of tokens in the place nodes. The initial distribution of tokens in the place nodes is called initial marking. In the specific case of the PN in Fig. 3.1, the marking can be represented as a vector with the following values [4, 1, 3, 1, 2].

In a PN, we say that a node X is an input node from another node Y, if and only if there is an arc directed from X to Y. Similarly, a node X is said to be an output node from another node Y, if and only if there is an arc directed from Y to X. Thus, we can be speak of input place nodes, output place nodes, input transitions, output transitions, input arcs and output arcs.

Considering again the PN in Fig. 3.1:

- Place nodes P1, P2 and P3 are input nodes for transition T1.
- Place nodes P4 and P5 are output nodes of transition T1.

[1]When the arc weight is missing, by default it's value is considered as 1.

[2]Also called *tokens* in the original nomenclature.

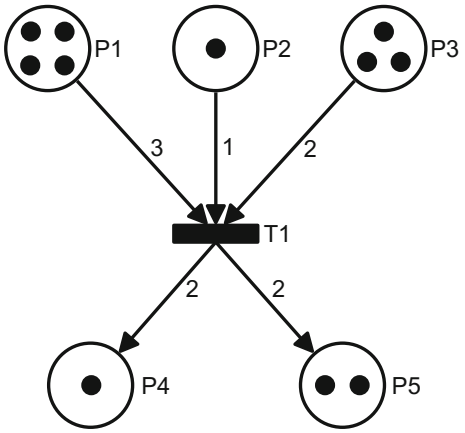

Fig. 3.1 Graphic representation of a PN

- The arcs that go from place nodes P1, P2 and P3 to transition T1 are input arcs to the transition and have a weight of 3, 1 and 2 respectively.
- The arcs that go from transition T1 to place nodes P4 and P5 are output arcs of the transition and have a weight of 2 each.

Although place nodes and transition nodes can be interpreted in different ways (Wang 1998), in this book the events shall be represented by transitions, the place nodes shall enable us to represent the state of the system, and the activities[3] shall be represented by place nodes encapsulated between 2 transitions.

3.2.2 *Formal Definition of Petri Nets*

In mathematical terms, a PN is defined as a tuple composed of five elements (Proth and Xie 1996; Wang 1998; Guasch et al. 2003):

$$\mathrm{RdP} = (\mathrm{P}, \mathrm{T}, \mathrm{A}, \mathrm{W}, \mathrm{M0})$$

where:

P = {P1, P2, P3, …, Pnp}: Non-empty finite set with place nodes (*np* is the number of place nodes in the PN).

T = {T1, T2, T3, …, Tnt}: Non-empty finite set with transition nodes, (*nt* is the number of transition nodes in the PN).

[3]According to the abstraction made of the system to be modeled, an activity can be represented in a PN as a place node or a transition node.

A = {a1, a2, a3, …, ana}: Non-empty finite set with arcs, (*na* is the number of arcs in the PN).
A is a subset of the Cartesian product of sets *P* and *T*:

$$A \subset (P \times T) \cup (T \times P)$$

where the first element of the ordered pair corresponds to the origin node and the second element to the destination node. The two nodes have to be different types, accordingly if a node is a transition type the other must be a place type and vice versa.

W: Ai → {1, 2, 3, …}: Weight associated with each arc Ai ∀ i = 1, 2, …, na.

M0 = [p1, p2, p3, …, pnp]: Initial marking, where pj is the number of tokens in the *j*th place node Pj: M(Pj) = pj ∀ j = 1, 2, …, np. The state of a system, after the occurrence of an event, is totally determined by the number of tokens in each place node, and can be mathematically described by the vector:

$$Ms = [p1, p2, p3, \ldots, pnp] \quad \forall\, s = 1, 2, \ldots, nm$$

where *nm* represents the total number of markings (states) of the system.
Thus, in the example of Fig. 3.1 the initial state can be represented as follows:

$$M0 = [4, 1, 3, 1, 2]$$

3.2.3 *Behavior or Dynamics of Petri Nets*

A PN can be treated as a games board where the tokens represent counters (which can only be put on the place nodes) (Jensen 1997). Each transition represents a potential movement in the game. A movement is possible if and only if each input place node of the transition contains at least the number of tokens prescribed by the weight of the corresponding input arc.

The rules for simulating the behavior or dynamics of a PN are (Proth and Xie 1996; Guasch et al., 2003):

- A transition It is enabled if each one of the place nodes Pj connected to the input contains at least W(Pj, It) tokens. Where each W(Pj, It) represents the weight of

the arc that joins node Pj to transition It. If its su does not appear on an arc, it is taken as being 1.

- An enabled transition can be fired at any instant in time.
- As a result of firing an enabled transition, W(Pj, It) tokens are eliminated from each node Pj in the It input, and W(It, Pk) tokens are added to each Pk node from the It output. Where W(It, Pk) corresponds to the weight of the arc that joins the It transition to the Pk node.

It is then said that a transition is *disabled* when there are less tokens in any of the input place nodes of the transition, for which the respective weights of input arcs to said transition prescribe, otherwise it is said that is *enabled.*

When a transition is enabled, the corresponding movement can take place. If this happens, we say that the transition has been fired. The effect of the occurrence of a transition is that the tokens are eliminated from the input place nodes and are added to the output place nodes. The number of tokens eliminate is specified by the weight of the corresponding input/output arc. It is important to point out that there is no relationship between the tokens eliminated from the input place nodes and those added to the output place nodes. The total number of tokens eliminated from the various input place nodes could be different from the number of tokens added to the various place output nodes.

When there are one or more enabled transitions for a given marking Ms, it means that every one of these transitions can be fired. Moreover, if there are enough tokens in the input place nodes for each one of them, so that each one of the transitions can obtain its own tokens without having to share them with the other transitions, it is said that the transitions are *concurrently enabled* in marking Ms. This means that the transitions can be fired "at the same time" or "in parallel".

It is worth mentioning that two transitions are only concurrently enabled if they are independent, in the sense that they can operate on disjoint sets of tokens. In this case the effect, when all the concurrently enabled transitions are fired, is that the resulting immediate actions are simultaneous and this effect is symbolically obtained as the "sum" of the effects of the individual transitions. The order in which each one of the individual transitions occurs does not affect the overall state of the system.

Using the PN from Fig. 3.1 as an example, we can illustrate the new state achieved by firing transition T1, according to the rules of behavior (Fig. 3.2):

The left side of Fig. 3.2 shows an enabled transition:

- The input nodes of the transition contain at least as many tokens as the weight of the arcs that connect them to the transition: P1 contains more than 3 tokens (W (P1, T1) = 3), P2 contains at least 1 token (W(P2, T1) = 1), and P3 contains more than 2 tokens (W(P3, T1) = 2).

The right side of Fig. 3.2 represents the state of the same PN once the transition is fired:

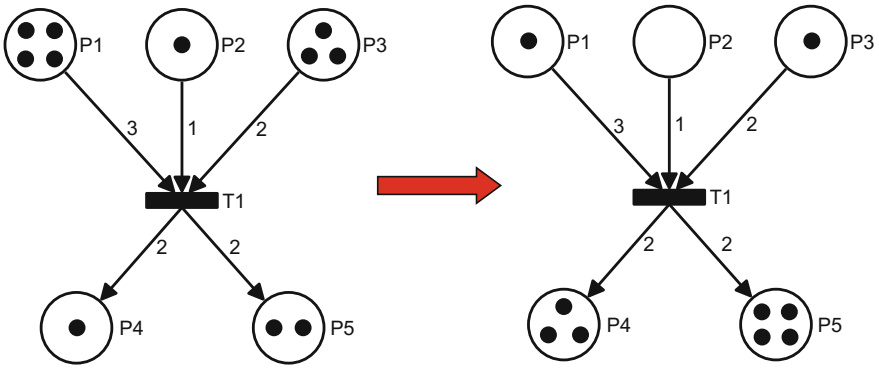

Fig. 3.2 Behavior or dynamics of a PN

- 3 tokens have been eliminated from place node P1 (W(P1, T1) = 3), one token from node P2 (W(P2, T1) = 1), and 2 tokens from place node P3 (W(P3, T1) = 2) 2 tokens have been added to place nodes P4 and P5 (W(T1, P4) = 2 and W(T1, P5) = 2).

The mathematical formalization of the PN allows us to analyze the dynamics of the modeled system based on the observation of possible events that could happen as a result of present state of the system and events that could happen because of the occurrence of a particular event.

Thus, in the example given in Fig. 3.2 we can observe the following states:

$$\mathrm{M0} = [4, 1, 3, 1, 2]$$
$$\mathrm{M1} = [1, 0, 1, 3, 4]$$

where M0 represents the initial state of the system and M1 the state of the system after the firing of transition T1.

3.3 Development of Petri Net Models of Systems

In this section we shall illustrate the concepts of ease of maintenance of models, as well as the relations of concurrency, synchronization and parallelism between the activities that are to be modeled. To this end, we propose the development of a set of submodels for the different functional specifications of a production system, which shall later be integrated to represent the entire flexible production system in PN.

There are many approximations for the modeling of systems using PN but in this section we are proposing the following steps:

1. Identifying all the events that could generate a change in the state of the system that is of interest for the simulation study.
2. Specifying the necessary logical *preconditions* for each event to be able to occur. Each logical condition will be formalized by means of a place node connected to the input of the transition, and with a certain weight on the arc to indicate the number of (physical or logical) resources required.
3. Specifying for each event the changes that shall occur to the state variables of the system because of the occurrence of the event. Said changes are known as *post-conditions* and are formalized by means of a set of place nodes connected to the transition output, with a certain weight on the corresponding arc to indicate the number of resources (both physical or logical) affected by said event.

Example 3.1 PN Model of a production unit

Let us consider a drilling machine with a set of pieces to be processed that are stored in stock S1, and a stock S2 with the already-drilled pieces. The machine has automatisms to be supplied with a piece from stock S1, drill it and as soon as said procedure is over, the piece is also deposited by means of automatisms in stock S2 with infinite capacity.

Under these operating conditions, machine M1 has 2 possible states: *free* and *working* and, in consequence, there are just only 2 possible events that make it possible to represent these changes of state:

1. *Start drilling procedure* event (T1): There are 2 logical preconditions for machine M1 to be able to start a drilling procedure:
   i. The machine must be *free*. It cannot drill 2 pieces at the same time, so it is necessary for the machine to be empty without any piece inside. In Fig. 3.8, this precondition is represented by an input arc of unit weight to transition T1 from place node P2 (machine M1 in *free* state).
   ii. Stock S1 must contain at least one piece awaiting drilling. There is no sense in starting a drilling procedure if there are no available pieces. In the Fig. 3.8, this precondition is represented by an input arc of unit weight to transition T1 from place node P1 (stock S1 with pieces to be processed).

As a consequence of the occurrence of a *Start drilling procedure* event, the state of machine M1 shall go to *working*, and one piece is eliminated from stock S1 that shall go on to be inside the automatisms of the machine. In Fig. 3.5, the output arc of transition T1 to place node P3 allows us to represent the change of state to *working*.

