Deterministic and stochastic dependability analysis of industrial systems using Coloured Petri Nets approach

Bruno Pinna, Génia Babykina, Nicolae Brinzei, Jean-François Pétin

Deterministic and stochastic dependability analysis of industrial systems using Coloured Petri Nets approach

B. Pinna, G.Babykina, N. Brînzei, J-F. Pétin
Centre de Recherche en Automatique de Nancy
Université de Lorraine, Vandœuvre-lès-Nancy, France

ABSTRACT: Industrial systems dependability analysis is a two-fold complex task. From one hand, it consists in quantitative reliability, maintainability and availability assessment and involves stochastic modelling of system behaviour. From the other hand, it requires deterministic modelling to capture the control system behaviour and to verify its safety properties. Generally two different models of system behaviour are used to achieve these two tasks, requiring different mathematical models: probabilistic and timed models for stochastic analysis and deterministic non-timed models for qualitative analysis.
The purpose of this work is to use one mathematical model for both dependability tasks. The Coloured Petri Nets tool (CPN), which is a high level Petri Net, is used in this paper. The model allows the stochastic simulation of system behaviour and dependability assessment by means of Monte Carlo simulations. The safety analysis is performed by means of state space analysis and model checking techniques. Main addressed issues are related to abstraction and model transformation in order to adapt the CPN model to the stochastic or deterministic context of the dependability analysis.
The described approach is tested on a case study, which is a part of a nuclear power plant sub-system developed by EDF company (Electricité de France). The considered system is characterised by components redundancy, different distribution laws (not only exponential) of failure and reparation times and control laws aiming to switch between configurations according to functional or dysfunctional purposes. Proposed approach appears to be efficient for evaluation of stochastic dependability indicators (availability, MTTF, MTTR, etc.) as well as for safety analysis (reachability of critical states, dead-locks, proof of control behavioural properties, etc.) of a concurrent controlled redundant system.

1 INTRODUCTION

Industrial systems are subject to high dependability constraints, including safety, reliability and availability. Safety proprieties are generally analysed using deterministic techniques and/or formal verification, such as Model Checking analysis (Clarke, Grumberg, & Peled 1999). Besides, the performance analysis is carried out using probabilistic indicators, such as a probability of a component failure in a fixed time period. Such analyses can be performed by means of Monte Carlo simulation or analytical resolutions. The **I**ntegrated **D**eterministic and **P**robabilistic **D**ependability **A**nalysis (IDPDA) combines within a same study a formal verification of deterministic behaviour of a system, which concerns the safety properties, and a stochastic quantitative assessment of system reliability and availability. This kind of mixed deterministic and probabilistic analysis is particularly relevant in the context of dynamic reliability, where the structure function evolves over time due to the physical parameters, characterising the system's operational environment, and to device ageing. This temporal evolution of the structure function impacts the dysfunctional behaviour of a system and its control architecture reconfiguration. In this paper we focus on a redundant and reconfigurable system, which is composed of a physical part and a control part. The physical part describes system's failures and reparations and is thus a stochastic-timed model. In contrast, the control part governing the overall system's behaviour is an untimed model. The link between these two types of models must been realised in an IDPDA study. In this paper the link is given by the use of the only model tool: the Coloured Petri nets (CPN). This tool allows both a Monte-Carlo simulation for a probabilistic assessment and a model-checking verification for the deterministic analysis.
The paper is organised as follows. The formal definition of CPN is given in Section 2, where it is also presented how a CPN can be used to perform the IDPDA assessment. Section 3 provides the case study descrip-

tion and its probabilistic reliability and availability assessment and safety verification results. Conclusion and future improvements are given in the last Section.

2 CPN MODELLING APPROACH

2.1 *Coloured Petri Nets (CPN)*

The Coloured Petri Nets (CPN) (Jensen & Kristensen (2009), Jensen (1997)) is a graphical language for constructing models of concurrent systems and analysing theirs properties. CPN is a discrete-event modelling language combining the capabilities of Petri nets with the capabilities of a high-level programming language. Petri nets provide the foundation of the graphical notation and the basic primitives for modelling concurrency, communication, and synchronisation. The main difference between a classical Petri Net and a CPN is that the CPN tokens can have different *colours* representing data types (*e.g.* Boolean, integer, string or more complex data structure). The formal definition of CPN is as follows.

