Extended Time Petri Net and Hybrid Petri Net : Modeling Multi-Instance Dynamic Hybrid Systems

Yamen El Touati†, Mohamed Ayari†† and Saleh Altowaijri†††

†Department of Computer Sciences,
††Department of Information Technology
†††Department of Information System,
Faculty of Computing and Information Technology,
Northern Border University, Kingdom of Saudi Arabia.

Summary
Hybrid Petri nets (HPN) are commonly used for describing Hybrid Dynamic Systems (DHS). Among these DHS, HPN are more convenient to systems where the dynamic is essentially flows such as massive manufacturing systems, liquid chemical transforming, etc.
However, an important number of hybrid systems are mainly described by instances that could not be assimilated to flows. HPN formalism is not adequate for these systems and number of nodes tend to be very important.
In another hand Time Petri nets are very useful for describing constraints of real time systems and especially systems which are event guided and where the system state can be described via several instances. Extended TPN [4,5] is an extension of TPN including continuous variable which allow modelling of DHS.
This paper aim to provide a systematic way to describe Extended Time Petri net with equivalent hybrid Petri nets in order to simplify the modeling process for systems where HPN is not the adequate model.
Key words:
Hybrid Petri Nest, Time Petri Nets, Dynamic Hybrid Systems.

1. Introduction

A wide range of computer automated systems, such us man-made systems, real-time and embedded systems are composed of continuous processes that are supervised and reconfigured by a discrete control. Systems in which the discrete and the continuous aspects interact, and where these interactions influence the system's quantitative and qualitative behavior, are called Dynamic Hybrid Systems (DHS) [8].

Tasks related to DHS, such as modeling, supervision and analysis, often pose complicated and challenging problems. Two types of communities are interested in DHS models: the discrete event systems (DES) community and the continuous systems community.

Within the community of continuous systems, DHS are modeled as systems that transition among various continuous models. This enables the researchers and engineers with continuous systems background to apply readily available techniques from the continuous systems literatures [10]. Nevertheless, performing computations and analysis with such models can easily become a daunting task, especially for hybrid systems with a strong discrete component, which exhibit frequent switchings between a multitude of different continuous models.

Currently within the DES field, several different modeling frameworks are being used for modeling DHS. The most commonly used amongst them are timed and hybrid extensions of Petri nets and automata. From the side of timed Petri nets, we distinguish Timed Petri nets [2] and Time Petri nets (TPN) [1]; both are based on autonomous Petri nets, enhanced with time constraints. However, in these models, the continuous aspects of the hybrid system they model, are synthetically reduced to a mere passage of time. This makes it difficult for a system designer/engineer to easily work with the resulting models, as the real quantities and parameters of the system become masked and untangled with time quantities.

Amongst the hybrid extensions based on Petri nets, timed Hybrid PNs [7], control hybrid PN, predicate transition PNs [6] and mixed PNs are the most commonly used. All these models suffer from a scalability problem, in terms the number of configuration and/or transitions. We shall illustrate this in later sections, for the case of hybrid Petri nets.

In this paper, we are interested in modelling a class of hybrid systems with a strong sequential component, a relatively simple continuous component. We believe that for a model to useful, from a practical perspective, it must truly capture a system's quantitative aspects, such as temperatures, pressures, distances and velocities, in an as faithful way as possible. This permits systems designers and engineers to more easily go back and forth between the model, the system, and the problems they are attempting to solve, in an as seamless way as possible. Moreover, the systems we consider here, require the memorization of thresholds and various intermediary values, in between

Manuscript received May 5, 2018
Manuscript revised May 20, 2018

nondeterministic preemptions and switchings across various different systems activities.

This paper is organized as follows. In the next section, we characterize the Time Petri net formalism and an intersting extension to model DHS. In section 3, we give hybrid PN models for a few hybrid systems. In section 4 we illustrate some case study modelling. Our aim is to show that while hybrid PNs are well suited for modeling hybrid dynamic systems with continuous flows, they easily lead to complex and unscalable models for an important class of hybrid systems. In section 5, we present a systematic method to transform an ETPN into HTPN.

2. Time Petri Net and its extension

2.1 Time Petri Nets

We start by defining autonomous Petri nets (PN).

Definition 1. A marked autonomous Petri Net is a 5-tuplet (P, T, •(.), (.)•, M_0) in which:

- P is a non empty finite set of places;
- T is a non empty finite set of transitions;
- $\bullet(.) \in (\{0, 1\}^P)^T$ is the backward function;
- $(.)\bullet \in (\{0, 1\}^P)^T$ is the forward function;
- $M_0 \in \mathbb{N}^P$ is the initial marking;

A transition Ti is considered as enabled if $M \geq {}^{\bullet}T_i$. A making M that enables the transition T_i leads to a new making M_0 by firing T_i ; The marking M_0 is given by $M_0 = M - {}^{\bullet}T_i + T_i^{\bullet}$. We consider the relation $\uparrow$enabled(T_k, M, T_i) $\in$ {true, false} defined in [12] and based on [1]. $\uparrow$enabled(T_k, M, T_i) is true if T_k is enabled by the firing of transition T_i from the marking M and f alse otherwise. Formally, $\uparrow\text{enabled}(T_k, M, T_i) = (M - {}^{\bullet}T_i + T_i^{\bullet} \geq {}^{\bullet}T_k) \wedge ((M - {}^{\bullet}T_i < {}^{\bullet}T_k) \vee (T_i = T_k))$. Thus, we consider that a transition, T_k, is newly enabled after firing T_i from the marking M if it is not enabled by $M - {}^{\bullet}T_i$ and is enabled by $M_0 = M - {}^{\bullet}T_i + T_i^{\bullet}$ [1].

