Stochastic activity networks for the modeling of repairable systems including diagnosis performance

Samia Maza

8th International Conference of Modeling and Simulation - MOSIM'10 - May 10-12, 2010 - Hammamet - Tunisia
"Evaluation and optimization of innovative production systems of goods and services"

STOCHASTIC ACTIVITY NETWORKS FOR THE MODELING OF REPAIRABLE SYSTEMS INCLUDING DIAGNOSIS PERFORMANCE

S. MAZA

CRAN / Nancy Université
2 avenue de la Forêt de Haye, BP F-54516
54500 Vandoeuvre lès Nancy - France
samia.maza@ensem.inpl-nancy.fr

ABSTRACT: *The productivity and quality requirements have conducted the manufacturing systems to be more and more complex. Indeed, many sub-systems are in interaction such that the process system, the control system and the supervision or diagnosis system. The dependability and availability analysis in those systems is of a major importance, since they impact directly on system's productivity and safety.*
The aim of this paper is to present a modelling framework based on an extension of Petri nets called Stochastic Activity Networks (SANs) that systematically includes the diagnosis performance for the dependability evaluation. The major advantage of such formalism is that it allows the modelling of dynamic systems by modelling all their possible states. And unlike tools such as automata and Markov processes, the modelling can be done simply and in a compact manner. Fault-tolerant systems are considered in here. That is systems including a diagnosis system to allow fault detection, and backup system(s) to allow fault-recovery. The systems under study are considered repairable. Monte-Carlo simulation study is conducted to show the impact of the diagnosis performance and corrective maintenance actions on the system's availability.

KEYWORDS: *Dependability, Diagnosis, Reconfiguration, Stochastic Activity Networks (SAN).*

1 INTRODUCTION

Dependability analysis is an important problem to deal with in modern industrial systems. It is becoming a more and more difficult task due to the rapid technology evolution and increasing complexity of the systems, which often cause the increase of failures occurrence in the system. For manufacturing systems for example, an important failure rate will induce the availability of the system to be low which, at its turn, will reduce the system productivity and throughput.
This is why some sub-systems are added to improve the system's dependability, such that supervision or diagnosis systems, and backup systems. The overall system (i.e. the one including the process, supervision and backup systems) is called fault-tolerant system. The supervision system allows the diagnosis of faults. That is the detection and localization of system's faults. However backup systems allow the reconfiguration of the system when faults occur. Obviously, those systems are not completely reliable. Thus, their performances should be included when dealing with the dependability of the overall system.

Indeed, the fault detection is based on some diagnosis algorithm which defines a procedure to detect a failure based on some parameters, called design parameters. The choice of those parameters impacts the performance of the diagnosis algorithm, and thus the quality of the detection, and thus the actions that will be taken to recover from the faults. So these parameters act directly on the system's performance such that availability. This means that in one hand, the performance of diagnosis systems should be considered explicitly when evaluating system's dependability. On the other hand, the objectives in term of system's availability, safety, etc should be considered to constrain upstream the diagnosis problem by choosing correctly its design parameters.

The diagnosis problem and dependability analysis and design problem should be considered jointly in order to improve the system's performances. There are only few papers in the literature that deal with the interaction between supervision and dependability analysis and design. For example, Weber et al in (Weber et al., 2007) propose a new approach that improves the performance of the decision making in fault diagnosis by taking into account a priori knowledge of the system/components' reliability. Aslund et al in (Aslund et al., 2005 and 2007), consider the safety study of fault tolerant control systems that include diagnosis subsystem. They propose an approach allowing the inclusion of diagnosis performance in the fault-tree analysis in order to evaluate its impact on the overall system's safety. In the same way, Gustafsson et al, propose a method to optimize the detection threshold based on the previously cited approach (Gustafsson et al., 2008). Bonivento et al (Bonivento et al., 2006) propose a procedure for evaluating reliability of diagnostic systems in terms of capability of not generating false alarms and missed diagnosis using statistical tools. Castaneda et al in (Castaneda et al., 2009) address the problem of dynamic reliability estimation of hybrid systems modelled by stochastic hybrid Automata. Some

diagnosis performances amounts are included in their simulation study. Guenab et al in (Guenab et al., 2009) deal with the fault tolerant control systems and their reconfiguration. They propose a control strategy that incorporates both reliability and dynamic performance of the system for control reconfiguration.