In Fig. 3.3 we can see the changes to be expected as a result of the appearance of a *Start drilling procedure* event. At the top the system is showed before the occurrence of the event (left side). Here we can observe 3 pieces that are awaiting drilling in incoming stock S1 and machine M1 *free*, and the right side illustrates the same system after the occurrence of the event, showing 2 pieces in stock S1 and one piece inside machine M1, being drilled.

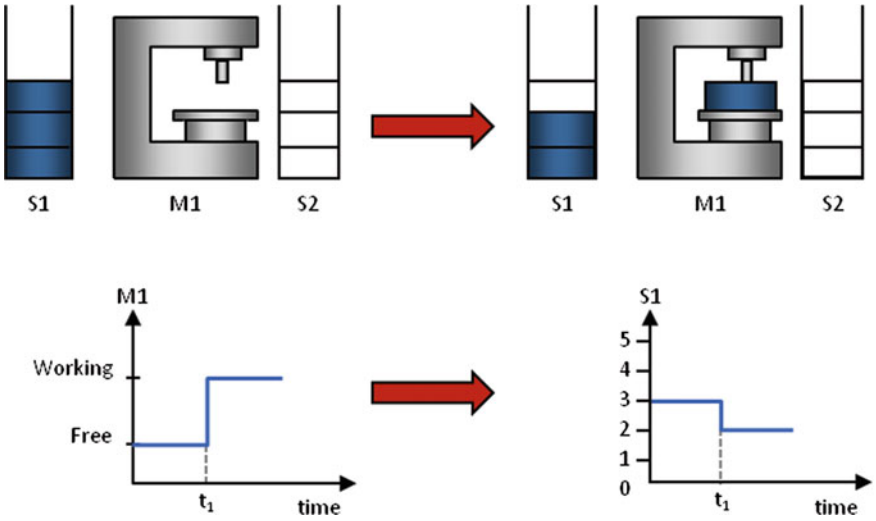

Fig. 3.3 Change of state because of the occurrence of the *start drilling procedure* event (T1)

At the bottom of the same figure, the changes to the state variables when the event happens in a particular instant t_1 have been represented. As can be seen, machine M1 will go from the *free* state to the *working* state, and stock S1 shall see a drop in the number of pieces.

2. *End drilling procedure* Event (T2): There is only one logical precondition for the event to be able to happen, and that is for machine M1 to be in a *working* state. It is not possible for an *End of drilling procedure* event to happen if the machine is in a *free* state. In Fig. 3.5, this precondition is represented by an input arc with a unit weight to transition T2 from place node P3 (machine M1 in *working* state).
3. As a consequence of the occurrence of an *End drilling procedure* event, the state of machine M1 shall go to *free*, and the number of drilled pieces in stock S2 shall be increased by one. In Fig. 3.5, the output arc of transition T2 to place node P2 allows us to represent the change of state to *free*, and the output arc from T2 to place node P4 enables us to represent a new already-drilled piece in stock S2.

Figure 3.4 shows the changes to be expected by the appearance of an *End drilling procedure* event. The top represents the system before the occurrence of the event (left side) where we can observe 2 pieces awaiting drilling in incoming stock S1, machine M1 drilling one piece and the stock of drilled pieces empty. The right side represents the same system after the occurrence of the event. Here we observe 2 pieces in stock S1, machine M1 empty and an already-drilled piece in stock S2. The bottom of the same figure represents the changes to the state variables when the event happens in a particular instant t_2. As can be seen, machine M1 will go from

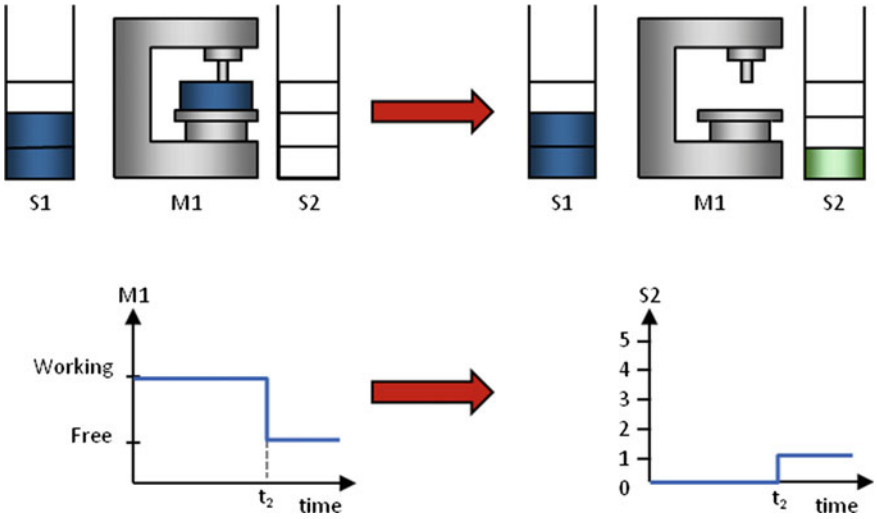

Fig. 3.4 Change of state because of the occurrence of the *end drilling procedure* event (T2)

working state to *free* state, and stock S2 shall see the number of drilled pieces increase by 1.

Figure 3.5 represents the PN for the drilling machine that has been described, while Table 3.1 describes the significance of the place nodes and Table 3.2, the significance of the transition nodes used.

Figure 3.5a shows the PN for the drilling machine considering as an initial condition that there are 3 pieces in stock S1 and that machine M1 is in a *free* state. Under these operating conditions, only transition T1 is found to be activated, as the number of tokens in P1 is higher than the weight of the arc that connects place node P1 to transition node T1 and the number of tokens in P2 is equal to the weight of the arc that connects place node P2 to transition node T1 (M(P1) >= W(P1, T1) and M

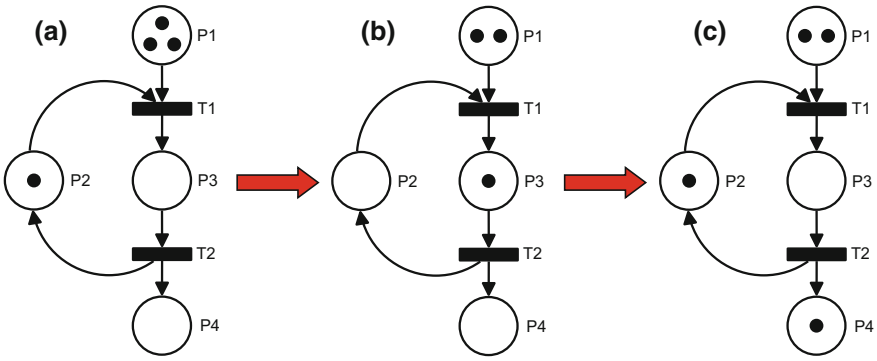

Fig. 3.5 PN model of a production machine

Table 3.1 PN place nodes for drilling machine

Place	Description
P1	Stock S1 of pieces awaiting drilling
P2	Machine M1 in *free* state
P3	Machine M1 in *working* state
P4	Stock S2 of already-drilled pieces

Table 3.2 PN transition nodes for drilling machine

Transition	Description
T1	Start drilling procedure
T2	End drilling procedure

(P2) >= W(P2, T1)). Whereas transition T2 is not activated because in place node P3 there is not at least one token.

In Fig. 3.5b we observe the same PN Model, but after transition T1 has been triggered. We can see that, as a consequence of the transition being triggered, one token has been eliminated in each one of the place nodes connected to the input of transition T1 (in other words, P1 and P2), and one token has been added to the place node connected to the output of the transition (i.e., P3). In (c) we observe the same network model as in (b), but after transition T2 is triggered. We can see that, as a consequence of the transition being triggered, one token has been eliminated from the place node connected to the input of transition T2 (in other words, P3) and one token has been added to each one of the place nodes connected to the output of the transition (i.e., P2 and P4).

Example 3.2 CPN Model for different sequences of production operations on different types of pieces

Consider in this example that 3 different types of pieces have to be processed, each of which requires a sequence of production operations specified in the recipe described in Table 3.3.

The manufacturing system consists of the production operations having to be done on different machines, each one with an incoming and outgoing stock that permits the production activities to be independent between the machines. The stocks leaving the machines correspond to the incoming stock for the machine that must perform the following production operation.

Figure 3.6 shows the elements of the production system where there is a transport subsystem (handling device) that makes it possible to move a processed piece in a machine to the stock for any other machine, or to the output stock S7, if the piece has already gone through all the production operations. Thus, in the case

Table 3.3 Sequence of procedures for type Π_1, Π_2, Π_3 pieces

Type of pieces	Sequence of procedures
Π_1	Adjustment—Pressing—Drilling—Polishing
Π_2	Molding—Milling—Drilling
Π_3	Adjustment—Molding—Milling—Pressing—Polishing

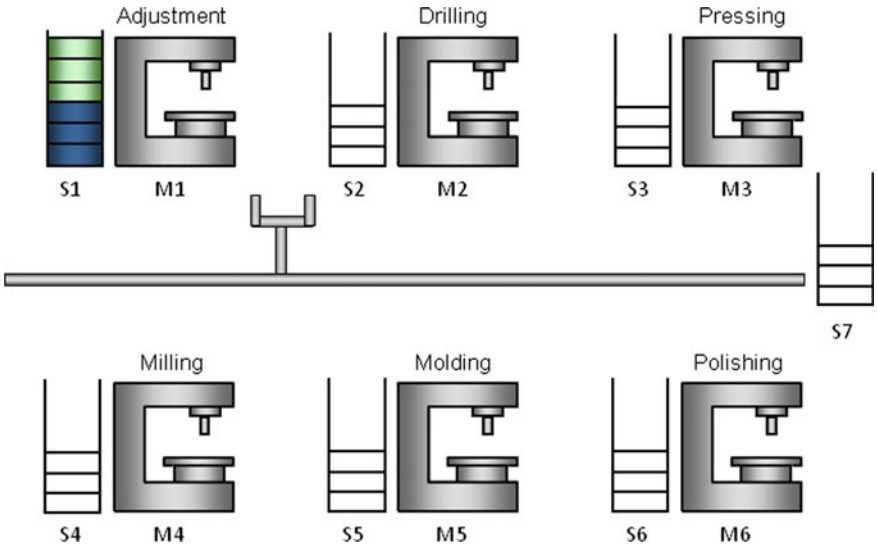

Fig. 3.6 Production system

of type Π_1 pieces, after the adjustment procedure has been performed, the handling device shall transport the piece to stock S3, whereas with type Π_3 pieces, after the adjustment procedure has been done, the handling device shall transport the piece to stock S5. The initial state of the system consists of 3 type Π_1 pieces and 3 type Π_3 pieces in stock S1, that are waiting for the adjustment procedure to be performed on them and 3 type Π_2 pieces that are waiting in stock S5 for a molding procedure to be performed on them.