Time Petri Net (TPN) [1,2] is basically a discrete formalism capturing continuous time intervals. The Continuous aspect is reduced to time flow. In what follows we present the definition of the TPN formalism.

Definition 2. A Time Petri Net is a 6-tuplet (P, T, •(.), (.)•, M_0, IO) in which:

- (P, T, • (.), (.) • , M_0) is an autonomous marked Petri net
- $IO \in (Q+ \times (Q+ \cup +\infty))^T$ is the time constraints function. For a transition T_i, $IO(T_i) = [EFT(T_i), LFT(T_i)]$, $0 \leq EFT(T_i) \leq LFT(T_i)$ designates the firing interval of T_i ;

The IO function defines the firing interval of the transition. The two end-points of the interval are referred to as static earliest and latest firing times. Thus, a transition T_i of a TPN is considered as enabled, if it is enabled in usual Petri net sense. Moreover, the enabled transition T_i may not fire before its earliest firing time EFT(T_i) and must fire before its latest firing time LFT(T_i) unless another transition fires before and modifies the previous marking.

An ETPN [4,5] contains two kinds of transitions: transitions that are in nature temporal (called **time transitions**), as used in Time Petri Nets [1], and transitions that are hybrid in nature (called **linear transitions**). Typically, a time transition is used for modeling the elapse of time, while a linear transition is used to model a continuous behavior that can be described with a linear evolution.

During the enabling of a time transition, a clock, initialized to zero, is started. This clock changes its value with a constant speed equal to one. The corresponding temporal activity can be triggered at any time within the transition firing interval.

In an ETPN, a token carries information capturing values of continuous variables. These values, may change through the execution of linear transitions. At the enabling of a linear transition, T_i, a variable is initialized with the value carried by the token at a particular place, preceeding T_i, called the active place. The progression of the value of this variable occurs only when the designated T_i is enabled, and this progression occurs with a constant speed. The firing of the transition can happen anytime the value of the continuous variable is within the range of the firing interval. However, this firing must happen before the value of the variable becomes outside the firing interval (the rate of change can be negative and the value of the variable may incremented to the max limit or decremented to the min limit). When the value of the variable reaches either limits, the corresponding transition instantaneously fires. When a (time or linear) transition is enabled, the initial value is either a rational constant (non memory case) or a value transmitted from token in a place (memory case). The memory case, is typically used to model the transmission of continuous values between activities, whereas, the non memory case, is used to reset an activity independently of previous activities. In other words, a non memory case at a transition activity corresponds to a constant initial value that is independent of previous attempted values, while a memory case at a transition activity corresponds to a initial value recovered from a previous activity and transmitted by token in the input place. The formal definition of an ETPN is given as follows.

Definition 3. An Extended Time Petri Net is a 6-tuplet (P, T, •(.), (.)•, M$_0$, IO) in which:

- (P, T, • (.), (.) • , M$_0$) is an autonomous marked PN
- A : T ⟶ $(\mathbb{Q}^* \cup \Delta) \times \mathbb{Q} \times \mathbb{Q} \times (P \cup \mathbb{Q})$, is the transition activity relation, where, A(Ti) = (d$_i$, a$_i$, b$_i$, P$_j$), or A(T$_i$) = (d$_i$, a$_i$, b$_i$, k), with;
  - o d_i , is the evolution speed of the activity associated with T_i (the Δ value corresponds to a time constraint). We use also $AD(T_i)$ to denotes d_i
  - o a_i, b_i, are the bounds of the firing interval such that, $a_i \leq b_i$. We use also $Aa(T_i)$ and $Ab(T_i)$ to denote respectively the a_i and b_i bounds.
  - o $P_j \in {}^{\bullet}(T_i)$, is the active place, and serves to determine the initial value of the continuous variable carried by its token in a memory case. We use also $AP(T_i)$ to denote P_j.
  - o $k \in \mathbb{R}$, is the initialization value, in non-memory case. note also that, $AP(T_i)$, denotes k.

The token can hold the current value of the continuous variables, which are rational values. The semantic of an ETPN is defined using a Timed Transition System (TTS) [11] as follows.

Definition 4. The semantic of an ETPN $R = (P,\ T,\ {}^{\bullet}(.),\ (.)^{\bullet},\ A,\ M_0)$ is given by a timed transition system $S_R = (Q, q_0, \rightarrow)$ with

- $Q = \mathbb{N}^P \times \mathbb{R}^{2n+m}$ with $n = |T|,\ m = |P|$. $q = (M, v) \in Q$ if $v = (v^t, v^x, v^P)$, with $v^t \in \mathbb{R}_+^n$ is the time component and $v^x \in \mathbb{R}^n$ is the linear component, and $v^P \in \mathbb{R}^m$ is the place-memory component.
- $q_0 = (M_0, \overline{0})$ is the initial state.
- $\rightarrow \in (Q \times (T \cup \mathbb{R}_+) \times Q)$ is defined by :
- $(M, v)T_i \longrightarrow (M', v'),\ T_i \in T$ with $v = (v^t, v^x, v^P)$, $v' = (v'^t, v'^x, v'^P)$, $v^t, v'^t \in \mathbb{R}_+^n,\ v^x, v'^x \in \mathbb{R}^n$, $v^P, v'^P \in \mathbb{R}^m$ (**discrete transition relation**) if

