**Preprint Proc. IFAC Conf.**
**Analysis and Design of Hybrid Systems**
**16-18 June 2003, Saint-Malo, Brittany, France**

# PETRI-NETS AND HYBRID-STATE MARKOV PROCESSES IN A POWER-HIERARCHY OF DEPENDABILITY MODELS

**Mariken H.C. Everdij** [*] **Henk A.P. Blom** [*]

[*] *National Aerospace Laboratory NLR*
*P.O. Box 90502, 1006 BM, Amsterdam, The Netherlands*
*Fax: +31 20 511 3210, Tel: +31 20 511 3522*
*Email: everdij@nlr.nl, blom@nlr.nl*

Abstract: This paper extends the power hierarchy of dependability models developed by Malhotra and Trivedi (1994) and Muppala *et al.* (2000) to include Piecewise Deterministic Markov Processes (PDP) and PDP-related Petri Nets. PDPs are known as the largest class of continuous-time hybrid state Markov processes not involving diffusions. Since Petri Nets have proven to be extremely useful in developing Markov process models of complex practical processes, there is a clear need for a type of Petri Net that can play such role for developing PDP models. This paper defines such Petri Nets and shows their relation to PDPs and other Petri Nets. *Copyright ©, 2003, IFAC*

Keywords: Hierarchies, Markov models, Petri Nets, Hybrid modes, Modelling, Poisson processes, Discrete event systems, Continuous time systems

## 1. INTRODUCTION

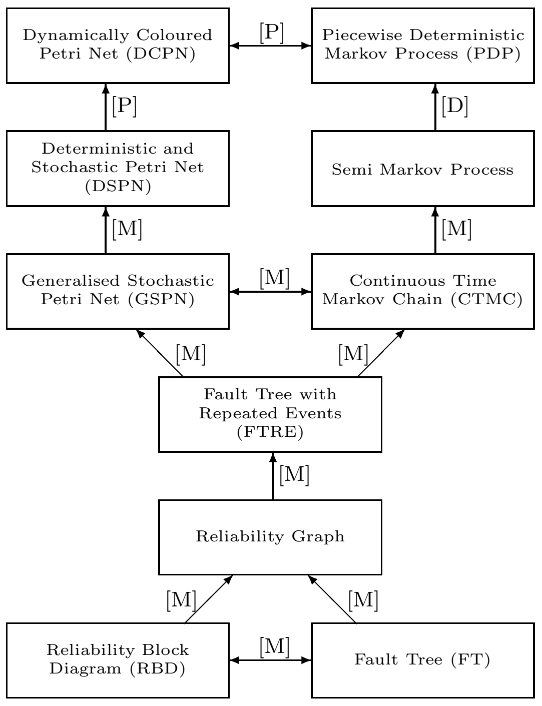
Malhotra and Trivedi (1994) and Muppala *et al.* (2000) developed a hierarchy of various dependability models based on their modelling power. The aim of this paper[1] is to extend this power hierarchy such that it includes Piecewise Deterministic Markov Processes (PDP) and a PDP-related Petri Net.

Davis (1984, 1993) has introduced PDPs as the most general class of continuous-time Markov processes which include both discrete and continuous processes, except diffusion. In his 1984 paper, Davis shows that PDP have more modelling power than Semi Markov Processes.

(Everdij *et al.*, 1997; Everdij and Blom, 2000) have introduced a novel type of Petri Net, named Dynamically Coloured Petri Net (DCPN), which has the same modelling power as PDP. This paper will show this by identifying into-mappings between DCPN and PDP, and will also show that DCPN have more modelling power than Deterministic and Stochastic Petri Nets (DSPN). The combination of these results with those by Muppala *et al.* (2000) leads to Figure 1, in which well known dependability models Reliability Block Diagrams and Fault Trees are at the basis of the hierarchy.

The motivation for this research stems from the following unsolved issue in air traffic: under which conditions is it possible to reduce established criteria for separation between aircraft without sacrificing safety in the form of collision risk. Studying this issue is most urgent for those regions in the world where air traffic is most dense, and consequently where the interplay between aircraft and air traffic management centres is most complex.