In Fig. 3.7 we observe the PN that formalizes the 3 sequences of procedures that must be performed for the processing of the pieces stored in stock S1 and S5. Note that stock S1 cannot be represented in PN by a single place node but rather by 2 place nodes (in other words, P1 and P21) in order to be able to differentiate type Π_1 and type Π_3 pieces. Thus, although there is physically a single stock S1, at a logical level 2 stocks are considered, one for type Π_1 pieces (in other words, place node P1) and another stock for type Π_3 pieces (in other words, place node P21).

In the Table 3.4 the meaning of the place nodes is described and in Table 3.5 the meaning of the transition nodes that were used to model the manufacturing system is described.

As can be seen, the states machine *free* (in other words, P2, P5, P8, P11, P25, P28) are common to the different production sequences, as they do not have an assigned type of piece. Each one of these nodes participates in a pattern of behavior of the Decision/Conflict type, where there are two or more transitions that compete to be able to use the token (or tokens) of the place node. By way of example, it is easy to observe that place node P2 (in other words, adjustment machine M1 in a *free* state), acts as a precondition (in other words, input place node) both for the

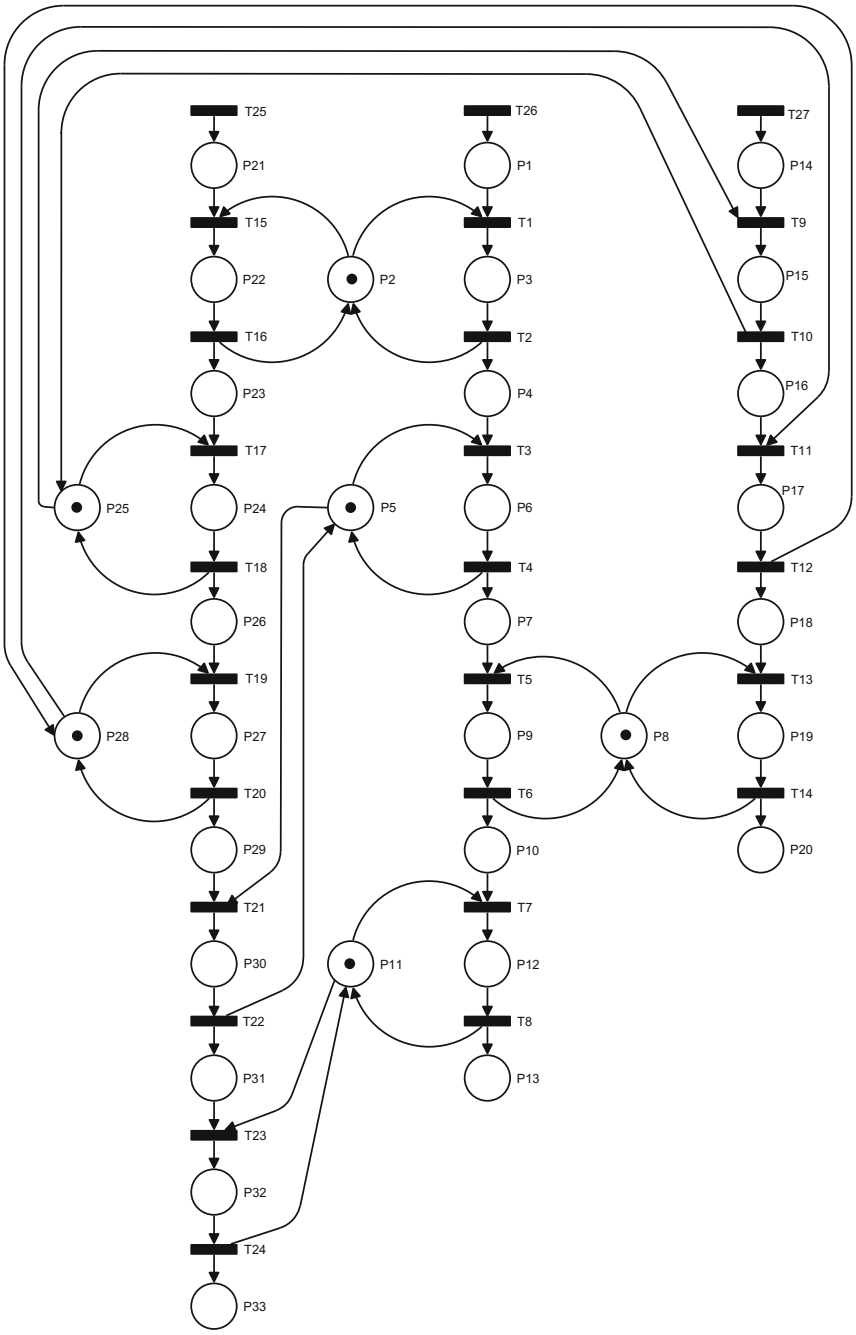

Fig. 3.7 PN process-oriented model for several production sequences

Table 3.4 Place nodes PN system with 3 production sequences

Place	Description
P1	Stock S1 of type Π_1 pieces waiting for an adjustment procedure
P2	Adjustment machine M1 in *free* state
P3	Adjustment machine M1 in *working* state on type Π_1 pieces
P4	Stock S3 of pieces waiting for a pressing procedure
P5	Machine press M3 in *free* state
P6	Machine press M3 in *working* state on type Π_1 pieces
P7	Stock S2 of pieces waiting for a drilling procedure
P8	Drilling machine M2 in *free* state
P9	Drilling machine M2 in *working* state on type Π_1 pieces
P10	Stock S6 of pieces waiting for a polishing procedure
P11	Polisher M6 in *free* state
P12	Polisher M6 in *working* state on type Π_1 pieces
P13	Stock S7 of already processed type Π_1 pieces: end product
P14	Stock S5 of type Π_2 pieces waiting for a molding procedure
P15	Molding machine M5 in *working* state on type Π_2 pieces
P16	Stock S4 of type Π_2 pieces waiting for a milling procedure
P17	Milling cutter M4 in *working* state on type Π_2 pieces
P18	Stock S2 of type Π_2 pieces waiting for a drilling procedure
P19	Drilling machine M2 in *working* state on type Π_2 pieces
P20	Stock S7 of already processed type Π2 pieces: end product
P21	Stock S1 of type Π_3 pieces waiting for an adjustment procedure
P22	Adjustment machine M1 in *working* state on type Π_3 pieces
P23	Stock S5 of type Π_3 pieces waiting for a molding procedure
P24	Molding machine M5 in *working* state on type Π_3 pieces
P25	Molding machine M5 in *free* state
P26	Stock S4 of type Π_3 pieces waiting for a milling procedure
P27	Milling cutter M4 in *working* state on type Π_3 pieces
P28	Milling cutter M4 in *free* state
P29	Stock S3 of type Π_3 pieces waiting for a pressing procedure
P30	Machine press M3 in *working* state on type Π_3 pieces
P31	Stock S6 of type Π_3 pieces waiting for a polishing procedure
P32	Polisher M6 in *working* state on type Π_3 pieces
P33	Stock S7 of already processed type Π_3 pieces: end product

Start adjustment procedure event on a type Π_1 piece (transition T1) and the *Start adjustment procedure* event on a type Π_3 piece (transition T15). In the case of firing transition T1, transition T15 shall be disabled and vice versa. Similarly, also a Decision/Conflict pattern of behavior is presented, but with 2 transitions (T7 and T23) that compete to be able to use the token of place node P11.

Table 3.5 Transition nodes in PN system with 3 production sequences

Transition	Description
T1	Start adjustment procedure on type Π_1 piece
T2	End adjustment procedure on type Π_1 piece
T3	Start pressing procedure on type Π_1 piece
T4	End pressing procedure on type Π_1 piece
T5	Start drilling procedure on type Π_1 piece
T6	End drilling procedure on type Π_1 piece
T7	Start polishing procedure on type Π_1 piece
T8	End polishing procedure on type Π_1 piece
T9	Start molding procedure on type Π_2 piece
T10	End molding procedure on type Π_2 piece
T11	Start milling procedure on type Π_2 piece
T12	End milling procedure on type Π_2 piece
T13	Start drilling procedure on type Π_2 piece
T14	End drilling procedure on type Π_2 piece
T15	Start adjustment procedure on type Π_3 piece
T16	End adjustment procedure on type Π_3 piece
T17	Start molding procedure on type Π_3 piece
T18	End molding procedure on type Π_3 piece
T19	Start milling procedure on type Π_3 piece
T20	End milling procedure on type Π_3 piece
T21	Start pressing procedure on type Π_3 piece
T22	End pressing procedure on type Π_3 piece
T23	Start polishing procedure on type Π_3 piece
T24	End polishing procedure on type Π_3 piece
T25	Arrival of type Π_3 pieces
T26	Arrival of type Π_1 pieces
T27	Arrival of type Π_2 pieces

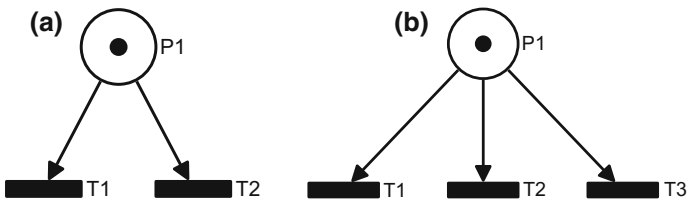

Fig. 3.8 Decision/conflict behavior pattern

Figure 3.8 generically shows the Decision/Conflict pattern of behavior. The left side of the same figure is characterized by a decision-making with 2 alternatives, while on the right side (Fig. 3.8b) there are 3 alternatives.

It is also easy to note that in the "*working* machine" procedures it is necessary to differentiate the type of piece being *worked* on so as to be able to redirect the piece to the next procedure. A similarly thing happens with the stocks which also have to distinguish the type of piece that they store, so different place nodes are required for the same stock.

Another pattern of behavior that is also shown in the PN of Fig. 3.7 is concurrency, which is characterized by two or more changes of state variables in the same instant of time. One example of concurrency is to be found in transition T2 as its firing causes a change of state in machine M1 (changes to be *free*) and at the same time increases by one the number of type Π_1 pieces in stock S3.

Figure 3.9 generically represents the pattern of concurrency behavior. The left side of the same Fig. 3.9 is characterized by a concurrency with 2 changes to the state variables, while on the right side (Fig. 3.9b), there are 3 changes to the state variables.

Finally, we also can see the pattern of synchronization behavior, which is usually utilized when waiting for a resource to continue with the sequence of procedures. One example can be seen in transition T3, that is waiting for a type Π_1 piece to arrive at stock S3 to carry out a pressing procedure. The left side of Fig. 3.10a is characterized by a synchronization of 2 resources, while on the right side (Fig. 3.10b) the synchronization of 3 resources is required.