$$\left(\begin{array}{l}
M \geq^{\bullet} T_i \quad (1) \\
Aa(T_i) \leq v^t(T_i) \leq Ab(T_i) \quad if\ \ AD(T_i) = \Delta \quad (2) \\
Aa(T_i) \leq v^x(T_i) \leq Ab(T_i) \quad if\ \ AD(T_i) \neq \Delta \quad (3) \\
M' = M - {}^{\bullet}T_i + T_i^{\bullet} \quad (4) \\
\forall T_k \in T \\
\left(\begin{array}{l}
v'^t(T_k) = \left(\begin{array}{l} 0\ \ if \uparrow enabled(T_k, M, T_i) \quad (5) \\ v^t(T_k)\ \ otherwise \quad (6) \end{array}\right. \\
v'^x(T_k) = \\
\left(\begin{array}{l} AP(T_k)\ \ if \uparrow enabled(T_k, M, T_i) \wedge AP(T_k) \in \mathbb{R} \quad (7) \\ v'^P(AP(T_k))\ \ if \uparrow enabled(T_k, M, T_i) \wedge AP(T_k) \in P \quad (8) \\ v^x(T_k) \quad otherwise \quad (9) \end{array}\right.
\end{array}\right. \\
\forall P_k \in P, \qquad v'^P(P_k) = \\
\left(\begin{array}{l}
v^x(T_i)\ \ if\ \ P_k \in (T_i)^{\bullet} \quad (10) \\
0\ \ if\ \ (P_k \in^{\bullet} (T_i) \wedge P_k \notin (T_i)^{\bullet}) \quad (11) \\
v^x(T_j)\ \ if\ \ \exists T_j \in T\ \ s.t. \\
\quad (P_k = AP(T_j) \wedge M'(P_k) > 0 \\
\quad \wedge (M - P_k) > 0 \wedge M \geq^{\bullet} (T_j) \wedge M' <^{\bullet} (T_j)) \quad (12) \\
v^P(P_k)\ \ otherwise \quad (13)
\end{array}\right.
\end{array}\right.$$

$(M, v) \overset{\varepsilon(t)}{\longrightarrow} (M', v'), \varepsilon(t) \in \mathbb{R}_+$ with $v = (v^t, v^x, v^P)$ and $v' = (v'^t, v'^x, v'^P)$ (**continuous transition relation**) if

$$\left(\begin{array}{l}
M' = M \quad (14) \\
\forall T_k \in T, \qquad M \geq^{\bullet} (T_k), \\
\left(\begin{array}{l}
(v'^t(T_k) = v^t(T_k) + t \wedge v'^x(T_k) = v^x(T_k)) \\
\quad if\ \ \ AD(T_k) = \Delta \quad (15) \\
(v^t(T_k) = v'^t(T_k) = 0 \wedge v'^x(T_k) = v^x(T_k) + t * AD(T_k)) \\
\quad if\ \ \ AD(T_k) \neq \Delta \quad (16)
\end{array}\right. \\
\forall T_k \in T, \qquad M \geq^{\bullet} (T_k) \Rightarrow \\
\left(\begin{array}{l}
(v'^t(T_k) \leq Ab(Tk)\ \ if\ \ AD(T_k) = \Delta \quad (17) \\
(v'^x(T_k) \geq Aa(Tk)\ \ if\ \ AD(Tk) < 0 \quad (18) \\
(v'^x(T_k) \leq Ab(Tk)\ \ if\ \ AD(Tk) > 0 \quad (19)
\end{array}\right. \\
\forall P_k \in P, \qquad v'^P(P_k) = v^P(P_k) \quad (20)
\end{array}\right.$$

- In our case, the firing is considered with a Strong Time Semantic (STS) as in [5]. According to STS semantic, an enabled transition must fire within its firing interval unless it is disabled by firing another transition.

3. Hybrid Petri Net and its extension

We start with the definitions of hybrid Petri nets, particularly an extension called time hybrid Petri net.

Hybrid Petri nets are presented in [7] to model systems combining discrete and continuous aspects. Under a unified graphical formalism, hybrid Petri nets [7, 13] allow for the representation of the discrete system dynamics using ordinary transitions and places, and represent dynamics using places (called continuous places) whose markings are real positive numbers (rather than integers). Continuous transitions correspond to continuous flows. A continuous transition T_i is fired with a speed $V_i(t)$. This means that between t and $t + dt$ a quantity $V_i(t).dt$ is removed from its input place and is deposited in its output places. An intuitive interpretation of the firing concept of a continuous transition can be illustrated, for example, by modeling the flow of an hourglass.

The influence of the discrete part on the continuous part is captured by elementary loops, connecting the discrete places to some of the continuous transitions. The influence of the continuous part on the discrete one (thresholds firing by continuous variables) is manifested through elementary loops linking continuous places (representing the tested variables) to discrete transitions.

Hybrid Petri nets allow a homogeneous representation of both the continuous aspects (flows) and the discrete aspect of a system, within the same formalism. This allows them to easily model liquid-transfer systems or continuous-mode batch systems or manufacturing systems where the flows of parts are easily approximated by a continuous flow [13] . However, when we introduce algebraic constraints, we must add equations to the model and do a regular update of variables. This causes the loss of the principal interest of hybrid Petri nets, and can therefore no longer represent in a single formalism the discrete and continuous parts.

The following definition, from [7], formally captures the marked autonomous hybrid Petri net.