[1] This research has been performed with support of the European Commission through the HYBRIDGE project



Dynamically Coloured Petri Net (DCPN)
[P]
Piecewise Deterministic Markov Process (PDP)
[P]
[D]
Deterministic and Stochastic Petri Net (DSPN)
Semi Markov Process
[M]
[M]
Generalised Stochastic Petri Net (GSPN)
[M]
Continuous Time Markov Chain (CTMC)
[M]
[M]
Fault Tree with Repeated Events (FTRE)
[M]
Reliability Graph
[M]
[M]
Reliability Block Diagram (RBD)
[M]
Fault Tree (FT)

Fig. 1. Power hierarchy among various model types. An arrow from a model to another model indicates that the second model has more modelling power than the first model. Arrows labelled by [M] have been established by (Malhotra and Trivedi, 1994) and (Muppala *et al.*, 2000). The arrow labelled by [D] is established by Davis (1984). Arrows labelled by [P] are established in the current paper.

During an earlier study (Bakker and Blom, 1993) a clear relation has been established between collision risk and the evolution of the density of the joint states of two or more flying aircraft. During the subsequent search for Markov process models to characterise the evolution of such densities the class of PDPs was identified as a very useful one (Everdij *et al.*, 1996). At this moment, this type of modelling and evaluation has been accomplished for several air traffic management situations (e.g. Blom *et al.*, 2001). It appeared to be less straightforward to develop an appropriate PDP model for a process as complex as air traffic management. For this reason a Petri Net has been developed (Everdij *et al.*, 1997; Everdij and Blom, 2000) that supports the modelling of PDPs for complex practical problems, similarly as Stochastic Petri Nets support the development of a Markov Chain for discrete valued complex problems. This new Petri Net and its precise relation with PDPs and other models form the subject of this paper.

A PDP (Davis, 1984, 1993) consists of two components: a discrete valued component and a continuous valued component. The discrete valued component models the mode process $\{\theta_t\}$. At discrete times, $\{\theta_t\}$ may switch to another mode value which is selected according to some probabilistic relation. The continuous valued component models the drift process $\{x_t\}$, as a solution of a $\theta_t$-dependent differential equation. At discrete moments in time, $\{x_t\}$ may jump according to some relation, which makes it only piecewise continuous. The PDP state is given by $\xi_t = \text{Col}\{\theta_t, x_t\}$, and is called a *hybrid state*. A switch and/or a jump occurs either when a doubly stochastic Poisson process generates a point or when $\{x_t\}$ hits the boundary of a predefined area. If $\{x_t\}$ also makes a jump at a time when $\{\theta_t\}$ switches, this is said to be a hybrid jump. PDPs are defined such that their sample paths are right-continuous and have left-hand-side limits (càdlàg, from the French 'continu à droite, limites à gauche', see e.g. Protter (1990)).

There are two potential formalisms available that might support the development of a PDP model for a complex multi-agent application: Hybrid Automata and Petri Nets. The first have shown to be useful for application in problems of decidability, formal verification and control synthesis (Alur *et al.*, 1993; Lygeros *et al.*, 1998; Van Schuppen, 1998; Sipser, 1997; Tomlin *et al.*, 1998; Weinberg *et al.*, 1996). Branicky (1995) identified a close relation between PDPs and Hybrid Automata. An important limitation, however, is that Poisson type of events are not covered by Hybrid Automata, which makes them rather restrictive in modelling stochastic effects that occur in practice and are covered by PDPs.

Petri Nets (see David and Alla (1994) for an overview) could provide another important modelling formalism for PDP processes. Several hybrid state Petri Net extensions have been developed in the past. Main classes are:

- Hybrid Petri Net (Le Bail *et al.*, 1991). Some places have a continuous amount of tokens that may be moved to other places by transitions.
- Fluid Stochastic Petri Net (FSPN) (Trivedi and Kulkarni, 1993). Some places have a continuous amount of tokens, the flow rate of which is influenced by the discrete part. The discrete part of the FSPN can be mapped to a continuous-time Markov chain.
- Extended Coloured Petri Net (ECPN) (Yang *et al.*, 1995). The token colours are real-valued vectors that may follow the solution path of a difference equation.
- High-Level Hybrid Petri Net (HLHPN) (Giua and Usai, 1996). Again, the token colours are real-valued vectors that may follow the solution path of a difference equation, but in addition, a token switch between discrete places may generate a jump in the value of the real-valued vector.