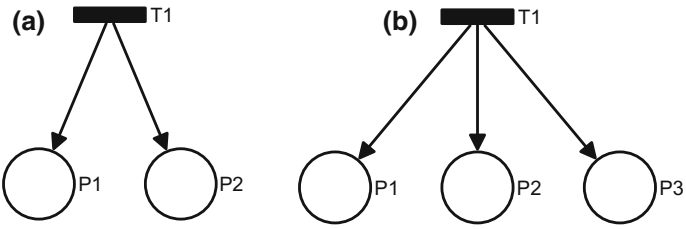

Fig. 3.9 Pattern of concurrency behavior

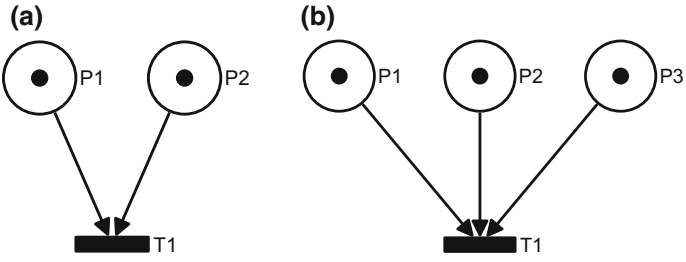

Fig. 3.10 Pattern of synchronization behavior

3.4 Redundant Place Nodes

The purpose is to facilitate the development, maintenance and comprehension of the models, the use of redundant nodes, which can be defined as a duplication of a node that must always maintain the same number of tokens, is admissible in the PN formalism.

Figure 3.11a represents the same PN described in Fig. 3.7, but using 2 redundant nodes for P2, P5, P8, P11, P25 and P28. On the right side of Fig. 3.11, we observe the same PN after having fired transition T1, in which one token has been eliminated from place nodes P1 and P2 and one token has been added to place node P3. It is important to note that in place node P2 connected to the input of transition T15, one token has also been eliminated.

Although colors have been used in Fig. 3.11 to identify the redundant nodes, the only rule there is in this regard is that the place redundant nodes must have the same identifying name.

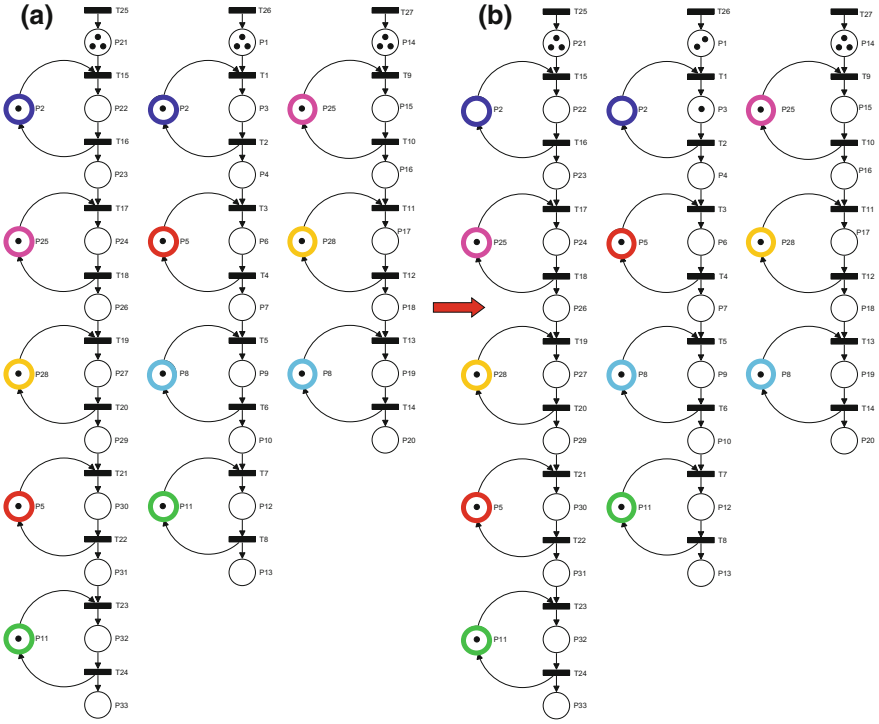

Fig. 3.11 PN model with redundant nodes for several production sequences

3.5 Limitations of Petri Nets

One of the great limitations to the PN formalism for the specification of simulation models for industrial, production, transport and logistics systems, is the inability to describe changes in the information about the entities: not only the resources but also the products that are moved by the production system. The inability to specify information changes in a more compact manner forces the modeler to resort to the specification of similar subnetworks (with the same patterns of behavior and dynamics) where they are only differentiated by the information associated with the place node. In the PN Model represented in Fig. 3.11 it is easy to see that the entire model is the integration of several Petri subnetworks similar to the one represented in Fig. 3.5, where only the information relating to the machine and the stock changes.

The representation of the information in the PN formalism by means of similar subnetworks will result in a considerable rise in the number of place nodes and transition nodes, thus hampering the maintenance and possible exploitation of the developed model (Guasch et al. 2003).

Furthermore, it is worth considering that the task of maintaining the model, which is necessary for the assessment by simulation of different behaviors of the system, becomes a hard and difficult task when the structures of the models are complex. Thus, despite the advantages of PN formalism, the limitations as regards being able to efficiently represent the information considerably restrict its use when it comes to modeling the systems' characteristics we want to represent to experiment with them in a simulation environment.

3.6 Colored Petri Nets

Colored petri nets (henceforth CPN) allow us to build compact and parametric models that would require structures with a high number of components if they were developed using PN formalism (Jensen 1997; Jensen and Kristensen 2009; CPN group 2015). As we have already pointed out, PN models of complex logistics systems are usually made up by similar subnetworks, so a more compact representation, in which the behavior of such subnetworks could be formalized through a single network, will considerably facilitate their maintenance for evaluating different possible settings of the system under study.

As most industrial systems require the specification of different types of attributes (characteristics) to describe the entities that flow through the system, we need to use a formalism that makes it easier to describe the flow of tasks to be performed as a function of said attributes, allowing us to stipulate:

- The priority of the entities in a queue.
- How an event can affect the values of the attributes of the entity being processed (state of the entity).
- Activation of events as a function of the attributes of each entity.

- Length of time of an activity as a function of the attributes of the entities involved.
- The flow of entities and what happens to each entity (changes of state of the entity) as it flows through a sequence of physical subsystems.
- The changes of state of the system and the sequence of events that would have to appear for an entity to finish a particular type of processing.

CPNs support a certain level of abstraction in the modeling stage by using colors to represent the attributes of the entities and which are supported by the majority of commercial simulation software packages.

3.6.1 *Elements Involved in the Modeling of Colored Petri Nets*

The main difference made by CPNs in respect of ordinary PNs is the ability to associate a type of data (set of values) known as *color* token with every entity (token). The value of a datum can be of an arbitrarily complex type; for example, a log where the first field is a real number, the second is a *string* text, while the third could be a list of whole numbers.

For a given place node, all the tokens must have colors that belong to the specified type. This type is called the *color set* of the place node. The use of color sets in CPN is completely analogous to the use of types in programming languages, which gives them the necessary power to be able to formalize the model of any system, no matter how complex.

Color sets determine the possible values of tokens, in a similar way to how types determine the possible values of variables and expressions. For historical reasons we talk about colored tokens that are distinguished from one another, in contrast to the flat tokens of ordinary PNs. However, we can also talk about values and types instead colors and color sets.

Apart from the basic elements of a PN described in Sect. 3.2 (place nodes, transition nodes and arcs), the concepts of color and color set introduce into a CPN the use of the following representation elements (Jensen 1997):

- **Color sets:** each place node can only have tokens with the same type of data, which is known as a color set of the place node. This constraint is totally compatible with the modeling objectives, always provided the place nodes represent either conditions or queues. This is graphically represented with the name of the color set to one of the place node.
- **Initialization expressions:** indicate the initial number of tokens and their color in each one of the place nodes. Graphically speaking, the initialization expressions express the number of tokens in a place node with a number in a circle beside the node. The colors of the tokens is specified by means of an underlined expression beside the place node with the following information:

$$\underline{n'(c_1, c_2, c_3, \ldots, c_r, \ldots, c_{nr})}$$

where:

- n: represents the number of tokens with the color values described inside the brackets.
- $\underline{c_r}$: represents the value of a color attribute.
- nr: represents the number of color attributes of the tokens.

When the values of the colors of the tokens are not identical for all the elements of the same place node, the operator "+" is used to specify the values of the colors of each token:

$$n1'(c_{11},\ c_{12},\ c_{13}, \ldots,\ c_{1r}, \ldots,\ c_{1nr}) + n2'(c_{21},\ c_{22},\ c_{23}, \ldots,\ c_{2r}, \ldots,\ c_{2nr})$$

- **Initial state:** the initial state (marking M0) shall be determined by assessing the initialization expressions associated with each place node, which shall determine the number of tokens in each place node, as well as the values of the colors of the tokens.
- **Arc expressions:** the colors of the tokens can be inspected in the transitions, which will make it possible to enable them not only according to the number of tokens in the place nodes connected to the input of the transition, but also according to the color values of the tokens available in said place node and, at the same time, will also make it possible to model the effects of each transition, defining new colors for the output tokens.
  Arc expressions consist of the formalization of constraints between the colors of the various tokens of the place nodes connected to the input of the transition, for which formalization variables that have been assigned specific token color values can be used to force a selection of those tokens whose colors coincide with the values of said variables.
- **Guards:** guards have a similar function to the arc expressions, but are only logical expressions (*Booleans*) that impose certain values on the colors of the tokens that can be chosen to enable a transition. They are graphically formalized between square brackets "[ ]" placed beside the transition.
- **Marking:** represents the minimum information required to be able to predict what possible events could be produced. It is described through the specification of the number of tokens in each place node, as well as the values of the colors of every one of the tokens.

3.6.2 Formal Definition of Colored Petri Nets

In formal terms, a CPN is defined as a tuple (Jensen 1997; Jensen and Kristensen 2009; Guasch et al. 2003; Narciso et al. 2010):

$$CPN = (\Sigma, P, T, A, N, C, G, E, I)$$

where:

Σ = {C1, C2, …, Cnc}: Finite and non-empty color sets (*nc* is the number of color sets specified for the CPN).

P = {P1, P2, P3, …, Pnp}: Finite set of place nodes (*np* is the number of place nodes in the CPN).

T = {T1, T2, T3, …, Tnt}: Finite set of transition nodes (*nt* is the number of transition nodes in the CPN).

A = {a1, a2, a3, …, ana}: Finite set of arcs that connect nodes *P* with nodes *T* and vice versa (*na* is the number of arcs in the CPN).

N: **Node** function, $N: A \rightarrow (P \times T) \cup (T \times P)$, that makes it possible to associate its terminal nodes with each arc in the form of an ordered pair, so that:

$$\forall \, Ai \in A \; \exists! \, Pj \in P \wedge \exists! \, Tk \in T : [N(Ai) = (Pj, Tk) \vee N(Ai) = (Tk, Pj)]$$

where the first element of the ordered pair corresponds to the origin node and the second element to the destination node. The two nodes have to be of different types, accordingly, if one node is a transition node the other must be a place node and vice versa.