Definition 5. A marked autonomous hybrid Petri net is a 6-tuplet R=(P, T, •(.), (.)•, M_0, h) fulfilling the following conditions:

- (P, T, • (.), (.) • , M_0) is an autonomous marked Petri net
- $h \in \{D,\ C\}^{(P \cup T)}$ called "hybrid function", indicates for every nodes wether it is a discrete node (sets P^D and T^D), or a continuous one (sets P^C and T^C);

In the definition of •(.), (.)•et M_0, ℕ correspond to the case where $P_i \in P^D$, and ℚ or $\mathbb{R}^+$ corresponds to the case where $P_i \in P^C$. Time can be added to the autonomous hybrid Petri net as in the definition 6.

Definition 6. A timed hybrid Petri net is a pair $(R, Tempo)$ where [7]:

- R is a marked autonomous hybrid PN
- Tempo: T $\longrightarrow \mathbb{Q}_+$
- if $T_j \in T^D$, $d_j = tempo(T_j)$ = timing associated with T_j
- if $T_j \in T^C$, $U_j = 1/tempo(T_j)$ = flow rate associated with T_j.

Let us now present the concept of time hybrid Petri net in the following definition.

Definition 7. A time hybrid Petri net (HTPN) is the triplet $(R,\ IO,\ U)$ where :

- R is a marked autonomous hybrid PN
- $IO \in (\mathbb{Q}_+ \times (\mathbb{Q}_+ \cup +\infty))^{T^D}$ is similar to definition 2 but limited to discrete transitions in T^D;
- $U \in \mathbb{Q}_+^{T^C}$ is flow rate associated with continuous transitions.

In definition 7, discrete transitions are associated with firing intervals according to the definition of time Petri nets [1]. Continuous transitions are associated with flow rates according to the definition of the continuous transitions of timed hybrid Petri nets [7]. The firing speed is the flow rate multiplied by the D-enabling degree of the transition, i.e. the enabling degree if only the discrete places are taken into account.

4. Modelling with TPN and HTPN

we start, in this section, by giving the salient features and exposing the main characteristics of the types of DHS that are the focus of our paper.

Generally speaking, systems within this class of DHS consist of a multitude of preemptible activities which can occur in parallel or sequentially. Each activity can exhibit both continuous as well as sequential behaviors. The important features of this class of DHS are detailed as follows.

- The continuous aspects of each activity composing the DHS evolve in a linear, or more precisely, constant speed fashion.
- The behavior of the DHS is dominated by discrete transitions.
- The DHS may involve a set of (alternate) tasks that may occur sequentially or in parallel. Each task may involve several stages, with each stage consisting of a set of dependent continuous activities. The parameters/values reached in one stage may constitute the initial values needed by later stages of the task.

Consider a chemical solution filling system illustrated in Figure 1. Filling action consists of two phases. At the first phase, the system fills a tray with a first chemical solution with a speed of $2cm^3/s$. At the start of the filling process, we assume that the tray contains $10 liters$ of a neutral liquid, such us water. The first action is terminated when the tray reaches a volume between $30l$ and $50l$. After the end of the first action, the system has at maximum $18\ seconds$ to finish the second action, otherwise, the final solution is considered inappropriate. The start of the second phase can be triggered by an authorization at any time. After authorization, the system must complete the task within at most $16\ seconds$. Once the second phase is started, a second chemical solution is filled with the speed $4cm^3/s$, until reaching a total volume between $70l$ and $90l$.

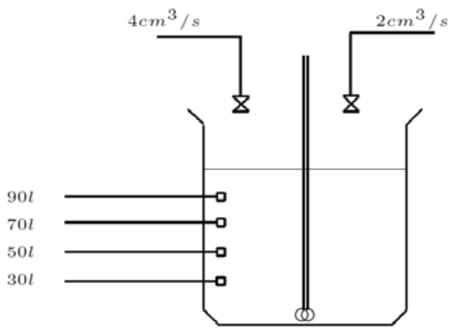


Fig. 1 Filler of chemical solution system

The system ETPN is given in Figure 2. The first (respectively second) filling phase is modeled by the linear transition T_1 (respectively T_4) in the ETPN model. The initial volume of the tray is given by $AP(T_1) = 10$. The transition T_2 reflects the authorization for beginning the second phase. This transition is a time transition, which justifies the Δ value of its dynamic. Firing T_2 leads to the enabling of transition T_4, and so, the starting of the second filling action. The initial value of the activity in T_4 transition is retrieved from token in the place P_2 in accordance within the T_4-activity definition, $AP(T_4) = P_2$. This value is carried from the activity value reached after firing T_1. Firing the T_3 (respectively the T_5) transition means that 18 $seconds$ (respectively 16 $seconds$) have been elapsed since the transition T_1 (respectively T_2) fires. The output places, P_6 of transitions T_4 represents the end of the filling action successfully. Non-successful situations are modeled by places P_5 and P_7.