A Coloured Petri Net is a 9-uplet $CPN = (P, T, A, \Sigma, V, C, G, E, I)$, where:

1. P is a finite set of **places**.
2. T is a finite set of **transitions**, $P \cap T = \emptyset$.
3. $A \subseteq P \times T \cup T \times P$ is a set of directed **arcs**.
4. Σ is a finite set of non-empty **colour sets**.
5. V is a finite set of **typed variables** such that *Type[*ν*]* $\in \Sigma$ for all variables $\nu \in V$.
6. $C : P \rightarrow \Sigma$ is a **colour set function** that assigns a colour set to each place.
7. $G : T \rightarrow EXPR_V$ is a **guard function** that assigns a guard condition to each transition *t* such that *Type[G(t)]* = Bool, bool standing for Boolean data type.
8. $E : A \rightarrow EXPR_V$ is an **arc expression function** that assigns an arc expression to each arc *a* such that $Type[E(a)] = Type[C(p)]$, where p is the place connected to the arc a.
9. $I : P \rightarrow EXPR_\emptyset$ is an **initialisation function** that assigns an initialisation expression to each place p such that *Type[I(p)]* $= Type[C(p)]$.

An example of CPN is shown in Figure 1. This CPN models a system with six components of two types (two components of type c_1 and four components of type c_2. These components are in the *Working* state until a failure occurs (firing of *Failure* transition). Two types of repairers exist in this system: the r_1 repairer can repair only c_1 component and the r_2 repairer can repair only c_2 component. This association repairer-component is modelled by the complex colour set (r_i, c_i) assigned to *Repairmen on standby* place. The initial marking of this place shows that one repairer r_1 and two repairers r_2 are available. When a component is in the *Fail* state and the corresponding repairer is on standby (this association is guaranteed by the same value of variable x on the output arcs from these states) the transition *Start repair* occurs. At the end of repair operation, the component returns in the *Working* state and the repairer returns in the standby state. The main interest of the use of CPN is the reduced size of the model. By comparison, a classical Petri net must have a number of places and transitions two times larger to model the same system.

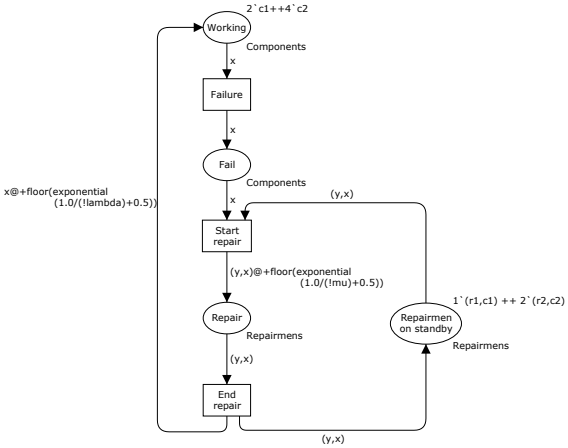


Figure 1: A system with six components and three repairmen.

Additionally, the probabilistic dependability assessment requires the time evolution of the system. For this, the CPN must take into account the *time* aspect. In a timed CPN (Jensen & Kristensen 2009, Jensen 1997), the time is given by a global clock. In addition to their colour, the tokens contain a time value, also called a *time stamp*. When a transition is enabled, it is fired and changes the time stamps of tokens which are deposited in its output places. In these places, the tokens remain *frozen* and can not be used to enable other transitions until the current model time (given by the global clock) is smaller than their time stamps. As soon as the time stamp of the tokens is greater than or equal to the current time model, these tokens can enable other transitions which are instantly fired. In other words, the time stamp describes the *earliest* model time from which a token can be used. In the CPN of Figure 1, this is modelled by the expressions of two input arcs of *Working* and *Repair* places. The expression *x@+floor(exponential(1.0/(!lambda)+0.5))* changes the time stamp of the token deposited in the *Working* place by adding the value obtained by the exponential distribution law with rate *lambda*. This value represents the duration before the occurrence of the next failure, when token sojourns in the *Working* place for this duration. In the same way, the duration of reparation is obtained by the expression *(y,x)@+floor(exponential(1.0/(!mu)+0.5))*, and the token sojourns in the *Repair* place for this dura-

tion.
A major interest of the use of a type of Petri net, which is formally defined, such as CPN, is the verification of CPN properties. This properties verification is supported by the *state space method*. The basic idea underlying the state space method is to compute all reachable states and state changes of the CPN model and to represent them as a directed graph, where nodes represent states and arcs represent occurring events. From a constructed state space, it is possible to answer a large set of verification questions concerning the behaviour of the system, such as absence of deadlocks, a possibility to always be able to reach a given state, and the guaranteed delivery of a given service. These types of CPN properties can represent a specific safety properties of the modelled system, and thus the safety analysis can be realised.
Finally, another concept present in the CPN is the concept of hierarchy which allows a *modular modelling* of complex system. The hierarchy is realised through *substitution transitions* A *substitution transition* (Figure 2) is associated to a more complex CPN (a module), which gives a more precise and detailed description of the activity represented by the substitution transition. The places connected to a substitution transition, called *socket places*, have clearly defined corresponding places, called *port places*, in the related CPN module. They can transmit a given marking from a high level (level of substitution transition) to a low level (level of module) and vice versa. The number of levels in a hierarchical CPN is not limited, because a CPN module can also contain other substitution transitions that are related to lower-level CPN modules.