C: **Color** function, $C: P \rightarrow \Sigma$, that makes it possible to specify, for each place node, the type of entities (tokens) that can stored, so that:

$$\forall \, P_j \in P \; \exists! \, Cq \in \Sigma : [C(Pj) = Cq]$$

G: **Guard** function, that allows us to associate each transition node with a logical expression, $G: T \rightarrow Boolean$, so that:

$$\forall \, Tk \in T : [type(G(Tk)) = Boolean \wedge type(variables(G(Tk)) \subseteq \Sigma]$$

E: **Arc expression** function, $E: A \rightarrow C(Pj)$ that makes it possible to specify the type of entity (token) of the input place node to a transition that must be chosen from among the tokens stored in said node to enable the transition, so that:

$$\forall \, Ai \in A : [type(E(Ai)) = C(Pj) \wedge type(variables(E(Ai))) \subseteq \Sigma]$$

where Pj represents the input or output place node from arc Ai.
When expression E is found associated with an output arc of the transition, the expression is used to assess the new color values of the attributes of the output entities (tokens).

I: **Initialization** function, $P \rightarrow C(Pj)$, that makes it possible to specify the color values of the attributes of the entities (tokens) initially stored in a place node, so that:

$$\forall P_j \in P : [\text{type}(I(Pj)) = C(Pj)]$$

Each one of these components of a CPN makes it possible to specify and/or represent any component of a simulation model for a DES, such as:

- The Σ set enables us to specify, for each type of entity to be modeled, the attributes that must be defined in the code of the simulation model.
- Each place node can represent, for example, one or more production units (machines) with one or more queues.
- The transitions in the model correspond to events that are usually encoded, such as the start or end of a certain activity,[4] or else the end of an external event as in the case of an arrivals process.
- Guard functions can be used to disinhibit the event associated with a transition as a function of the values of the attributes of an entity to be processed.

So CPNs provide us with the necessary knowledge representation tools to be able to formalize not only the attributes or characteristics of the entities that flow in the system, but also the properties that these entities need to have so that a certain event can happen, and said tools have demonstrated that they are the right ones for modeling logistical systems thanks to their various advantages, such as the ability to contain not only the static structure but also the dynamics of the system, the architecture of the system, and its graphic characteristics (Silva and Valette 1989; Zimmermann et al. 1996; Jensen 1997; Jensen and Kristensen 2009; Piera et al. 2009; Narciso 2010; Narciso and Piera 2015).

3.6.3 Behavior or Dynamics of Colored Petri Nets

As with the PN, the arc expressions indicate the necessary conditions for a transition to be activated. However, in a CPN it is not just enough for a place node to contain the number of tokens specified in the arc that goes from the place node to the transition, but it must also contain the color of the tokens that will enable said transition, while the condition expressed by the guard associated with the transition

[4]As with the PNs, depending on the abstraction made of the system to be modeled, an activity can be represented in a CPN as a place node or a transition node.

should also, when applicable, be satisfied. For example, let us consider the CPN that is formalized in Fig. 3.12, which represents a production machine with an incoming stock (place node P1), and 2 ongoing stocks where the type Π_1 pieces (place node P3) are stored and the type Π_2 pieces (place node P2). Place nodes P4 and P5 are used to indicate the *working* and *free* state of the machine respectively. Transition T1 indicates the "Start production operation" event, while event T2 represents the "End production operation on piece Π_1", and event T2 represents "End production operation on piece Π_2". Initially there are 3 type Π_1 pieces and three type Π_2 pieces in the stock coming into the machine that are represented by the initialization expression $\underline{3'(1) + 3'(2)}$ located at the side of node P1, and the machine is in a *free* state that is presented by the initialization expression $\underline{1'(1)}$ located at the side of place node P5. The rest of the place nodes do not have any token.

The initial marking is represented mathematically by the following vector:

$$M0 = [3'(1) + 3'(2), \ldots, 1'(1)]$$

where the tokens of each place node are delimited by the symbol ",". Thus, we can observe that tokens 3′(1) + 3′(2) correspond to place node P1 and token 1′(1) corresponds to place node P5, while there is no token in place nodes P2, P3 and P4.

For transition T1 in Fig. 3.12 to be enabled it is necessary for the following conditions to be fulfilled:

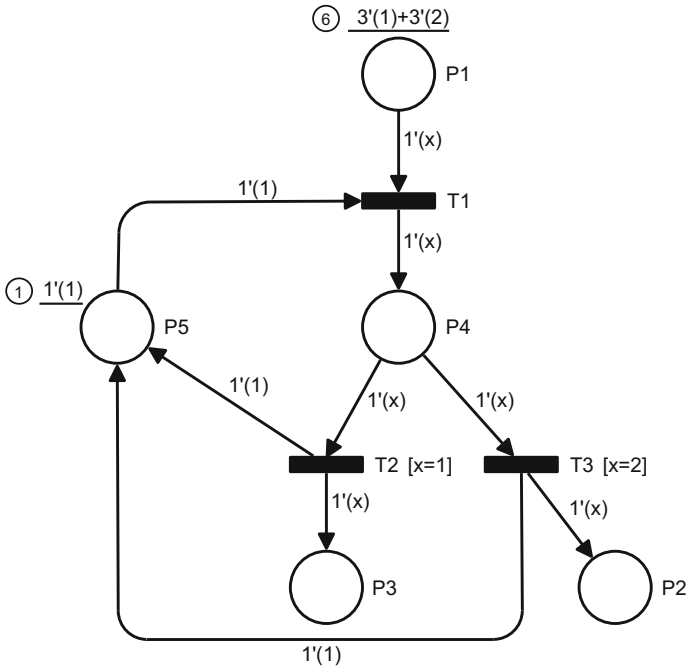

Fig. 3.12 CPN model of a production machine with 2 types of pieces

- There must be at least one token with any color in place node P1. Arc expression $1'(x)$ indicates one token whose color will be instantiated in variable x. Thus, if a color 1 token (in other words, $1'(1)$) is chosen, variable x will be instantiated at value 1. Whereas, if a color 2 token (in other words, $1'(2)$) is chosen, variable x will be instantiated at value 2.
- In place node P5 there must be at least one token with the color 1 (arc expression $1'(1)$).

As an effect of firing transition T1 one token is eliminated from place node P1 and one token is eliminated from place node P5, and a new token shall be added to place node P4 with the value of variable *x*. Thus, in the case of firing transition T1 with one token of color 1 (in other words, one type Π_1 piece) from place node P1, the new state that would be achieved is represented by vector M1. If a type Π_2 piece (in other words, token of color 2 of place node P1) is chosen, the new state that would be achieved is represented by vector M2.

$$\begin{aligned} \text{M1} &= [2'(1) + 3'(2), \ldots, 1'(1),] \\ \text{M2} &= [3'(1) + 2'(2), \ldots, 1'(2),] \end{aligned}$$

Given state M1, transition T2 is enabled because the guard associated with said transition demands that variable x takes a value of 1 (note that there is only one token with value 1 in place node P4). Similarly, given state M2, transition T3 is found enabled as the guard associated with said transition demands that variable x take a value of 2. Vector M3 corresponds to the state that is reached if transition T2 is fired from state M1, while vector M4 corresponds to the state that is reached if transition T3 is fired from state M2.

$$\begin{aligned} \text{M3} &= [2'(1) + 3'(2), , 1'(1), \ldots, 1'(1)] \\ \text{M4} &= [3'(1) + 2'(2), 1'(2), \ldots, 1'(1)] \end{aligned}$$

In this sense, the CPN formalism provides a formal method that permits us to represent and fully assess its behavior. This method is known as the CPN *scope tree* or occurrence graph (Jensen 1997; Narciso 2010).

Example 3.3 CPN Model of different sequences of production operations on different types of pieces while maintaining a flow of products

Let us consider that, in this example, we have to process 3 different types of pieces, each one with a sequence of production operations set forth in the recipe described in Table 3.3. Table 3.6 gives the information about the place nodes, Table 3.7 defines the colors used and Table 3.8 represents the information about the transition nodes, while Table 3.9 gives the information about the arc expressions.

Figure 3.13 graphically represents the CPN of the various sequences of production operations on the 3 types of pieces.

Initially all the place nodes are empty except for the place nodes that represent the machines in *free* state, whose tokens are:

Table 3.6 Process-oriented CPN place nodes for a system with 3 production sequences

Place	Color	Description
P1	C1	Stock S1 of type Π_1, Π_2 and Π_3 pieces, waiting the first production operation
P2	C2	Adjustment machine M1 in *free* state
P3	C2	Adjustment machine M1 in *working* state on type Π_1 pieces
P4	C1	Stock S3 of pieces waiting for a pressing procedure
P5	C2	Machine press M3 in *free* state
P6	C2	Machine press M3 in *working* state on type Π_1 pieces
P7	C1	Stock S2 of pieces waiting for a drilling procedure
P8	C2	Drilling machine M2 in *free* state
P9	C2	Drilling machine M2 in *working* state on type Π_1 pieces
P10	C1	Stock S6 of pieces waiting for a polishing procedure
P11	C2	Polisher M6 in *free* state
P12	C2	Polisher M6 in *working* state on type Π_1 pieces
P13	C1	Stock S7 of finished type Π_1, Π_2, and Π_3 pieces
P14	C2	Molding machine M5 in *free* state
P15	C2	Molding machine M5 in *working* state on type Π_2 pieces
P16	C1	Stock S4 of type Π_2 pieces waiting for a milling procedure
P17	C2	Milling cutter M4 in *working* state on type Π_2 pieces
P18	C1	Stock S2 of type Π_2 pieces waiting for a drilling procedure
P19	C2	Drilling machine M2 in *working* state on type Π_2 pieces
P20	C2	Adjustment machine M1 in *working* state on type Π_3 pieces
P21	C1	Stock S5 of type Π_3 pieces waiting for a molding procedure
P22	C2	Modeling machine M5 in *working* state on type Π_3 pieces
P23	C1	Stock S4 of type Π_3 pieces waiting for a milling procedure
P24	C2	Milling cutter M4 in *working* state on type Π_3 pieces
P25	C2	Milling cutter M4 in *free* state
P26	C1	Stock S3 of type Π_3 pieces waiting for a pressing procedure
P27	C2	Machine press M3 in *working* state on type Π_3 pieces
P28	C1	Stock S6 of type Π_3 pieces waiting for a polishing procedure
P29	C2	Polisher M6 in *working* state on type Π_3 pieces

Table 3.7 Definition of colors C1 and C2

Color	Definition	Description
C1	Integer 1…3	Identification of type of piece
C2	Integer 1…6	Identification of machine or stock

$$P2 = 1'(1); \quad P5 = 1'(3); \quad P8 = 1'(2); \quad P11 = 1'(6); \quad P14 = 1'(5); \quad P25 = 1'(4)$$

Unlike the production system described in Example 3.2, in this system the arrivals process of type Π_1, Π_2, Π_3 pieces has been modeled by means of events T1, T2 and T3 respectively. As can be seen in the CPN of Fig. 3.13, these events

Table 3.8 Process-oriented CPN transition nodes for a system with 3 production sequences