- Differential Petri Nets (Demongodin and Koussoulas, 1998). Differential places have a real-valued number of tokens and differential transitions fire with a certain speed that may also be negative.

For none of the above hybrid state Petri Nets it is clear how they relate to PDP. In order to characterise the exact relation to a PDP, a kind of hybrid state Petri Net is needed that makes direct use of the specific PDP structure. The newly developed Dynamically Coloured Petri Net (DCPN) presented in this paper does this. This makes that into-mappings between PDPs and DCPNs exist. An issue that deserves special attention when relating PDPs to Petri Nets is that for a PDP, at each moment in time, there is a unique realisation of the state, while a Petri Net may make a sequence of jumps at a single moment in time. The into-mappings between PDPs and DCPNs referred to in this paper take care of this issue.

The organisation of this paper is as follows. Sections 2 through 4 define Dynamically Coloured Petri Nets and show that DCPN have the same modelling power as PDP. Section 5 gives an example DCPN. Section 7 shows how DCPN have more modelling power than DSPN. Section 8 gives conclusions.

## 2. DCPN ELEMENTS

A Dynamically Coloured Petri Net (Everdij and Blom, 2000) is given by $DCPN = (\mathcal{P}, \mathcal{T}, \mathcal{A}, \mathcal{N}, \mathcal{S}, \mathcal{C}, \mathcal{I}, \mathcal{V}, \mathcal{G}, \mathcal{D}, \mathcal{F})$, where:

- $\mathcal{P}$ is a set of places.
- $\mathcal{T}$ is a set of transitions which consists of a set $\mathcal{T}_G$ of guard transitions, a set $\mathcal{T}_D$ of delay transitions, and a set $\mathcal{T}_I$ of immediate transitions.
- $\mathcal{A}$ is a finite set of arcs, which consists of a set $\mathcal{A}_O$ of ordinary arcs, a set $\mathcal{A}_E$ of enabling arcs, and a set $\mathcal{A}_I$ of inhibitor arcs.
- $\mathcal{N}$ is a node function which maps each arc to an ordered pair of one transition and one place.
- $\mathcal{S}$ is a set of colour types for the tokens occurring in the net (a colour is the value of an object or process in Petri Net terminology).
- $\mathcal{C}$ is a colour function which maps each place to a colour type in S.
- $\mathcal{I}$ is an initial marking which defines the set of tokens initially present, i.e., it specifies in which places they initially reside, and the colours they initially have.
- $\mathcal{V}$ is a set of place specific colour functions which describe what happens to (i.e. defines the rate of change of) the colour of a token while it resides in a specific place. It is determined by a token colour differential equation, which is locally Lipschitz continuous.
- $\mathcal{G}$ is a set of boolean-valued transition guards associating each transition in $\mathcal{T}_G$ with a guard function which is evaluated when the transition has a token in each of its input places. The guard function must evaluate to True before the transition is allowed to fire (i.e. remove and produce tokens). Its evaluation depends on the colours of the input tokens of the transition.
- $\mathcal{D}$ is a set of transition delays associating each transition in $\mathcal{T}_D$ with a delay function which is evaluated when the transition has a token in each of its input places. The delay function determines for how long the transition must wait before it is allowed to fire (i.e. remove and produce tokens). The firing rate depends on the colours of the input tokens of the transition.
- $\mathcal{F}$ is a set of (probabilistic) firing functions describing the quantity and colours of the tokens produced by the transitions at their firing. Its evaluation depends on the colours of the input tokens of the transition.

The set of places $\mathcal{P}$, the set of transitions $\mathcal{T}$, the set of arcs $\mathcal{A}$ and the node function $\mathcal{N}$ define a Petri Net graph. Below, the graphical representation of the elements in $\mathcal{P}$, $\mathcal{T}$ and $\mathcal{A}$ are given. The node function $\mathcal{N}$ describes how these components are connected.

| | | | |
|---|---|---|---|
| Place: | ○ | | |
| Guard transition: | ▭ | Ordinary arc: | ⟶ |
| Delay transition: | ▬ | Enabling arc: | —● |
| Immediate transition: | — | Inhibitor arc: | —○ |

## 3. DCPN EVOLUTION

Tokens and the associated colour values in a DCPN evolve through time quite similar as in a Coloured Stochastic Petri Nets (e.g. Haas, 2002). The main additions are that the colour of a token may evolve according to a differential equation that is governed by the colour function of the specific place where the token resides, and that guard transitions take the evolving colour values into account.