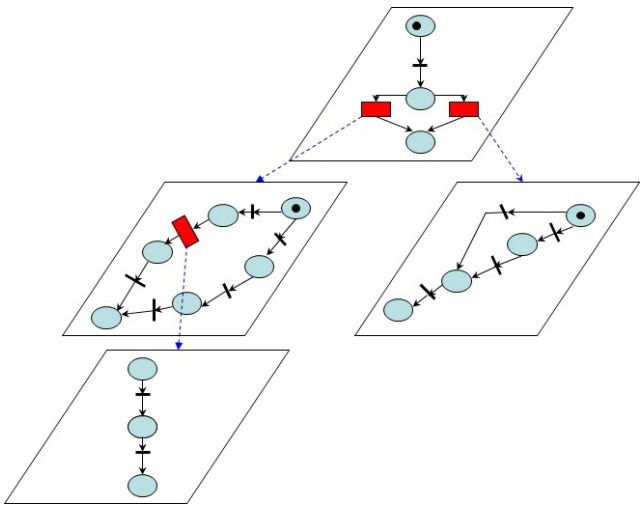


Figure 2: Hierarchy in CPN.

2.2 *CPN for IDPDA*

Dependability assessment discussed in IDPDA approach is summarised by RAMS (Reliability Availability Maintainability and Safety) acronym. This assessment is divided into two types of analysis: a probabilistic (and timed) assessment of reliability, availability and maintainability, and deterministic formal investigation of safety. Firstly, we cover the probabilistic evaluation.

2.2.1 *Probabilistic assessment: Reliability, Availability and Maintainability*

Reliability, Availability and Maintainability are defined as time-dependent probabilistic indicators (in a time interval for reliability and maintainability and at a fixed time instant for availability). Due to the timed and random *nature* of the failure and reparation events, a stochastic representation is needed. Other parameters can also be measured when dealing with the dependability analysis, such that the mean time to first failure (MTTFF), the mean time between failures (MTBF) and other typical time instants. Within a stochastic model failures and reparations can be modelled. Since Monte Carlo simulations are used to provide data for analysis, the present study is not restricted by the Markovian hypothesis (exponential distribution of failure and reparation times).
The *probabilistic indicators* are measured by a *probability*, and they can be determined based on marking invariants (a marking invariant is a subset of Petri net places where the number of tokens is constant). A token, that represents a component, a sub-system or a system, evolves in places which describe its state (waiting, working, failure, ...) and these places make a marking invariant. The probability, representing the indicator that should be estimated, is given by the ratio between the average marking of the place(s) that describe the state(s) characterising the searched indicator and the sum of the average marking of all places belonging to the invariant, i.e. the number of tokens contained by the places subset:

$$P(state_I) = \frac{M^*(state_I)}{\sum\limits_{P_i \in P_{subset_I}} M^*(P_i)} \tag{1}$$

where $state_I$ is the state that characterises the probabilistic indicator *I*, $M^*(state_I)$ is its average marking and P$subset_I$ is the places subset of invariant. For example, for one system (number of tokens is equal to 1), its unavailability can be estimated by the following equation:

$$\bar{A} = P\left(state_{\bar{A}}\right) = \frac{M^*(state_{\bar{A}})}{\sum\limits_{P_i \in P_{subset_{\bar{A}}}} M^*(P_i)} = M^*(state_{\bar{A}})$$

where $state_{\bar{A}}$ represent all the down states of the system.
Consequently, such probabilistic indicators can be estimated by average marking of the corresponding place(s).
The *mean time indicators* are measured by the average value of the sojourn time in the place(s) characterizing the searched indicator.

$$MTI = \sum_{P_i \in P_I} D^*(P_i) \tag{2}$$

where MTI is the Mean Time of Indicator *I*, P_i is a state characterising the indicator *I*, P_I the subset of all these places and D*(P_i) is the average value of sojourn time in the place P_i given by Little's formula:

$$D^*(P_i) = \frac{M^*(P_i)}{\sum\limits_{T_i \in {}^\circ P_i} w(T_j, P_i) F^*(T_j)} \quad (3)$$

The denominator of Eq. (3) gives the sum of the product of average frequency F of input transition T_j of place P_i and of the weight of the input arc from T_j to P_i, $w(T_j, P_i)$. The sum is given for all transitions T_j belonging to subset of input transitions of P_i, noted ${}^\circ P_i$.
For example, the MTTF of system can be estimated by the following equation:

$$MTTF = \sum_{P_i \in P_{operate}} D^*(P_i)$$

where $P_{operate}$ is the subset of operating places.