Transition	Description
T1	Arrival of type Π_1 piece
T2	Arrival of type Π_2 piece
T3	Arrival of type Π_3 piece
T4	Start adjustment procedure on type Π_1 piece
T5	End adjustment procedure on type Π_1 piece
T6	Start pressing procedure on type Π_1 piece
T7	End pressing procedure on type Π_1 piece
T8	Start drilling procedure on type Π_1 piece
T9	End drilling procedure on type Π_1 piece
T10	Start polishing procedure on type Π_1 piece
T11	End polishing procedure on type Π_1 piece
T12	Start molding procedure on type Π_2 piece
T13	End molding procedure on type Π_2 piece
T14	Start milling procedure on type Π_2 piece
T15	End milling procedure on type Π_2 piece
T16	Start drilling procedure on type Π_2 piece
T17	End drilling procedure on type Π_2 piece
T18	Start adjustment procedure on type Π_3 piece
T19	End adjustment procedure on type Π_3 piece
T20	Start molding procedure on type Π_3 piece
T21	End molding procedure on type Π_3 piece
T22	Start milling procedure de of type Π_3 piece
T23	End milling procedure on type Π_3 piece
T24	Start pressing procedure on type Π_3 piece
T25	End pressing procedure on type Π_3 piece
T26	Start polishing procedure on type Π_3 piece
T27	End polishing procedure on type Π_3 piece

Table 3.9 Process-oriented CPN arc expressions for a system with 3 production sequences

Arc	Expression
a1, a4, a5, a6, a7, a8, a9, a10, a12, a13, a15, a16, a18, a19, a21, a22, a24, a25, a27, a48, a50	1′(1)
a2, a17, a20, a28, a30, a31, a33, a34, a36, a37, a38, a40, a41, a42, a43, a44, a45	1′(2)
a3, a11, a14, a46, a47, a49, a51, a52, a54, a56, a57, a59, a60, a62, a63, a64, a65, a66, a67, a68, a69, a71, a72, a74, a75	1′(3)
a35, a39, a58, a61	1′(4)
a29, a32, a53, a55	1′(5)
a23, a26, a70, a73	1′(6)

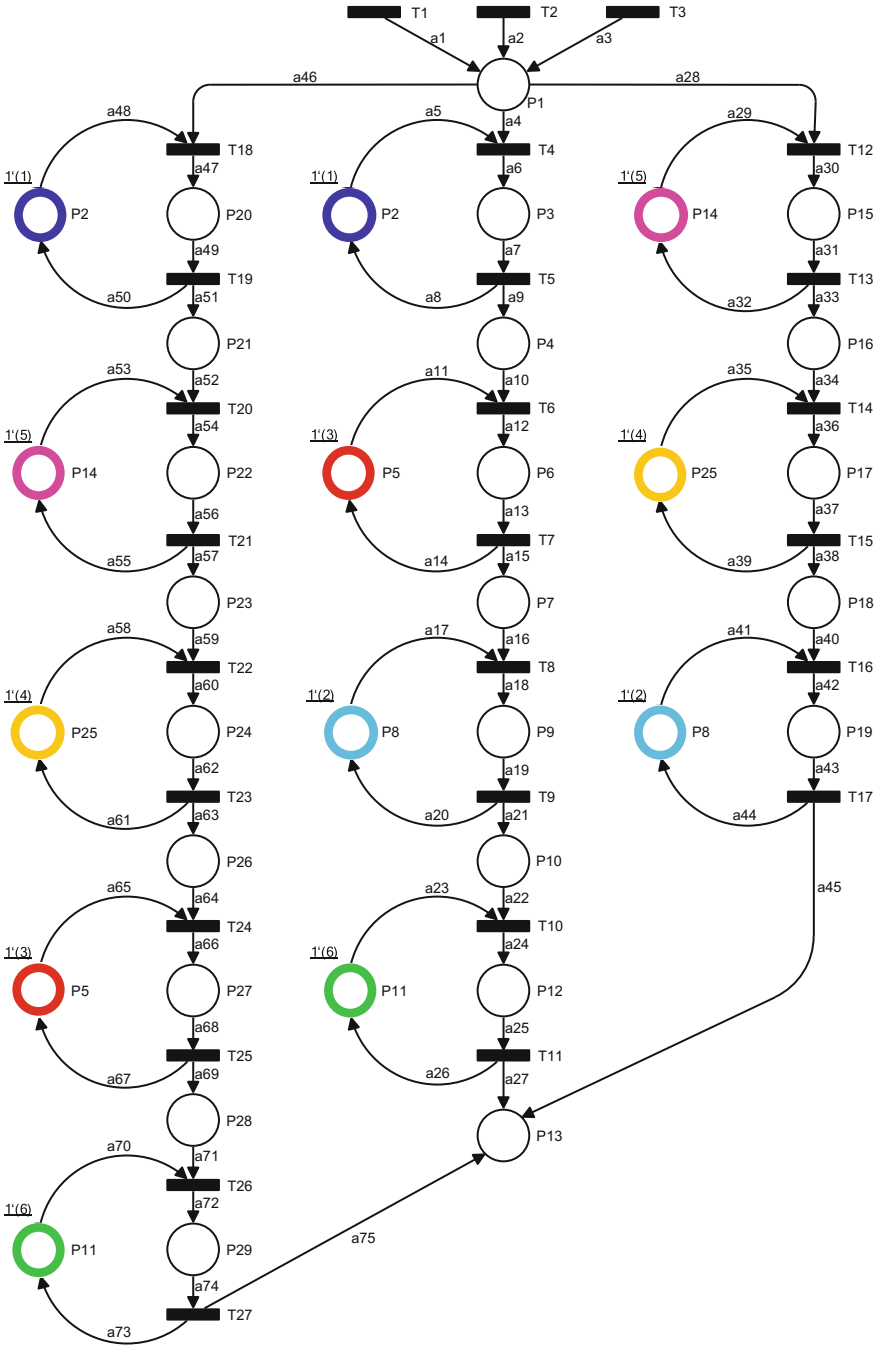

Fig. 3.13 Process-oriented CPN model

are always activated (there is no precondition) so, every time they are fired, one token 1′(1) is added in the case of T1, 1′(2) in the case of T2, or 1′(3) in the case of T3 to the stock of pieces to be produced. Given the initial state that has been described, the only events that are activated are precisely T1, T2 and T3 which, once fired, shall permit different sequences of events to be fired depending on the availability of production resources (machines in *free* state), just like in the model obtained in Example 3.2.

Example 3.4 CPN Layout-oriented model

Considering the system in the earlier example, Fig. 3.14 presents another formalization of the same production system at a level of abstraction where the flow of events considers a layout orientation, instead of a process orientation as represented in Fig. 3.13. Table 3.10 gives the information about the place nodes, Table 3.11 defines the colors used, Table 3.12 represents the information the transition nodes, while Table 3.13 represents the information about the arc expressions.

3.7 Timed Colored Petri Nets

The CPNs can be extended if the concept of time is incorporated into the models. In the CPNs this concept is based on the introduction of a discrete *global clock*. The value of the clock represents the time model, and each token has an associated time value, also known as *time stamps*. In intuitive terms, the timestamp describes the earliest time model in which a token can be used, in other words, when it can be eliminated from a place node, as a consequence of the firing of a transition. This means that the timestamp of the tokens that are to be eliminated must be less or equal to the current time model. The CPN model remains in the current time model while there are tokens that could be used to fire any of the transitions of the CPN (Jensen 1997).

Once all the possible transitions for the model have been fired in current time, the global clock updates in accordance with the time when a transition is enabled. In order to model an activity or event that corresponds to a transition that requires r units of time, a timestamp, which is r units of time greater than the model in which the transition was fired is associated and located in the tokens added to the output place nodes of said transition. The tokens will not then be available during r units of time and cannot be eliminated by firing the transitions, before the time model has been increased by at least r units of time (Jensen 1997; Kristensen and Christensen 2004; Narciso et al. 2012).

So, in order to incorporate the concept of time into the CPN, we need to introduce the following modeling elements into this model:

- A *global clock* that represents the time immediately before the occurrence of an event or the firing of a transition.
- A *time value* (timestamp) associated with each token in a marking, that describes the smallest time value where a token can be used to enable a

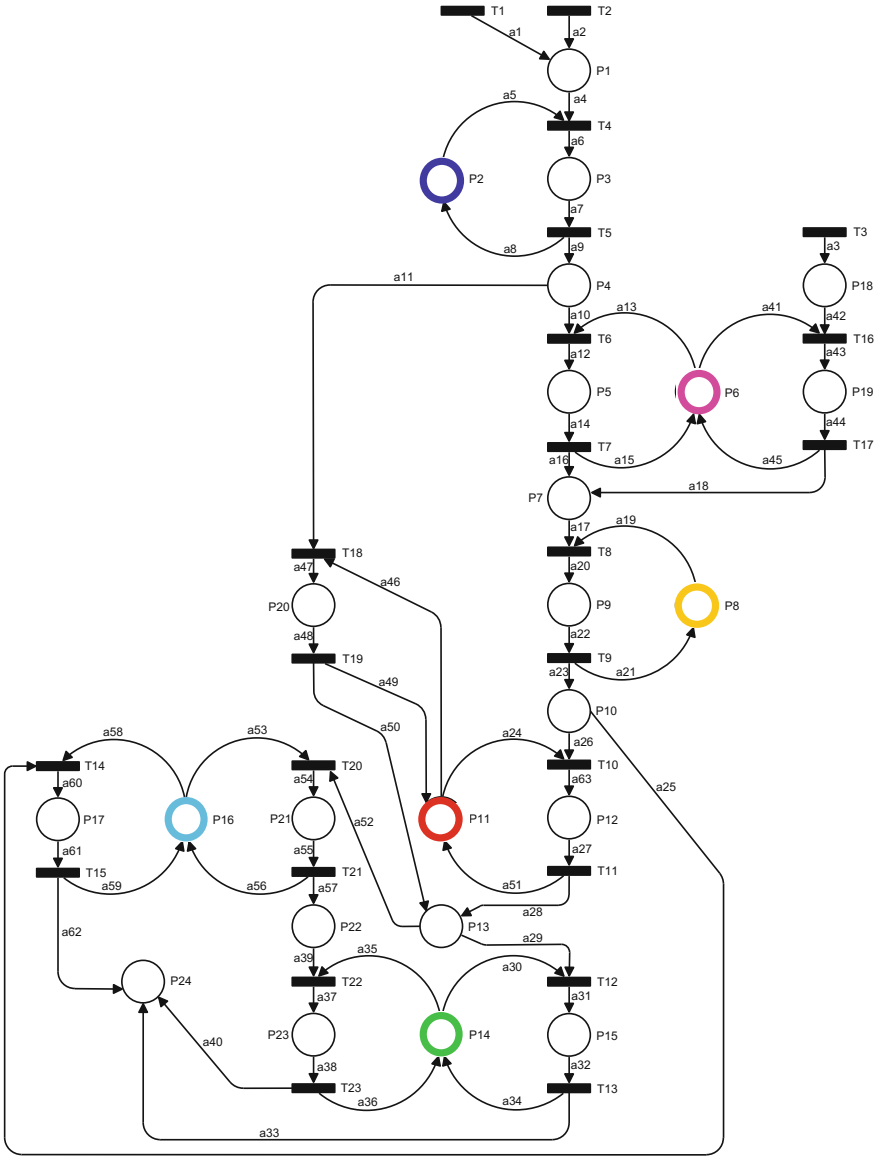

Fig. 3.14 Layout-oriented model

transition, in other words, it indicates when the token is "ready" to be used by a transition.