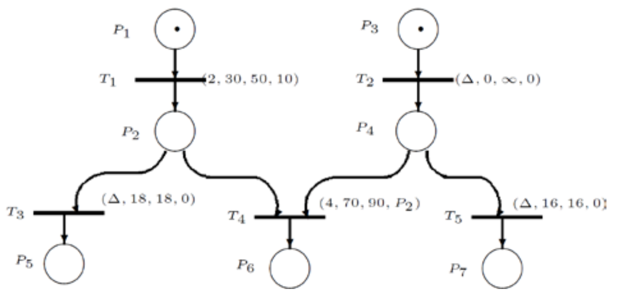


Fig. 2 ETPN of the filler of chemical solution

The first (respectively second) phase of filling is represented by a dotted rectangle labeled phase 1 (phase 2, respectively). The firing of transition T_4 models the fact that the water level reached the first threshold, which means that the end of the first phase (water filling) is possible. The firing of transition T_5 represents the end of the filling process. The arc between T_5 and P_5 has a weight 50, to return the amount that the model substituted artificially during the first passage over the threshold. Firing transition T_6 allows the start of the second phase of filling. Two conditions are necessary for firing transition T_6: the first condition is the presence of a token in place P_7 indicating the end of the first phase of filling; the second is the presence of a token in place P_3 indicating validation of the time constraint that starts the second phase. The firing of transition T_9 indicates the end of phase 2. The continuous place P_5 indicates the total level of liquid in the tank.

The first phase precede usually the second phase. Only one phase at a time is considered. The system specification can tolerate pipeline execution. In that case, if we are in phase 2, phase 1 can be started for a new filling action for another tray. In general, if several sequential instances of such activities occur, we'll be constrained to duplicate place P_5 several times (using a transition with infinite speed), so as not to interfere with actions that are in pipeline. Each one of The places P_5, models the corresponding water level. Parallel activities are modeled by a duplication of the various branches of the Petri net, in order to model activities (and not constraints, such us capacity etc.). A simplification of the graphical representation could be achieved by using a colored extension [7] of the basic model, in order to represent the different branches.

In the general case, hybrid Petri nets memorize the continuous state. However, taking into account a time interval gives easily model more complex and illegible, which leads to the loss of the general interest of Petri nets.

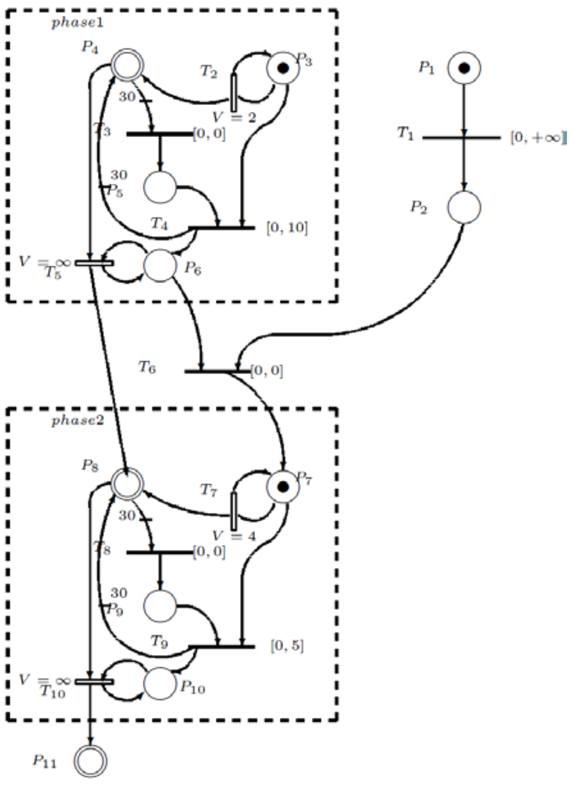


Fig. 3 HTPN of the filler of chemical solution

In this thermal treatment and processing system, parts undergos a cycle of heating and cooling processes. We assume that there are two pallets for transporting parts and an unlimited storage, between the heater and furnace cooling tray. Transfers take intermediate between zero and one unit of time. Initially, the part temperature is $20°$, it undergoes a heating in the oven with a speed of $2°/s$ until it reaches a temperature between $80°$ and $100°$. At the end of heating, the part is in stock deposited thermally insulated. Then it is deposited in a tray to undergo a dynamic cooling with a $-1°/s$. Cooling stops when the part reaches a temperature between $15°$ and $20°$. After putting the parts into the intermediate stock, the part is heated in the oven one last time with a speed of $4°/s$. Finally, the task of the system for that part ends when the part temperature reaches a value between $50°$ and $60°$. Note also that the furnace and the cooling tank can contain only one part at a time.

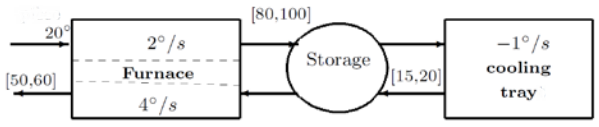


Fig. 4 Thermic Processing System

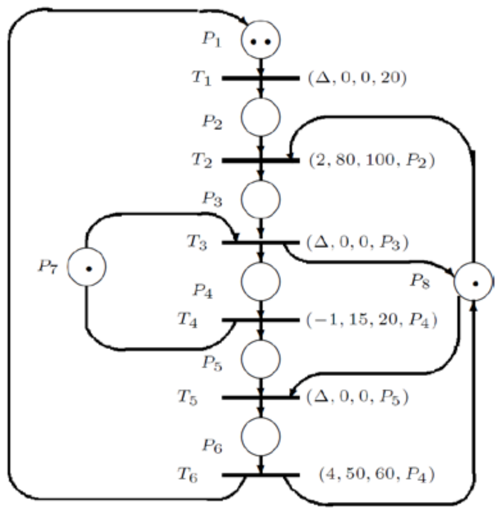


Fig. 5 ETPN of the thermic Processing System

Figure 5 describes the ETPN modeling of the heating system, in accordance with definition 5. Places P_8 and P_1 represents the availability of the furnace and of the cooling tray, respectively. In our case, place P_1 is initialized with the value 20 corresponding to the initial temperature: fourth terms in $(\Delta, 0,0,20)$. When transition T_1 is enabled (initially, it is already enabled), it recovers the initial value of the token in P_1, ie the value corresponding to $20 = IP(P_1)$. The first heater is activated by firing of transition T_1, indicating freedom of the oven. The evolution of room temperature is modeled by the validation of transition T_2; the initial value of the activity is recovered from the token in place P_2 (already containing the value 20). The rate of change is 2. The end of the activity can be reached when the value is in the interval [80, 100].