2.2.2 *Deterministic verification: Safety*

Safety verification employs the deterministic analysis of the system behaviour in order to prove that the system operates according to a given specification and to identify critical events sequences (*e.g.* leading from a given state to an undesired state) and their length. The main method used in the exploration of system operation is the *state space* method. The system state space is the coverage of all of system markings, linked by arcs. To perform a formal safety verification the Model Checking approach defined in Clarke, Grumberg, & Peled (1999) is used. The Model Checking problem definition is provided below:
Let a Kripke Structure K, defining the system structure and an LTL (Linear Temporal Logic) formula φ, defining a property to be verified. The model checking problem consists in verifying that for all path π of the atomic proposition $p \in AP$, AP being a set of atomic propositions, the following relation is satisfied:

$$K \models \varphi \quad (4)$$

If Eq.(4) is satisfied, the result of model checking is a true value, else the result is a counter-example indicating the paths where φ is not satisfied.
The Kripke structure is basically a graph having the reachable states of a system as nodes and state transitions of a system as edges. It also contains labelling of system's states indicating the properties that hold in each state. The *atomic proposition* is a type of sentence which is either true or false and which cannot be broken down into simpler sentences. A type of Kripke structure provided by CPN Tools is the marking graph. Indeed, it is not possible to perform Model Checking directly based on a CPN model. It is however possible to translate a CPN (and a generic Petri Net) into a Kripke structure by different algorithms. The marking graph used as Kripke structure is a graph which contains all the possible system markings. A marking is a vector defining a system state and representing the distribution of the tokens in different places. To obtain a marking graph it is necessary to explore the whole system state space.
For example, let one simple three-state system (for easier understanding we suppose there the system can be off, on and on failure). The corresponding CPN is given in Figure 3 and its marking graph, which can be used as Kripke structure allowing model-checking is presented in Figure 4.

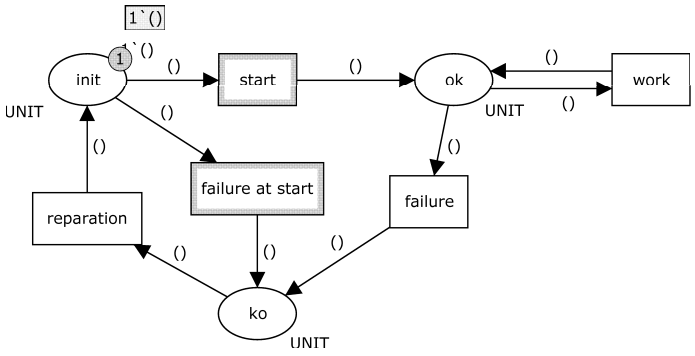


Figure 3: CPN model for one system with three states.

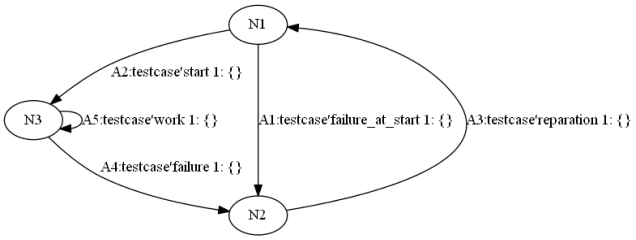


Figure 4: The marking graph of CPN given in Figure 3, used as Kripke structure.

When a stochastic-timed CPN model is considered, the marking graph is infinite because the stamp of tokens depend on time, thus a deterministic version of the system behaviour, having a finite state space, is necessary to be used. The simplest solution in this sense is to cancel the timed and stochastic features of transitions, in order to obtain a deterministic version of timed-stochastic CPN. However, this cancellation of time (determinisation) can imply some sequences of system behaviour, which are impossible in presence of time dependencies. For example, in case when an immediate transition (a transition with zero firing time is called immediate transition) is conflicting with a timed one in a stochastic-timed model, determinisation would imply that both transitions become immediate and are in competition leading to more markings than possible within a timed net. To avoid this, it is necessary to limit the marking graph to the only behaviour enabled in the stochastic timed model, and in dynamically disabling some transition firings. This allows to remove the *impossible* (due to physical or logical consistency) states. To solve this problem, the idea consists in introducing additional places, called

control places which limit the marking graph to the only behaviour enabled in the stochastic timed model, and in dynamically disabling some transition firings. This allows to remove the *impossible* (due to physical or logical consistency) states. This control places can be simply determined using the supervisory control theory (Giua, DiCesare, & Silva 1992, Iordache & Antsaklis 2006). An example of control place is presented in the Figure 8 for the system used below.

3 CASE STUDY

To illustrate the proposed approach a toy example is considered. This example is an extract from a real and more complex case study, developed by EDF (Electricité de France) for the Approdyn project (Aubry et al. 2012). The modelled system is a controlled system composed by one physical subsystem and one control subsystem.
The physical subsystem is composed of two feed-water turbo-pumps (TPA) working in parallel. Each pump is composed of two sub-systems: a turbine part (noted T) and an out-of-turbine part (noted Out-of-T). If one of these sub-system fails, the corresponding feed-water pump fails. The reliability block diagram of this physical system is given in Figure 5.