In this paper, we propose a method to include systematically the diagnosis performance when evaluating system's dependability parameters like the availability. The systems under study are repairable and fault-tolerant. We propose to use the stochastic activity networks (SANs) to model the supervised fault-tolerant system, and the Monte-Carlo simulation to evaluate the system's availability.

This paper is organized as follows:
In section 2, some tools for the dependability analysis are briefly and formally presented. Section 3 presents the principle of detection in diagnosis systems and expresses the performance amounts that will be used later in the paper. Section 4 deals with the inclusion of diagnosis performance in the dependability analysis. Static and dynamic modeling methods are presented. Section 5 is devoted to the simulation study. The model used in simulation will be presented and explained. Some simulation results will be reported and discussed. We conclude the paper in section 6, where many perspectives of our work are given.

2 DEPENDABILITY ANALYSIS

The dependability of a system is described by various non-functional properties of that system such as, reliability, availability, safety and security (Laprie, 1992). It can be defined as a property that allows its users to have a justified reliance on the service they are delivered. In this paper, we are dealing with the availability factor. It is defined as the probability that a system is operational at the time of interest.

A variety of classical methods for dependability analysis exist (Villemeur, 1988). These include reliability block diagrams (RBDs), Markov processes, failure mode and effect analysis (FMEA), fault-trees analysis (FTA), Petri nets (PNs), Monte- Carlo simulation, etc. Some of these methods are static (such as RBDs and FTA) since they allow the representation of logic relations. Others are dynamic (such as Markov processes, PNs and Monte-Carlo simulation) and allow the modelling of system dynamic states and events. For our needs, only FTA and PNs will be briefly presented.

2.1 Static methods: Fault-tree analysis

A fault-tree arises from the logic diagram that is used to analyze the probabilities associated with various causes and their effects. FTA starts by identifying a problem (catastrophic accident or other undesirable result) and all possible ways that a failure occurs. FTA has been widely used for the safety and reliability assessment. It is equivalent to the structure function which defines for a system the set of all components whose failures lead to the failure of the global system. Both FTA and the structure function express the logical relationship between the event "failure of the system" and the events "failure of component j". Then, the probability of the first event can be easily calculated from the probability of the other events (i.e. the failure of its components). An example is shown and explained in paragraph (4.1).

2.2 Dynamics methods: Petri nets

2.2.1 General definitions

Petri nets are used for the modelling and validation of discrete event systems in which concurrency, communication, and synchronisation play a major role.
They are widely used as a tool for analyzing the safety and dependability of complex systems. First developed by Adam Petri in the 1960s, Petri nets have become a powerful and generic tool for modelling and simulation (Cassandras and Lafortune, 1999).
Formally, a Petri net structure is a directed weighted bipartite graph defined by a 4-tuple N=(*P, T, Pre, Post*), where *T* and *P* are two distinct sets of vertexes (see figure 1). $T=\{t_1, t_2,\ldots t_n\}$ is a set of transitions, and $P=\{p_1, p_2, \ldots p_k\}$ is a set of places. A transition can be seen as an event or an action, and a place represents the condition for the event or the consequence of it. *Pre* and *Post* are two applications, defined from the set of arcs to the set of natural numbers $\mathbb{N}$:
pre (Pi, Tj): $PxT \rightarrow \mathbb{N}$ and *post* (Ti, Pj): $TxP \rightarrow \mathbb{N}$.
They define the valuation of arcs relating places to transitions (Pre) and transitions to places (Post). If *pre* (Pi,Tj)=0 (resp. *post* (Ti,Pj)=0) there is no arc relating Pi to Tj (resp. Ti to Pj). Arcs from P to T are called input arcs and arcs from T to P are called output arcs. A marked Petri net is a 5-tuple $N_m=(P, T, Pre, Post, M_0)$, where M_0 is the initial marking of the Petri net representing the initial state of the system. It is a k-dimension vector, where k is the number of places. A marking vector could be written as: $M=(m_1, m_2, \ldots m_k)$`, where m_i is the marking of the place i and indicates the number of tokens in that place. A place with (or without) a token indicates that the state represented by the place is true (or false).