A change in the system specification in the form of a change in the number of pallets corresponds to a change of marking at the level of the ETPN without having to change the structure of the PN. Consider, for example, that the heat treatment process with initially 4 pallets. This change of the specification has no effect on the structure of the PN in Figure ?. We simply change the initial marking of P_1 to the value 4. However, with models based on hybrid PN, two additional branches would be needed for the modeling of a 4 pallet specification. This in turn correspond to doubling the number of places and transitions in the model, thus visibly, increasing its complexity and hindering its readability. A colored control hybrid PN may be used [7].

In Figure 6 we model the behavior of the thermic treatment system by a time hybrid Petri net. In this model, the behavior of discrete transitions is that defined in time Petri nets by associating a firing interval to each transition. transitions are associated with discrete firing intervals with the semantics of time Petri nets. The activity corresponding to the first heater causes the firing of either the transition T_1 or T_{17}. We can cut the Petri net into six areas as in the previous example.

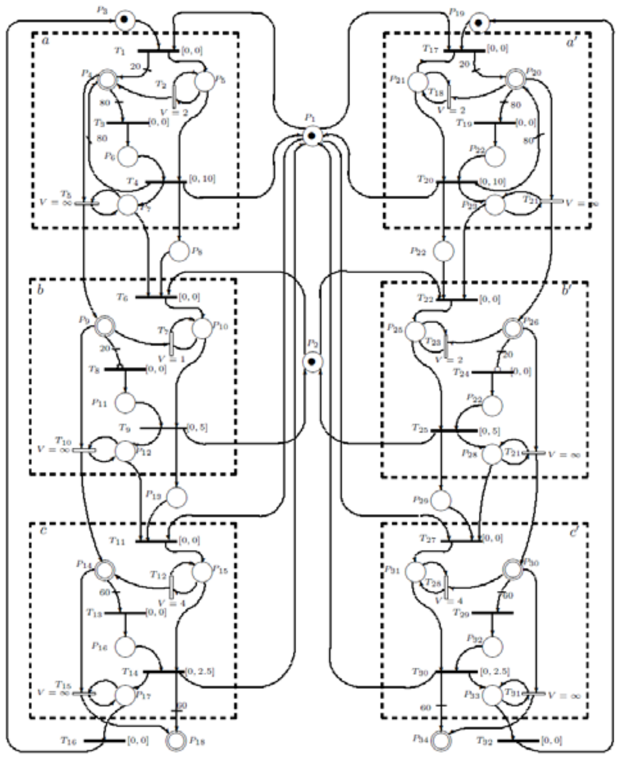
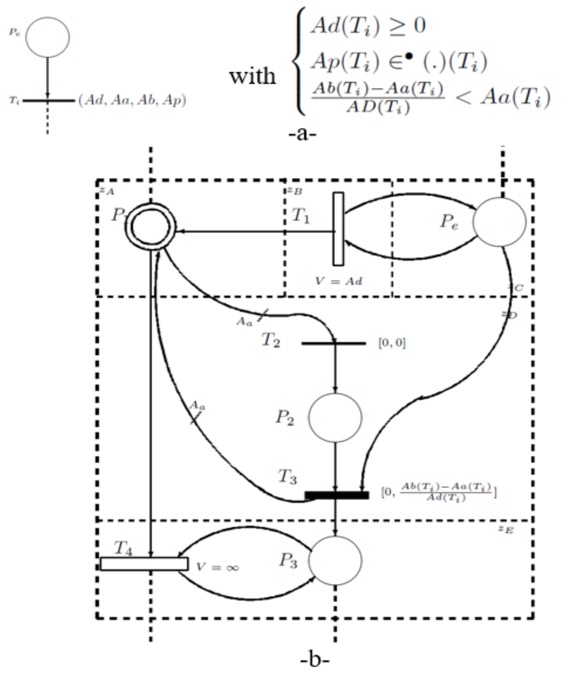