- A *delay time* (r) associated with a transition, that indicates, for example, that an activity modeled through a transition consumes time. In this chapter the term *transition time* is used to represent this concept.

Table 3.10 CPN layout-oriented place nodes for a system with 3 production sequences

Place	Color	Description
P1	C1	Stock S1 of type Π_1 and Π_3 pieces, waiting the first production operation: Adjustment
P2	C2	Adjustment machine M1 in *free* state
P3	C1	Adjustment machine M1 in *working* state on type Π_1 or Π_3 pieces
P4	C1	Stock S3 of type Π_1 and Π_3 pieces that have completed the adjustment procedure
P5	C1	Molding machine M5 in *working* state on type Π_3 pieces
P6	C2	Molding machine M5 in *free* state
P7	C1	Stock S2 of type Π_3 and Π_2 pieces that have finished the molding procedure
P8	C2	Milling cutter M4 in *free* state
P9	C1	Milling cutter M4 in *working* state on type on type Π_2 or Π_3 pieces
P10	C1	Stock S6 of type Π_2 and Π_3 pieces that have completed the milling procedure
P11	C2	Machine press M3 in *free* state
P12	C1	Machine press M3 in *working* state on type Π_3 pieces
P13	C1	Stock S7 of type Π_1 and Π_3 pieces that have completed the pressing procedure
P14	C2	Polishing machine M6 in *free* state
P15	C1	Polishing machine M6 in *working* state on type Π_3 pieces
P16	C2	Drilling machine M2 in *free* state
P17	C1	Drilling machine M2 in *working* state on type Π_2 pieces
P18	C1	Stock S3 of type Π_2 pieces waiting for the first production operation: molding
P19	C1	Molding machine M5 in *working* state on type Π_2 pieces
P20	C1	Machine press M3 in *working* state on type Π_1 pieces
P21	C1	Drilling machine M2 in *working* state on type Π_1 pieces
P22	C1	Stock S5 of type Π_1 pieces with the drilling procedure finished
P23	C1	Polishing machine M6 in *working* state on type Π_1 pieces
P24	C1	Stock S4 of type Π_1, Π_2 and Π_3 pieces with all the procedures completed

Table 3.11 Definition of colors C1 and C2

Color	Definition	Description
C1	Integer 1…3	Identification of the type of piece
C2	Integer 1…6	Identification of machine or stock

- An *arrival time* of a marking that represents the time it takes the firing of a transition to change the state of the system to a new state.

In this case, a transition will not only be enabled when the tokens satisfy the corresponding arc input expressions of the transition, but these tokens must also be "ready". This means that all the times (timestamps) associated with the tokens that enable the transition, should be shorter than or equal to the current value of the

Table 3.12 CPN layout-oriented transition nodes for a system with 3 production sequences

Transition	Description
T1	Arrival of type Π_1 piece
T2	Arrival of type Π_3 piece
T3	Arrival of type Π_2 piece
T4	Start procedure to adjust type Π_1 or Π_3 piece
T5	End adjustment procedure on type Π_1 or Π_3 piece
T6	Start molding procedure on type Π_3 piece
T7	End molding procedure on type Π_3 piece
T8	Start milling procedure on type Π_2 or Π_3 piece
T9	End milling procedure on type Π_2 or Π_3 piece
T10	Start pressing procedure on type Π_3 piece
T11	End pressing procedure on type Π_3 piece
T12	Start polishing procedure on type Π_3 piece
T13	End polishing procedure on type Π_3 piece
T14	Start drilling procedure on type Π_2 piece
T15	End drilling procedure on type Π_2 piece
T16	Start molding procedure on type Π_2 piece
T17	End molding procedure on type Π_2 piece
T18	Start pressing procedure on type Π_1 piece
T19	End pressing procedure on type Π_1 piece
T20	Start drilling procedure on type Π_1 piece
T21	End drilling procedure on type Π_1 piece
T22	Start polishing procedure on type Π_1 piece
T23	End polishing procedure on type Π_1 piece

Table 3.13 CPN layout-oriented arc expressions for a system with 3 production sequences

Arc	Expression
a4, a6, a7, a9, a17, a20, a22, a23	1′(x)
a1, a5, a8, a11, a37, a38, a39, a40, a47, a48, a50, a52, a54, a55, a57	1′(1)
a3, a18, a25, a42, a43, a44, a45, a53, a56, a58, a59, a60, a61, a62	1′(2)
a2, a10, a12, a14, a16, a24, a51, a26, a63, a27, a28, a29, a31, a32, a33, a46, a49	1′(3)
a19, a21	1′(4)
a13, a15, a41, a45	1′(5)
a30, a34, a35, a36	1′(6)

clock. Otherwise, the global clock must advance to the lowest timestamp value of the tokens for which the transition would be enabled.

To illustrate how the timed CPNs behave, let us look again at the CPN in Fig. 3.12, and assume:

- A global clock initialized at 0.
- A delay time of 10 units associated with transition T2 that indicates the time required to complete a production operation on a type Π_1 piece.
- A delay time of 4 units associated with transition T3 that indicates the time required to complete a production operation on a type Π_2 piece.
- Transition T1 does not consume time.

The symbol @ is used to indicate the information about time. So, if we assume that for the initial marking all the tokens have associated times that are equal to zero, M0 can be represented, with its corresponding timing information:

$$\mathrm{M0} = [3'(1)\ @0 + 3'(2)\ @0, \ldots, 1'(1)\ @0]$$

The three tokens of value 1 in place node P1 have a token time 0. The same information is read in the rest of the tokens. If we consider that the tokens of value 1 in P1 arrived in the instants of time 2, 4 and 6, the marking would be represented by the following vector:

$$\mathrm{M0} = [1'(1)\ @2 + 1'(1)\ @4 + 1'(1)\ @6 + 3'(2)\ @0, \ldots, 1'(1)\ @0]$$

Figure 3.1 represents the timed RdPC.

Below the evolution of the state of the system is represented, considering the firing sequence of transitions T1:1′(2) − T3 − T1:1′(1) − T2 − T1:1′(1), after initial state M0 represented in Fig. 3.15 and clock time 0.00:

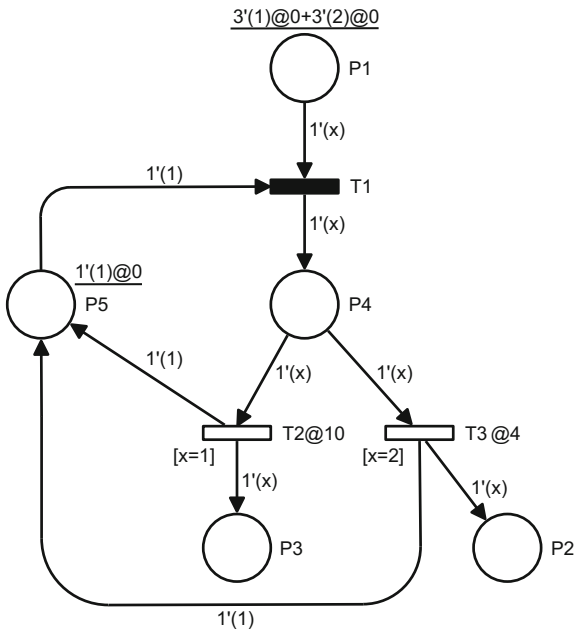

Fig. 3.15 Timed CPN model of a production machine with 2 types of pieces

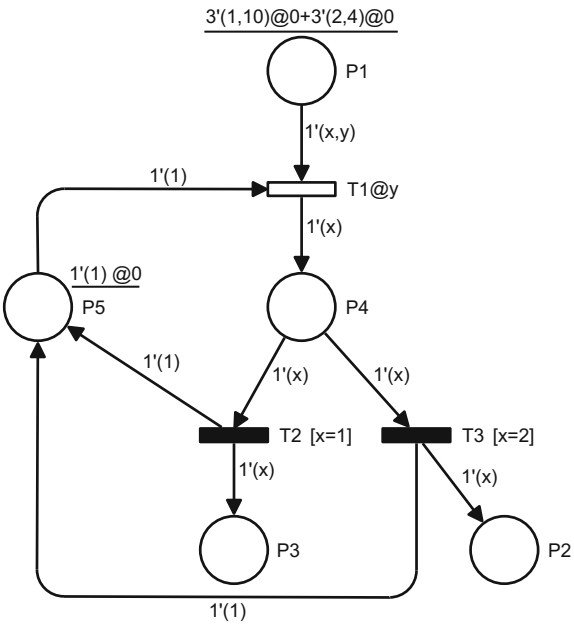

Fig. 3.16 Timed CPN model of a production machine with 2 types of pieces, using color to specify the time

$$
\begin{aligned}
\text{M0} &= [3'(1)\,@0 + 3'(2)\,@0, \ldots, 1'(1)\,@0]\ \text{Clock} : 0.00\\
\text{M1} &= [3'(1)\,@0 + 2'(2)\,@0, \ldots, 1'(2)\,@0,]\ \text{Clock} : 0.00\\
\text{M2} &= [3'(1)\,@0 + 2'(2)\,@0, 1'(2)\,@4, \ldots, 1'(1)\,@4]\ \text{Clock} : 0.00\\
\text{M3} &= [2'(1)\,@0 + 2'(2)\,@0, 1'(2)\,@4, \ldots, 1'(1)\,@4,]\ \text{Clock} : 4.00\\
\text{M4} &= [2'(1)\,@0 + 2'(2)\,@0, 1'(2)\,@4, 1'(1)\,@14, \ldots, 1'(1)\,@14]\ \text{Clock} : 4.00\\
\text{M5} &= [1'(1)\,@0 + 2'(2)\,@0, 1'(2)\,@4, 1'(1)\,@14, 1'(1)\,@14,]\ \text{Clock} : 14.00
\end{aligned}
$$

With the incorporation of timing into the models, it is possible to use the CPN to assess not only the behavior but also the performance of a system (Fig. 3.16).