Tokens can be removed from places by transitions that are connected to these places by incoming ordinary arcs. A transition can only remove tokens if two conditions are both satisfied. If this is the case, the transition is said to be *enabled.* The first condition is that the transition must have at least one token per ordinary arc and one token per enabling arc in each of its input places and have no token in the input places to which it is connected by an inhibitor arc. When the first condition holds, the transition is said to be *pre-enabled.* The second condition differs per type of transition. For immediate transitions the second condition is automatically satisfied if the transition is pre-enabled. For guard transitions the second condition is specified by the set of transition guards $\mathcal{G}$ and for delay transitions it is specified by the set of transition delays $\mathcal{D}$.

When these two conditions are satisfied, the transition removes the tokens from the input places by which it is connected through an ordinary arc. It does not remove the tokens from places by which it is connected through an enabling arc. Subsequently, the transition produces a token for some or all of its output places, specified by the firing function $\mathcal{F}$. The colour of a produced token (which must be of the correct type, indicated by what $\mathcal{C}$ defines for the output place), and the place for which it is produced is also specified by the firing function $\mathcal{F}$. The evaluation of $\mathcal{G}$, $\mathcal{D}$ and $\mathcal{F}$ may be dependent on the colours of the input tokens of the corresponding transition.

In order to avoid ambiguity, for a DCPN the following rules apply when two transitions are enabled simultaneously:

$R_0$ The firing of an immediate transition has priority over the firing of a guard or a delay transition.

$R_1$ If one transition becomes enabled by two or more disjoint sets of input tokens at exactly the same time, then it will fire these sets of tokens independently, at the same time.

$R_2$ If one transition becomes enabled by two or more non-disjoint sets of input tokens at exactly the same time, then the set that is fired is selected randomly.

$R_3$ If two or more transitions become enabled at exactly the same moment by disjoint sets of input tokens, then they will fire at the same time.

$R_4$ If two or more transitions become enabled at exactly the same moment by non-disjoint sets of input tokens, then the transition that will fire is selected randomly, with the same probability for each transition.

## 4. INTO-MAPPINGS BETWEEN DCPN AND PDP

An important property of DCPN is that they have the same modelling power as Piecewise Deterministic Markov processes (PDP's). This is proven in (Everdij and Blom, 2000) through making use of a construction of into-mappings between DCPN and PDP, see the two theorems below.

**Theorem 1:**
Each Piecewise Deterministic Markov Process with a finite domain $\mathbf{K}$ can be represented by a process generated by a Dynamically Coloured Petri Net ($\mathcal{P}$, $\mathcal{T}$, $\mathcal{A}$, $\mathcal{N}$, $\mathcal{S}$, $\mathcal{C}$, $\mathcal{I}$, $\mathcal{V}$, $\mathcal{G}$, $\mathcal{D}$, $\mathcal{F}$) satisfying $R_0$ through $R_4$.

**Proof:** See (Everdij and Blom, 2000).

**Theorem 2:**
Each process generated by a Dynamically Coloured Petri Net ($\mathcal{P}$, $\mathcal{T}$, $\mathcal{A}$, $\mathcal{N}$, $\mathcal{S}$, $\mathcal{C}$, $\mathcal{I}$, $\mathcal{V}$, $\mathcal{G}$, $\mathcal{D}$, $\mathcal{F}$) satisfying $R_0$ through $R_4$ can be represented by a Piecewise Deterministic Markov Process if the following conditions are satisfied:

$D_1$ There are no explosions, i.e. the time at which a token colour equals $+\infty$ or $-\infty$ approaches infinity whenever the time until the first guard transition enabling moment approaches infinity.

$D_2$ After a transition firing (or after a sequence of firings that occur at the same time instant) at least one place must contain a different number of tokens, or the colour of at least one token must have jumped

$D_3$ In a finite time interval, each transition is expected to fire a finite number of times.

**Proof:** See (Everdij and Blom, 2000).

## 5. EXAMPLE DCPN

To illustrate the advantages of DCPN when modelling a complex system, consider a very simplified model of the evolution of an aircraft in one sector of airspace.