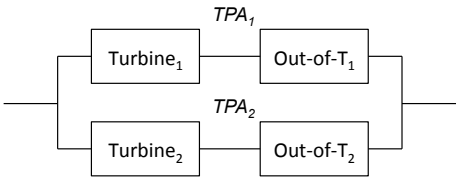


Figure 5: Reliability block diagram of TPAs system.

The data characterising failure and reparation process of each component are presented in Table 1. The failure phenomena are characterised by the exponential law, with the following cumulative distribution function:

$$F(t) = 1 - e^{-\lambda_c t},$$

where: $\lambda_c = 1/MTTF_c$ (c=T or Out-of-T component) is the rate parameter.
For reparation times an Erlang law is considered. Its cumulative distribution function is the following:

$$F(t) = 1 - \sum_{k=0}^{n-1} \frac{1}{k!} e^{-\mu_c t} (\mu_c t)^k,$$

where: $\mu_c = 1/MTTR_c$ (c=T or Out-of-T component) is the rate parameter and $n = 2$ is the order parameter.

Table 1: Components MTTF and MTTR, (*in hour*).

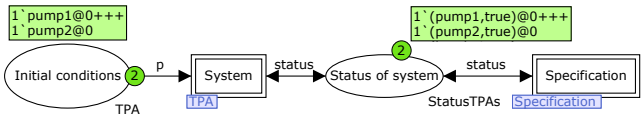
TPA	$MTTF_T$	$MTTF_{HT}$	$MTTR_T$	$MTTR_{HT}$
TPA_1	6780	6854	4	48
TPA_2	2260	6.8×10^6	48	288

The control subsystem describes the specifications of the control used for this system, which is the following. If both pumps are in ON state, the system is working at nominal parameters. If one of the components of a pump fails, the other component of the same pump is stopped and a reparation order is given. The system works in a degraded operating mode. When the repair is finished, the system restarts immediately the repaired pump and the system is working again at its nominal parameters. When both pumps are in failure state, the entire system is failed.

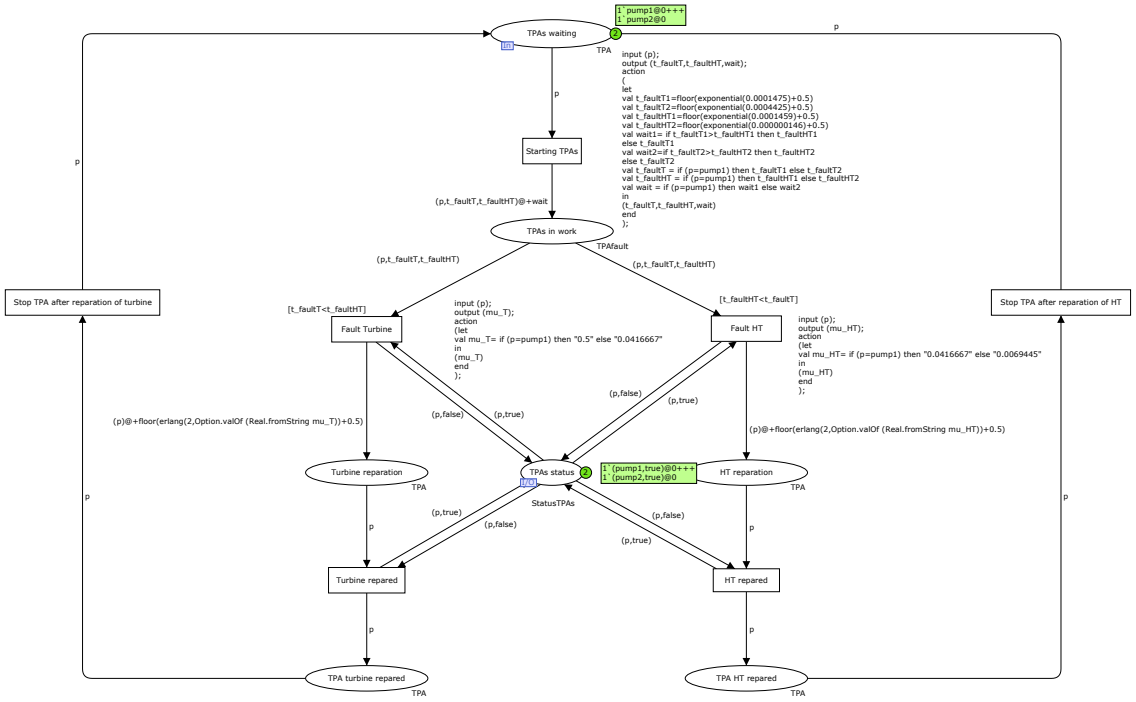
3.1 *CPN modelling of case study*

The CPN model of this system (a physical part and a control part) is implemented using the CPN Tools freeware. The system model has two levels of hierarchy: one for the components themselves and another for coupling process and control subsystems. In Figure 6(a) the hierarchy module is presented. This module links two models: the stochastic model, representing the physical subsystem with failure and reparation probability distribution functions, and the deterministic logical model, which is a control subsystem. The Top model contains the initial conditions and a place shared by the two modules, allowing to check the system status.