Fig. 6 HTPN of the thermic Processing System

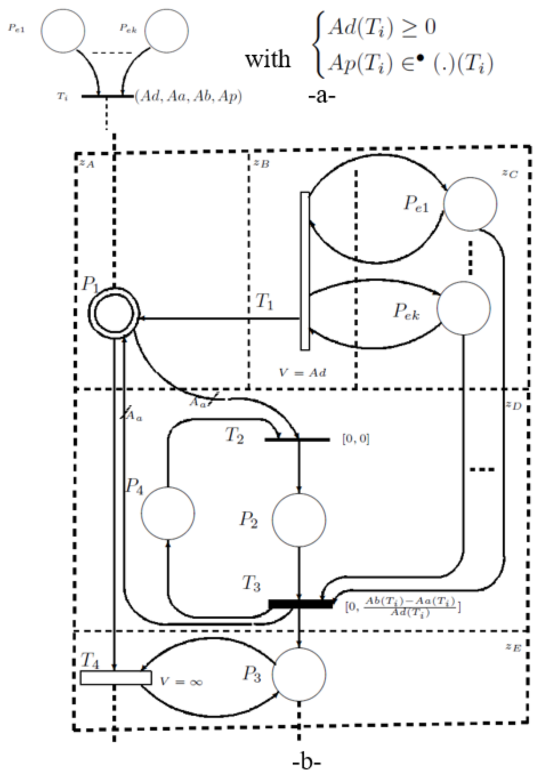
5. Generalization

In this section, we aim to generalize the process of transforming an ETPN into a HTPN. We present the possible pattern representing several scenario in DHS and their presentation in both formalism. Broadly speaking, it is possible to model -with ETPN- many tasks with sequential activities. In this case, a task is considered as completed if the change in marking takes us either to a terminal place (without downstream transition), or to a new initialization. This means that memory effect, which is obtained by transmitting continuous values by tokens, is the link between activities which form a task. When this memory effect comes to an end, either by an initialization, or simply by terminal place(with no output transitions), the task is considered as finished.



$$\text{with}\begin{cases} Ad(T_i) \geq 0 \\ Ap(T_i) \in^{\bullet} (.)(T_i) \\ \frac{Ab(T_i)-Aa(T_i)}{AD(T_i)} < Aa(T_i) \end{cases}$$

Fig. 7 Conversion of ETPN into HTPN- **case 1**



$$\text{with}\begin{cases} Ad(T_i) \geq 0 \\ Ap(T_i) \in^{\bullet} (.)(T_i) \end{cases}$$

Fig. 8 Conversion of ETPN into HTPN- **case 2**

In Figure 8, we present a more general form of linear transition that may have more than one input place. The new added place P4 is used as a capacity representation to limit the multi-firing to one instance exclusively.

Case 1 corresponds to a linear transition with non-negative slope with memory. The transition T3 of the HTPN is a discrete transition that represents the amount of time necessary to fulfil the task of the ETPN transition. The infinite speed of T4 means the transfer of the content of P1 is done instantaneously once P3 is fed with a token by T3 firing.

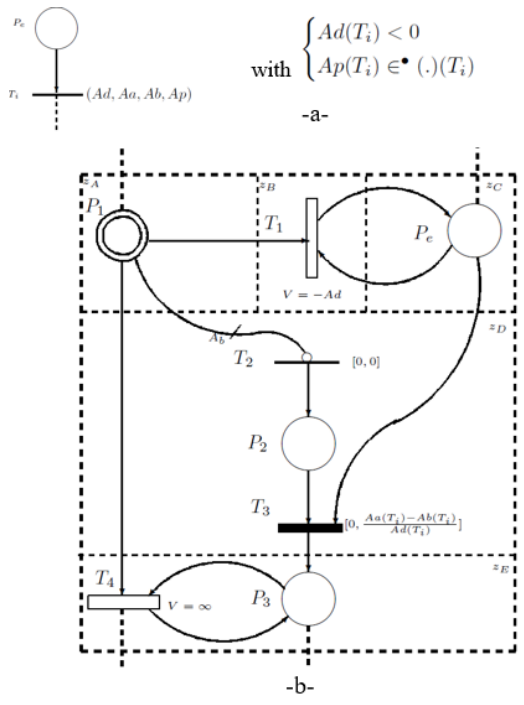


Fig. 9 Conversion of ETPN into HTPN- **case 3**

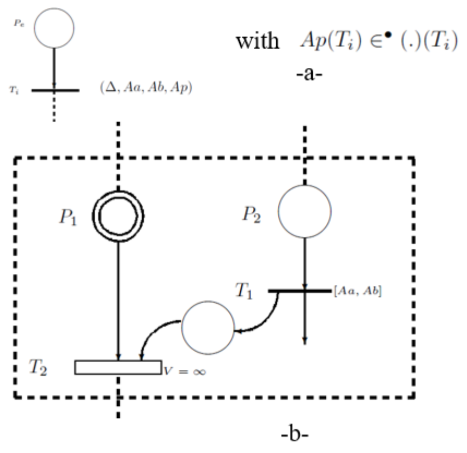


Fig. 10 Conversion of ETPN into HTPN- **case 4**

The case of linear transition with negative slope is considered in Figure 9. We use an Inhibitor arc to deal with the negative rate.

Regarding Time transition as in case 4, the continuous place is used only to conserve the content of the variable and transfer to any eventual further activity. This is crucial to maintain the memory case available.

These block will be joined to represent structurally an equivqlent HTPN of a given ETPN.

6. Conclusion

The objective of this work is to propose a formalism that makes it possible to model a subclass of the hybrid dynamic systems in a simple and concise way based on Petri nets. Although it is possible to model these types of systems using various extensions of hybrid Petri nets, we show that modeling them with the ETPN is much less complex in terms of number of nodes and a simpler task. Our model makes it possible to extend the concept of time to more complex dynamic and maintain at the same time a dominating event aspect. This makes it possible to model the evolution of the continuous parts with constant speeds (temperature, etc). Thus, for certain transition of ETPN, we associate a dynamics with linear equation.

HTPN are more used in literature for modelling various aspect related to hybrid dynamic systems. However, the model becomes too complex for the subclass of systems we consider (DHS event-driven, strong discrete aspect, multi-instances, cumulative continuous variables….). Thus we proposed to model this subclass of systems with more suitable formalism –ETPN- and a systematic fashion to translate it into HTPN.

Acknowledgments

The authors wish to acknowledge the approval and the support of this research study by the grant N°CIT-2017-1-7-F-7076 from the Deanship of the Scientific Research in Northern Border University, Arar, KSA.