Assume the deviation of this aircraft from its intended path depends on the operationality of two of its aircraft systems: the engine system, and the navigation system. Each of these aircraft systems can be in one of two modes: *Working* (functioning properly) or *Not working* (operating in some failure mode). Both systems switch between their modes independently and on exponentially distributed times, with rates $\delta_3$ (engine repaired), $\delta_4$ (engine fails), $\delta_5$ (navigation repaired) and $\delta_6$ (navigation fails), respectively. The operationality of these systems has the following effect on the aircraft path: if both systems are *Working*, the rate of change of the position and velocity of the aircraft is given by function $\mathcal{V}_1$ (i.e. if $z_t$ is a vector containing this position and velocity then $\dot{z}_t = \mathcal{V}_1(z_t)$). If either one, or both, of the systems is *Not working*, this rate of change is given by $\mathcal{V}_2$. Initially, the aircraft has a particular position $x_0$ and velocity $v_0$, while both its systems are *Working.* The evaluation of this process may be stopped when the aircraft position crosses the boundary $\partial G$ to a neighbouring airspace sector.

Figure 2 shows a DCPN instantiation for this example, where,

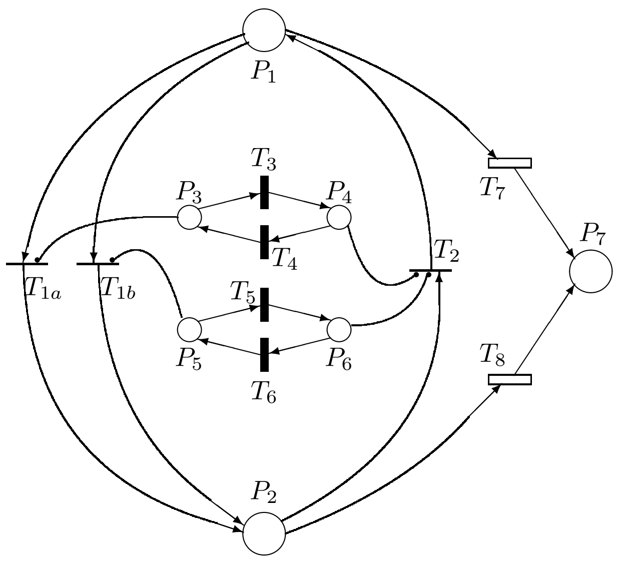


Fig. 2. Example DCPN graph

- $P_1$ denotes aircraft evolution *Nominal*, i.e. evolution is according to $\mathcal{V}_1$.
- $P_2$ denotes aircraft evolution *Non-nominal*, i.e. evolution is according to $\mathcal{V}_2$.
- $P_3$ and $P_4$ denote engine system *Not working* and *Working*, respectively.
- $P_5$ and $P_6$ denote navigation system *Not working* and *Working*, respectively.
- $P_7$ denotes the aircraft having crossed to the neighbouring airspace sector.
- $T_{1a}$ and $T_{1b}$ denote a transition of aircraft evolution from *Nominal* to *Non-nominal*, due to engine system or navigation system *Not working*, respectively.
- $T_2$ denotes a transition of aircraft evolution from *Non-nominal* to *Nominal*, due to engine system and navigation system both *Working* again.
- $T_3$ through $T_6$ denote transitions between *Working* and *Not working* of the engine and navigation systems.
- $T_7$ and $T_8$ denote transitions of the aircraft to the neighbouring airspace sector.

The graph in Figure 2 completely defines DCPN elements $\mathcal{P}$, $\mathcal{T}$, $\mathcal{A}$ and $\mathcal{N}$, where $\mathcal{T}_G = \{T_7, T_8\}$, $\mathcal{T}_D = \{T_3, T_4, T_5, T_6\}$ and $\mathcal{T}_I = \{T_{1a}, T_{1b}, T_2\}$. The other DCPN elements are specified below:

$\mathcal{S}$: One colour type is defined; $\mathcal{S} = \{\mathbb{R}^6\}$.

$\mathcal{C}$: $\mathcal{C}(P_1) = \mathcal{C}(P_2) = \mathcal{C}(P_7) = \mathbb{R}^6$. The colour components model the 3-dimensional position and 3-dimensional velocity of the aircraft. For places $P_3$ through $P_6$, no colour type needs to be defined (or one can define a dummy colour type).