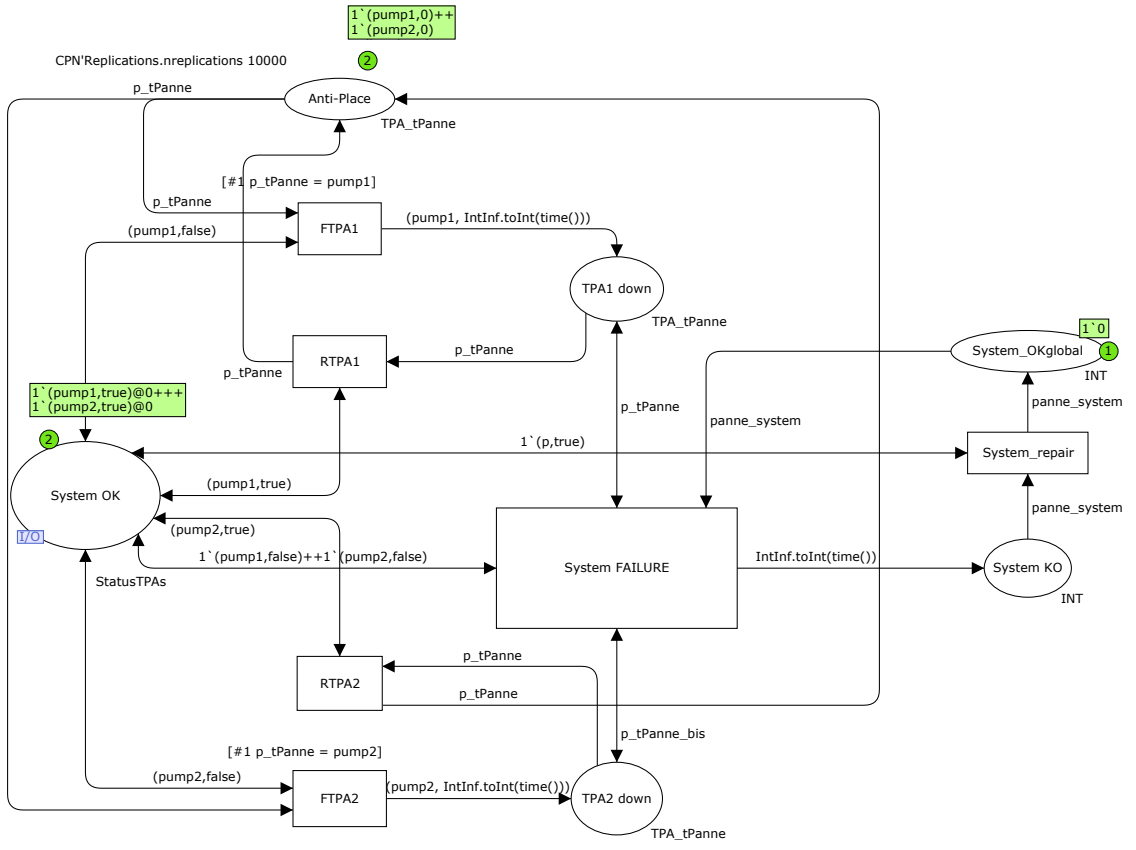
The substitution transition *System* in Figure 6(a) is the physical TPA model represented in Figure 6(b). The TPA model describes a generic TPA with two types of failures, illustrated in the RBD in Figure 5. Two tokens represent the TPA_1 and TPA_2. When the *Starting TPAs* transition is fired, the instants time of the next failures of turbine and out-of-turbine part of each TPA calculated. These instants time are then used to determine the sojourn time of token in the *TPAs in work* place. When the first failure occurs of $pump1$ or $pump2$ the corresponding transition is fired (*Fault Turbine* for the turbine part or *Fault HT* for the out-of-turbine part). The fires of these transitions change the status of the corresponding pump, from on (*true* colour) to off (*false* colour), that is representing by the colour of tokens in the *Status TPA* place. This place is associate to the shared place *Status of system* in the top module. The repair of the failed component is realised and when this reparation is finished (*Turbine repaired* or *HT repaired* transition) the pump returns in the *TPAs waiting* place and the status of the pumps is updated in the *Status TPA* place.
The control model, illustrated in Figure 6(c), corresponds to the substitution transition *Specification* of the Figure 6(a). The control model is used to investigate the behaviour of the physical system, to stop a TPA and to allow a reparation. The *System OK* place is associate to the the shared place *Status of system* in the top module. When the status of one pump changes from On to Off by the physical system, the *FTPA1* or *FTPA2* transition is fired and the pump is in the fail state (*TPA1 down* or *TPA2 down* place). In this state,



(a) Top module.