References

[1] B. Berthomieu and M. Diaz. Modeling and verification of time dependent systems using time Petri nets. IEEE transactions on software Engeneering, 17(3):259-273, 1991.
[2] C. Ramchandani. Analysis of Asynchronous Concurrent systems by Timed Petri Nets. Ph.d thesis, MIT, Project MAC TR-120, 1974.
[3] Cao, Z. B., et al. "Modelling a complex system based on the hybrid Petri net." Information Science and Electronic Engineering: Proceedings of the 3rd International Conference

of Electronic Engineering and Information Science (ICEEIS 2016), 4-5 January, 2016, Harbin, China. CRC Press, 2016.
[4] Yamen EL TOUATI, Moez YEDDES, Nejib BEN HADJ ALOUANE. "Modeling and control of constant speed Dynamic Hybrid Systems using Extended Time Petri Networks," 24th Chinese Control and Decision Conference (CCDC), 2012, pp.634-641, 23-25 May 2012.
[5] Yamen EL TOUATI, Moez YEDDES, Nejib BEN HADJ ALOUANE. « Time Petri Networks with Memory-Enabled Tokens: An Application to the Modeling of Dynamic Hybrid Systems ». 11th international workshop on Discrete Event systems WODES'12. Guadalajara, Mexico. October 3-5, 2012.
[6] R. Champagnat, H. Pingaud, and R. Valette. Modeling and simulation of a hybrid system through pr/tr pn-dae model. In Proceeding of ADPM'98, pages 131-137, Reims, France, Mars 1998.
[7] R. David and H. Alla. Discrete, Continuous, and Hybrid Petri Nets. Springer, Heidelberg, 2 edition, 2010.
[8] H. Guguen and J. Zaytoon. Abstractions of hybrid systems for verification. In JuanAndrade Cetto, Joaquim Filipe, and Jean-Louis Ferrier, editors, Informatics in Control Automation and Robotics, volume 85 of Lecture Notes in Electrical Engineering, pages 15-28. Springer Berlin Heidelberg, 2011
[9] T. A. Henzinger. The theory of hybrid automata. In LICS '96:Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science, page 278, Washington, DC, USA, 1996. IEEE Computer Society Press. An extended version appeared in Verification of Digital and Hybrid Systems (M.K. Inan, R.P. Kurshan, eds.), NATO ASI Series F: Computer and Systems Sciences, Vol.170, Springer-Verlag, 2000, pp. 265-292.
[10] Lennartson, Bengt, Kristofer Bengtsson, and Oskar Wigström. "Optimization of hybrid Petri nets with shared variables." Automation Science and Engineering (CASE), 2015 IEEE International Conference on. IEEE, 2015.
[11] T. A. Henzinger, Z. Manna, and A. Pnueli. Timed transition systems. In J.W. Bakker, C. Huizing, W.P. Roever, and G. Rozenberg, editors, Real-Time : Theory in Practice, volume 600 of Lecture Notes in Computer Science, pages 226–251. Springer Berlin Heidelberg, 1992.
[12] Franck Cassez and Olivier H. Roux. Structural translation from time petri nets to timed automata. Journal of Software and Systems, 79(10):1456–1468, October 2006
[13] M. Allam and H. Alla. From hybrid Petri nets to hybrid automata. In Journal Européen des Systèmes Automatisés (JESA), volume 32, pages 1165–1185, 1998.

Yamen El Touati received his Engineering, M.S, and PhD degrees In Computer Science from the National School of Computer Science (ENSI), University of Manouba in 2003, 2005 and 2014, respectively. He is an assistant professor at ISAMM, University of Manouba, Tunisia and a permanent research member of the OASIS laboratory at the National School of Engineers of Tunis, University of Tunis El Manar, Tunis, Tunisia. Currently, he is on secondment as assistant professor in computer science at the Department of Computer Science, Faculty of Computing and Information Technology – Northern Border University (NBU) in the Kingdom of Saudi Arabia. His research interests include modelling, diagnosis and control supervision of dynamic hybrid systems and timed systems with various automata based and Petri-net based models. His research also, focuses on security and opacity issues for composed web services.

Mohamed Ayari received the Dipl.-Ing., M.S, and PhD degrees in Telecommunications respectively in 2003, 2004 and 2009 from the National Engineering School of Tunis (ENIT)-Tunisia in collaboration with National Polytechnic Institute of Toulouse-France and Virginia-Tech-USA. He is a teacher in several universities since 2003. His is a permanent research member in 6'COM laboratory at ENIT since 2003 till now. In 2005 he joined RCEM-Inc. at Toulouse-France. Since 2010 he has been a tenure track Assistant-Professor at National Engineering School of Carthage (ENICAR)-Carthage University-Tunisia. Since 2015 he is joined as assistant professor IT Department of Faculty of Computing and Information Technology – Northern Border University (NBU) in the Kingdom of Saudi Arabia. His current research interests are electromagnetic (EM) fields, numerical EM methods, computer-aided design of microwave circuits and antennas. His research interests include also information security and wireless applications.

Saleh Altowaijri is the Dean of the Faculty of Computing and Information Technology, Rafha, Northern Border University. He has obtained his PhD from Swansea University in the area of cloud computing. He has over 7 years of research experience and has published several book chapters, conference and journal papers. He is a reviewer of several international conferences and journals. His research interests include grid and cloud computing, database management systems, data mining, information systems, information technology risk management, and emerging ICT systems in healthcare and transportation sectors.