$\mathcal{I}$: Place $P_1$ initially has a token with colour $z_0 = (x_0, v_0)' \in \mathbb{R}^6$. Places $P_4$ and $P_6$ initially each have a token with no colour.

$\mathcal{V}$: The token colour functions for places $P_1$, $P_2$ and $P_7$ are defined by $\mathcal{V}_{P_1} = \mathcal{V}_1$, $\mathcal{V}_{P_2} = \mathcal{V}_2$ and $\mathcal{V}_{P_7} = 0$. For places $P_3$ – $P_6$ the token colour function is not applicable.

$\mathcal{G}$: Transitions $T_7$ and $T_8$ have a guard that is defined by $\partial G_{T_7} = \partial G_{T_8} = \partial G \times \mathbb{R}^3$.

$\mathcal{D}$: The jump rates for transitions $T_3$, $T_4$, $T_5$ and $T_6$ are $\delta_{T_3}(\cdot) = \delta_3$, $\delta_{T_4}(\cdot) = \delta_4$, $\delta_{T_5}(\cdot) = \delta_5$ and $\delta_{T_6}(\cdot) = \delta_6$, respectively.

$\mathcal{F}$: Each transition has a unique output place, to which it fires a token with a colour (if applicable) equal to the colour of the token removed.

## 6. MODELLING POWER OF DCPN VERSUS DSPN

This section shows that DCPN have more modelling power than DSPN (Deterministic and Stochastic Petri Nets), as shown in the power hierarchy as presented by Figure 1, which is based on the one presented in Muppala *et al.* (2000).

The existence of an arrow from DSPN to DCPN can be shown as follows: GSPN (Generalised Stochastic Petri Nets) are generalisations of Stochastic Petri Nets allowing transitions to have either zero firing times (immediate transitions) or exponentially distributed firing times (timed transitions). Immediate transitions which can be simultaneously enabled must have probabilities assigned. For timed transitions, the decision as to which transition fires next is decided by race; the transition with the minimal delay prior to firing will fire next. Firing of immediate transitions has priority over firing of timed transitions. Other extensions include inhibitor arcs.

A DSPN is a GSPN in which the firing delays of timed transitions may be either constant or exponential. Through the equivalence of GSPN and CTMC it can be easily shown that any GSPN can be written as a DCPN: Such DCPN will have constant exponential delay rates and constant colours. The extension to DSPN can also be covered by a DCPN: For each DSPN transition with a constant firing time, create a DCPN transition with a guard function that evaluates to True when the input token colour equals the DSPN transition's constant firing time plus the colour of the input token at the time the transition is pre-enabled. This input token colour has a token colour function equal to $+1$, and an initial colour equal to zero.

## 7. CONCLUSIONS

This paper extended the power hierarchy of dependability models developed by Malhotra and Trivedi (1994) and Muppala *et al.* (2000) to include Piecewise Deterministic Markov Processes

(PDP) and Dynamically Coloured Petri Nets. The paper explained the existence of into-mappings between PDP and DCPN, yielding that they have similar modelling power, and has shown that DCPN have more modelling power than Deterministic and Stochastic Petri Nets.

PDPs are known as the largest class of continuous-time Markov processes not involving diffusions. Dynamically Coloured Petri Nets are defined to make ample use of these PDP properties and have shown to be very useful in developing PDP models for complex practical problems. This usefulness has been explicitly used for accident risk assessment modelling application to Air Traffic Management (e.g. Blom *et al.*, 2001).

## REFERENCES

Alur, R., Courcoubetis, C., Henzinger, T., Ho, P-H. (1993). Hybrid Automata: An algorithmic approach to the specification and verification of hybrid systems, *Hybrid Systems I, Lecture notes in computer science*, pp. 209-229, Springer-Verlag.

Bakker, G.J., Blom, H.A.P. (1993). Air Traffic Collision risk modelling, *Proc. of the 32nd IEEE Conference on Decision and Control*, pp. 1464-1469.

Blom, H.A.P., Bakker, G.J., Blanker, P.J.G., Daams, J., Everdij, M.H.C., Klompstra, M.B. (2001). Accident risk assessment for advanced ATM, In: *Air Transportation Systems Engineering*, AIAA, Eds. G.L. Donohue, A.G. Zellweger, AIAA, pp. 463-480.