(b) Physical TPA model.



(c) Logical control model.

Figure 6: CPN models associated to the case study.

Table 2: System performance results

Indicators	Type	TPA1	TPA2	System
MTTFF	Average	3111.7	2234.9	6340.5
(in hour)	$95\%CI$	61.75	43.9	375
MTBF	Average	3144.2	2259.2	6312.8
(in hour)	$95\%CI$	62.4	44.35	373.35
MTTR	Average	26.4	48	16.4
(in hour)	$95\%CI$	0.5239	0.9428	0.9734
Unavailability	Average	0.00694	0.01901	0.00014
(%)	$95\%CI$	0.000113	0.000196	0.000012

if the component are repaired firstly, the status is updated to On by the physical system in the place *System OK*. Now the transition *RTPA1* or *RTPA1* is fired to update the state of the control system. If one pump is failed (*TPA1 down* or *TPA2 down* place) and the second pump fails *1'(pump1,false)++1'(pump2,false)*, the *System failure* transition is fired and the system is breakdown (*System KO* place).

3.2 *Reliability and dependability assessment*

The reliability and dependability assessment is provided by means of Monte Carlo simulations that is the only way to carry out the performance evaluation when the Markovian hypothesis is not verified, due to Erlang laws that modelling the repair processes. The following indicators are assessed: the availability, MTTFF, MTBF and MTTR for the entire system and for each pump (an subsystem composed by two components). This indicators are measured by the indicators defined by Eq. (1) for the availability and by Eq. (2) for the others indicators. In the CPNTools this indicators are indicated by monitor functions, that are the functions developed in ML language used to inspect the CPN during its simulation. In our case four types of monitors are used:

- *MTTFF*, for which the monitors should record only the time of the first entity failure.
- *MTTR*, for which the monitors should record every reparation time.
- *MTBF*, for which the monitors should record every failure time.
- *Unavailability*, for which the monitors should record the duration of state when the entity is broken (the unavailability is the complementary probability of the availability).

These monitors are implemented for the two TPAs and for the whole system.
The statistical results obtained in the Monte Carlo simulation (over 10000 replications) are given in Table 2 where $95\%CI$ represents the half-length of a 95% Confidence Interval.

The data obtained by Monte-Carlo simulation is also used to determine the empirical distributions of different mean times are given in Figure 7. As shown in Figure 7(a), the MTTFF of the controlled system is uniformly distributed, due to the mixture of several exponential and non-exponential laws. The distributions of MTTF and MTBF are quite similar in average and type of law. Indeed, the reparation rate is too small in comparison to the failure rate to modify heavily the probability density function. The empirical distribution of MTTR is similar to a exponential distribution. This is due to the small values of reparation times (the probability of another event during the repair is relatively small) and because the repair starts instantaneously after a failure occurs.

3.3 *Safety verification*

As mentioned in Section 2.2.2, a deterministic model is needed to perform the safety verification. This is realised by adding a control or "referee" place, as shown in Figure 8. This "referee" place and its markings, which are obtained by means of supervision theory, forbid the firing of the stochastic-timed transitions of physical system before the immediate transitions of the control system. The stochastic-timed transition are authorised only one the immediate transitions of control system are fired. Thus the markings graph of deterministic model is limited to the only reachable states of the stochastic model.

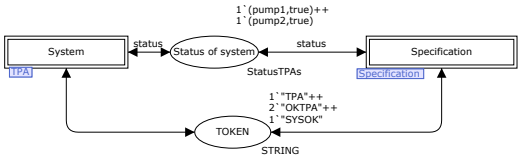
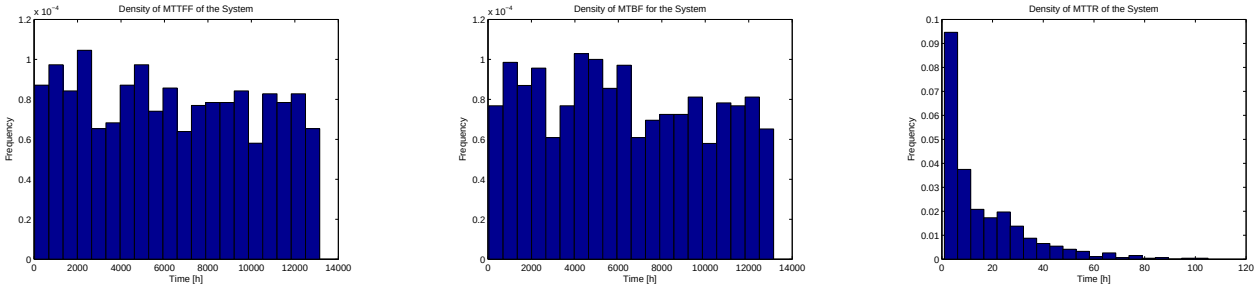


Figure 8: The "referee" place named *TOKEN* in CPN Tools.