Figure 3.1 represents the same system, but the time in transition T1 has been associated, using the second color attribute of the tokens in place node P1 (variable *y*), which represents the time required by each type of piece to complete the corresponding production operation. The dynamics of the system obtained are exactly the same but with a different representation of the information, in that the time of the tokens increases at the start of the production operation and not at the end but it keep the same evolution of the simulation clock, as can be seen in the evolution of the states

$$
\begin{aligned}
\text{M0} &= [3'(1,10)@0 + 3'(2,4)@0, \ldots, 1'(1)@0] \text{ Clock : 0.00} \\
\text{M1} &= [3'(1,10)@0 + 2'(2,4)@0, \ldots, 1'(2)@4,] \text{ Clock : 0.00} \\
\text{M2} &= [3'(1,10)@0 + 2'(2,4)@0, 1'(2)@4, \ldots, 1'(1)@4] \text{ Clock : 4.00} \\
\text{M3} &= [2'(1,10)@0 + 2'(2,4)@0, 1'(2)@4, \ldots, 1'(1)@14,] \text{ Clock : 4.00} \\
\text{M4} &= [2'(1,10)@0 + 2'(2,4)@0, 1'(2)@4, 1'(1)@14, \ldots, 1'(1)@14] \text{ Clock : 14.00} \\
\text{M5} &= [1'(1,10)@0 + 2'(2,4)@0, 1'(2)@4, 1'(1)@14, 1'(1)@24,] \text{ Clock : 14.00}
\end{aligned}
$$

CPN formalism also enables us to specify the consumption of time in the place nodes so that they can represent activities. Figure 3.1 represents the same system associating the timed activity with place node P4, which uses the second color attribute (variable *y*) that indicates the time that each piece requires to complete the production operation (Fig. 3.17).

The evolution of the states is shown below:

$$
\begin{aligned}
\text{M0} &= [3'(1,10)\,@0 + 3'(2,4)\,@0, \ldots, 1'(1)\,@0] \text{ Clock : 0.00} \\
\text{M1} &= [3'(1,10)\,@0 + 2'(2,4)\,@0, \ldots, 1'(2,4)\,@0,] \text{ Clock : 0.00} \\
\text{M2} &= [3'(1,10)\,@0 + 2'(2,4)\,@0, 1'(2)\,@4, \ldots, 1'(1)\,@4] \text{ Clock : 0.00} \\
\text{M3} &= [2'(1,10)\,@0 + 2'(2,4)\,@0, 1'(2)\,@4, \ldots, 1'(1,10)\,@4,] \text{ Clock : 4.00} \\
\text{M4} &= [2'(1,10)\,@0 + 2'(2,4)\,@0, 1'(2)\,@4, 1'(1)\,@14, , 1'(1,10)\,@14] \text{ Clock : 4.00} \\
\text{M5} &= [1'(1,10)\,@0 + 2'(2,4)\,@0, 1'(2)\,@4, 1'(1)\,@14, 1'(1,10)\,@24,] \text{ Clock : 14.00}
\end{aligned}
$$

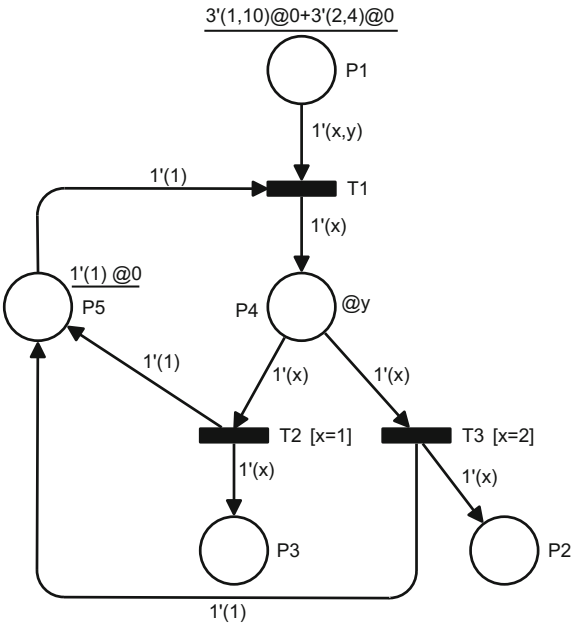

Fig. 3.17 Timed CPN model for a production machine with 2 types of pieces with the time associated with the place node

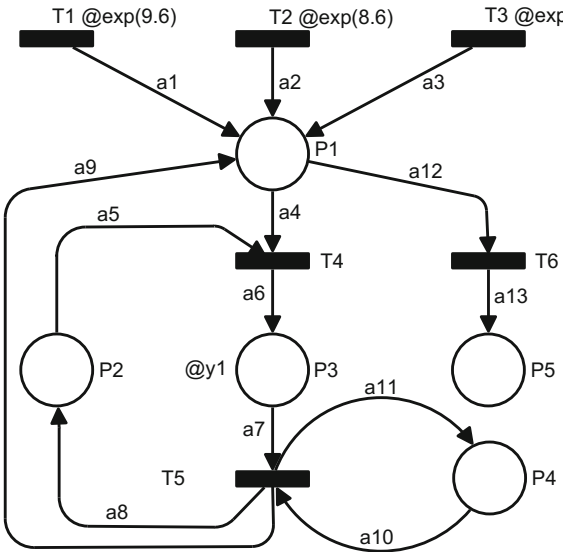

Fig. 3.18 Timed CPN model for a production system with 3 types of pieces

Example 3.5 Timed CPN model of different sequences of production operations on different types of pieces

Figure 3.18 represents the system described in Example 3.3, but with timed representation, where the pieces arrive at the production system following an exponential probability density function with different parameters. Tables 3.14, 3.15, 3.16 and 3.17 describe the information represented by the place nodes, the colors used, the transition nodes and the arc expressions, respectively.

Table 3.18 gives the production times in every one of the machines for each type of piece.

The initial conditions of place node P4 are:

Table 3.14 Description of place nodes of a system's CPN with 3 production sequences

Place	Color	Description
P1	C1	Stock of type Π1, Π_2 and Π_3 pieces
P2	C2	Machine in *free* state
P3	C3	Machine in *working* state
P4	C4	Sequence de scheduled procedures
P5	C1	Stock of finished type Π_1, Π_2, and Π_3 pieces

Table 3.15 Definition of colors C, C1, C2, C3 and C4

Color	Definition	Description
CP	Integer 1…3	Identification of the type of piece
CT	Integer	Time assigned to a production operation
C1	Product CP*C2*CT	Identification of the type of piece, stock where it is to be found, as well as the time required in the following production operation
C2	Integer 1…6	Identification of machine in *free* state
C3	Product CP*C2*CT	Identification of type of piece, machine in *working* state and length of time of the procedure
C4	Product CP*C2*C2*CT	For each type of piece it specifies the completed procedure, the next production operation to be performed and the required time

Table 3.16 Description of transition nodes of a system's CPN with 3 production sequences

Transition	Description
T1	Arrival of type Π_1 piece
T2	Arrival of type Π_2 piece
T3	Arrival of type Π_3 piece
T4	Start production operation
T5	End production operation
T6	End sequence of procedures

Table 3.17 CPN arc expressions for a system with 3 production sequences

Arc	Expression
a1	1′(1, 1, 125)
a2	1′(2, 5, 105)
a3	1′(3, 1, 135)
a4, a6, a7	1′(x, y, z)
a5, a8	1′(y)
a9	1′(x, y1, z1)
a10, a11	1′(x, y, y1, z1)
a12	1′(x, 7)
a13	1′(x)

Table 3.18 Production times

Type of piece	Sequence procedures with times
Π_1	Adjustment (125′)—Pressing (35′)—Drilling (20′)—Polishing (60′)
Π_2	Molding (105′)—Milling (90′)—Drilling (65′)
Π_3	Adjustment (135′)—Molding (250′)—Milling (50′)—Pressing (30′)—Polishing (25′)

$$\begin{aligned} \mathrm{P4} \;= {} & 1'(1,1,3,35) + 1'(1,3,2,20) + 1'(1,2,6,60) \\ & + 1'(1,6,7,0) + 1'(2,5,4,90) + 1'(2,4,2,65) \\ & + 1'(2,2,7,0) + 1'(3,1,5,250) + 1'(3,5,4,50) \\ & + 1'(3,4,3,30) + 1'(3,3,6,25) + 1'(3,6,7,0) \end{aligned}$$

Chapter 4 explains how PN and CPN implemented in SIMIO.

References

CPN Group, University of Aarhaus. (2015). http://cs.au.dk/cpnets/

Guasch, A., Piera, M. A., Casanovas, J., & Figueras, J. (2003). *Modelado y simulación: aplicación a procesos logísticos de fabricación y servicios* (2a ed.). Barcelona: Ediciones UPC.

Jensen, K. (1997). *Coloured petri nets: Basics concepts, analysis methods and practical use* (Vol. 1, 2, 3). Berlin: Springer.

Jensen, K., & Kristensen, L. M. (2009). *Coloured petri nets: Modelling and validation of concurrent systems*. Berlin: Springer.

Kristensen, L. M., & Christensen, S. (2004). Implementing coloured petri nets using a functional programming language. In *Higher-Order and Symbolic Computation* (Vol. 17, pp. 207–243). Netherlands: Kluwer Academic Publishers.

Narciso, M. E., & Piera, M. A. (2015). Robust gate assignment procedures from an airport management perspective. *Omega The International Journal of Management Science, 50*, 82–95.

Narciso, M. E., Piera, M. A., & Guasch, A. (2010). A methodology for solving logistic optimization problems through simulation. In *SIMULATION: Transactions of the Society for Modeling and Simulation International* (Vol. 86(5–6), pp. 369–389).

Narciso, M. E., Piera, M. A., & Guasch, A. (2012). A time stamp reduction method for state space exploration using colored Petri nets. In *SIMULATION: Transactions of the Society for Modeling and Simulation International* (Vol 88(5), pp. 592–616).

Petri Nets World, Universidad de Hamburgo. (2015). http://www.informatik.uni-hamburg.de/TGI/PetriNets/

Piera Eroles, M. A., Narciso Farias, M. E., & Buil Giné, R. (2009). Flexible manufacturing systems. In *Simulation-Based Case Studies in Logistics. Education and Applied Research*. London: Springer.

Proth, J.-M., & Xie, X. (1996). *Petri nets: A tool for design and management of manufacturing systems*. Inglaterra: Wiley.

Silva, M., & Valette, R. (1989). Petri nets and flexible manufacturing. In *Lecture Notes in Computer Science, Advances in Petri Nets* (Vol. 424, pp. 374–417).

Wang, J. (1998). *Timed petri nets: Theory and application*. The Kluwer International Series on Discrete Event Dynamic Systems. Norwell, Massachusetts: Kluwer Academic Publishers.

Zaremba, M. B., & Prasad, B. (1994). Modern manufacturing: Information control and technology. In *Advanced Manufacturing Series*. Berlin: Springer.

Zhou, M., & Venkatesh, K. (1999). Simulation and control of flexible manufacturing systems: A petri net approach. In *Series in Intelligent Control and Intelligent Automation* (Vol. 6). Singapore, New Jersey, London, Hong Kong: World Scientific Publishing.

Zimmerman, A. (1995). Modeling of manufacturing systems and production routes using colored petri nets. In *Proceedings of the 3rd IASTED International Conference on Robotics and Manufacturing* (pp. 380–383).

Zimmermann, A., Dalkowski, K., & Hommel, G. (1996). A case study in modelling and performance evaluation of manufacturing systems using colored petri nets. In *Proceedings of the 8th European Simulation Symposium, ESS '96* (pp 282–286).