Branicky, M.S. (1995). Studies in Hybrid Systems: Modelling, analysis and control, *PhD thesis, M.I.T., Cambridge, MA.*

David, R., Alla, H. (1994). Petri Nets for the modeling of dynamic systems — A survey, *Automatica*, Vol. 30, No. 2, pp. 175-202.

Davis, M.H.A. (1984). Piecewise Deterministic Markov Processes: a general class of non-diffusion stochastic models, *Journal Royal Statistical Soc. (B)*, Vol. 46, pp. 353-388.

Davis, M.H.A. (1993). Markov models and optimization, Chapman & Hall.

Demongodin, I., Koussoulas, N.T. (1998). Differential Petri Nets: Representing continuous systems in a discrete-event world, *IEEE Transactions on Automatic Control*, Vol. 43, No. 4.

Everdij, M.H.C., Klompstra, M.B., Blom, H.A.P. (1996). Final report on safety model, Part II: Development of mathematical techniques for ATM safety analysis, *Report TR 96197 L*, National Aerospace Laboratory NLR, Amsterdam.

Everdij, M.H.C., Blom, H.A.P., Klompstra, M.B. (1997). Dynamically Coloured Petri Nets for Air Traffic Management safety purposes, *Proc. 8th IFAC Symposium on Transportation Systems, Chania, Greece*, pp. 184-189.

Everdij, M.H.C., Blom, H.A.P. (2000). Piecewise Deterministic Markov Processes represented by Dynamically Coloured Petri Nets, *Report TP-2000-428*, National Aerospace Laboratory NLR, Amsterdam.

Giua, A., Usai, E. (1996). High-level Hybrid Petri Nets: a definition, *Proc. 35th Conference on Decision and Control, Kobe, Japan*, pp. 148-150.

Haas, P.J. (2002). Stochastic Petri Nets, Modelling, Stability, Simulation, Springer-Verlag, New York.

Le Bail, J., Alla, H., David, R. (1991). Hybrid Petri Nets, *European Control Conference, Grenoble, France*, pp. 1472-1477.

Lygeros, J., Pappas, G.J., Sastry, S. (1998). An approach to the verification of the Center-TRACON automation system, *Proc. 1st Int. Workshop Hybrid Systems: Computation and Control*, pp. 289-304.

Malhotra, M., Trivedi, K.S. (1994). Power-hierarchy of dependability-model types, *IEEE Transactions on Reliability*, Vol. R-43, No. 3, pp. 493-502.

Muppala, J.K., Fricks, R.M., Trivedi, K.S. (2000). Techniques for system dependability evaluation, *In: Computational probability*, W. Grasman (ed.), pp 445-480, Kluwer Academix Publishers, The Netherlands.

Protter, P. (1990). Stochastic integration and differential equations, a new approach, Springer-Verlag.

Sipser, M. (1997). Introduction to the theory of computation, PWS publishing company, Boston.

Tomlin, C., Lygeros, J., Sastry, S (1998). Synthesising controllers for nonlinear hybrid systems, *Proc. 1st Int. Workshop Hybrid Systems: Computation and Control*, pp. 360-373.

Trivedi, K.S., Kulkarni, V.G. (1993). FSPNs: Fluid Stochastic Petri Nets, *Lecture notes in Computer Science*, Vol. 691, M. Ajmone Marsan (ed.) *Proc. 14th Int. Conference on Applications and theory of Petri Nets*, pp. 24-31, Springer Verlag, Heidelberg.

Van Schuppen, J.H. (1998). A sufficient condition for controllability of a class of hybrid systems, *Proc. 1st Int. Workshop Hybrid Systems: Computation and Control*, pp. 374-383.

Yang, Y.Y., Linkens, D.A., Banks, S.P. (1995). Modelling of hybrid systems based on Extended Coloured Petri Nets, *Hybrid Systems II*, P. Antsaklis *et al.* (eds.), pp. 509-528, Springer.

Weinberg, H.B., Lynch, N., Delisle, N. (1996). Verification of automated vehicle protection systems, *Hybrid Systems III, Verification and control*, R. Alur *et al.* (eds.) pp. 101-113, Springer.