After determinisation, the state space can be drawn in CPN Tools and the verification report is produced. The obtained entire state space is composed of 72 markings and 100 arcs.
The Model checking analysis is provided by the ASAP tool (ASCoVeCo State Space Analysis Platform) presented by Kristensen & Westergaard (2007) and based on LTL formulae and ASK-CTL tool based on CTL formulae, both of these tools are implemented in CPN Tools. The following properties have been checked:

- The controller safety property: the controller must immediately restart the TPA_1 after it has been repaired.
- The controller liveness property: from any state, where TPA_1 is not started, it is always possible to restart it.
- Analysis of critical sequences: which is the minimum path from a broken TPA_1 to a state where the entire system fails?

The first two properties can be expressed by LTL or CTL formulae and are verified using ASAP (with



(a) Mean Time To First Failure. (b) Mean Time Between Failure. (c) Mean Time To Reparation.

Figure 7: Empirical distribution of the MTTFF, MTBF and MMTR of the whole controlled system.

a limitation for the counter-example generation) and ASK-CTL:

- The event "TPA_1 is repaired" implies that TPA_1 is restarted in the next step. This can be formalised as: $AG(TPA_1 repaired \Rightarrow AX\ TPA_1 restarted)$.
- For all the paths from a selected node where TPA_1 is not started, the event "TPA_1 is restarted" should be TRUE in a finite number of steps. This can be formalised as: $AG(TPA_1 notstarted \Rightarrow EF\ TPA_1 restarted)$.

The third property is analysed using exploration tool that follows three steps: firstly, finding all the states where TPA_1 is down, secondly, finding all the states where the entire system is down and, finally, finding paths from the TPA_1 down states to the system down states. Six nodes where TPA_1 is down $[9, 30, 26, 28, 25, 11]$ and a unique node where the entire system is down $[34]$ are found. The shortest trace from down TPA_1 to node $[34]$ is obtained from node $[9]$ and has a length of 3.

4 CONCLUSIONS AND PERSPECTIVES

Integrated Deterministic and Probabilistic Dependability Analysis is a current scientific and industrial challenge for dependability community as shown by recently organised workshops (Adolfsson, Holmberg, Karanta, & Kudinov 2012). In this paper, an approach based on Coloured Petri Net (CPN) has been proposed to cover an IDPDA analysis. The proposed approach consists in using a only one CPN model for the both: deterministic verification of safety properties and for the probabilistic assessment of reliability, availability or maintainability indicators. The probabilistic assessment is realised by means of Monte-Carlo simulation, without any modification of the stochastic CPN model. The probabilistic indicators are defined in terms of marking of different places of CPN and can easily be implemented by the monitor functions in the CPNTools. The verification of deterministic safety properties requires firstly a determinisation of the stochastic CPN model. Once this determinisation is realised, the model-checking techniques can be used to verify the safety properties expressed as LTL or CTL formulae. In the future, we will apply this approach to a real and large controlled system. This will be realised in the framework of CONNEXION project (French governmental project that brings together the main actors of the French Nuclear Power Plant).

REFERENCES

Adolfsson, Y., J.-E. Holmberg, I. Karanta, & P. Kudinov (Eds.) (2012, November). *Proceedings of the Integrated Deterministic-Probabilistic Safety Analysis Workshop*, Stockholm, Sweden.

Aubry, J.-F., G. Babykina, N. Brinzei, A. Barros, C. Bérenguer, A. Grall, Y. Langeron, G. Deleuze, et al. (2012). The approdyn project: dynamic reliability approaches to modeling critical systems. *Supervision and Safety of Complex Systems*, 181–222.

Clarke, Jr., E. M., O. Grumberg, & D. A. Peled (1999). *Model checking*. Cambridge, MA, USA: MIT Press.

Giua, A., F. DiCesare, & M. Silva (1992). Generalized mutual exclusion contraints on nets with uncontrollable transitions. In *Systems, Man and Cybernetics, 1992., IEEE International Conference on*, pp. 974–979. IEEE.

Iordache, M. V. & P. J. Antsaklis (2006). *Supervisory control of concurrent systems: a Petri net structural approach*. Springer.

Jensen, K. (1997). *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use (Volume 1)*, Volume 1. Springer Verlag.

Jensen, K. & L. Kristensen (2009). *Coloured Petri nets: modeling and validation of concurrent systems*. Springer-Verlag New York Inc.

Kristensen, L. & M. Westergaard (2007). The ascoveco state space analysis platform: Next generation tool support for state space analysis. In *Eighth Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools*, pp. 1.