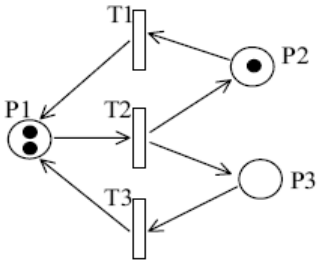


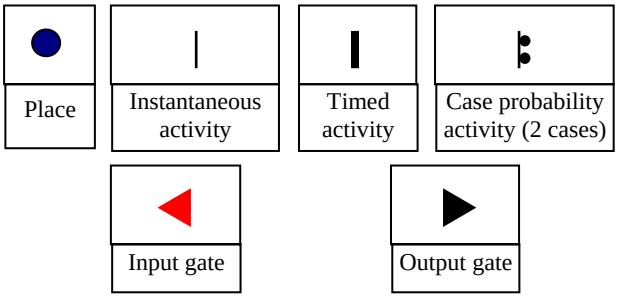
Figure 1: An example of Petri net with three places and transitions (k=3 and n=3)

2.2.2 Stochastic activity networks

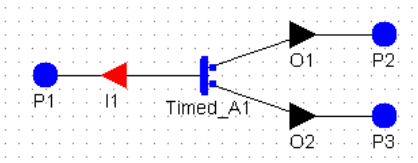
Stochastic activity networks (SANs) are stochastic extensions to Petri nets. They were first introduced by Mogavar et al., (Mogavar et al., 1984). SANs have the modelling power of Petri nets and allow a compact representation of systems. They consist of: *places*, *activities*, *input gates* and *output gates*.

- *Places* can be seen as a state of the modelled system. Each place of a SAN contains a certain number of tokens which represents the marking of the place. Places are represented graphically by circles.

- *Activities* represent actions of the modelled system that could take some specified amount of time to complete. They are similar to transitions in ordinary Petri nets, and are of two types: timed and instantaneous. *Timed activities* have durations that impact the performance of the modelled system such as a packet transmission time. This duration can be stochastic. *Instantaneous activities* represent actions that complete or fire immediately when enabled in the system. Activities are graphically represented by thick lines for the timed ones, and thin lines for the instantaneous ones. Unlike autonomous Petri nets, SANs allow the use of uncertainties associated with the completion of an activity. It is called *Case probabilities*, and is reprensented graphically by small circles on the right side of an activity (see Figure 2). Each case stands for a possible outcome, such as a routing choice in a network, or a failure mode in a faulty system. So each activity in the SAN can have a probability distribution associated with its cases. Moreover, this distribution can depend on the marking of the network at the moment of completion of an activity. This shows how SAN could be a high level modelling formalism.



(a) SAN's elements



(b) An example of SAN model

Figure 2: Presentation of stochastic activity networks

- *Input gates* are used to control the "enabling" of activities. An activity is enabled when the conditions, called predicates, of all input gates connected the activity are true. They are graphically represented by triangles, with the flat side inside connected to the activity via its input arc (figure 2).

- *Output gates* are used to change the state of the system when an activity "completes" by defining the marking change that will occur. They are graphically represented by triangles, with the flat side inside connected to the activity via its output arc (Figure 2).

3 DIAGNOSIS PERFORMANCE

The diagnosis system is a key component in fault-tolerant control system. Indeed, it allows the detection of abnormal functioning of components which is in charge of. One common way in performing diagnosis is the use of a set of tests r_i, called residuals, and to compare them to a threshold J_i. These residuals are fault indicators and are defined as the difference between the measured values of some system's variables and the expected ones, estimated from the system's fault-free model (Figure 3).

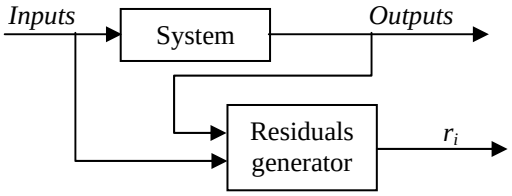


Figure 3: Example illustrating the residuals generator

Several methods to perform residual generation and fault detection are proposed in literature (Weber et al., 2007).
A residual r_i is designed such that r_i is small if the system to be diagnosed is okay (OK) and large otherwise. This test quantity is compared to a predefined threshold J_i and if $r_i > J_i$ then the test is said to alarm and the process to be diagnosed is said to be not okay and will be denoted "KO". Otherwise the system will be considered OK.
The residuals are always corrupted by noise, which affects the decision making. The performance of such tests and the efficiency of detection are related to the probability of two events: *false alarm* (FA_i) and *missed detection* (MD_i).
In statistical theory, the hypothesis "the component *i* is OK" is called the *null hypothesis* of a test and is denoted H_i^0. Its complementary hypothesis, i.e. "the component *i* is KO" is denoted H_i^1. The events: good detection (D), false alarm (FA) and missed detection are related to the residuals value and whether the null hypothesis is true or not as depicted in table 1:

	H_i^0 is true	H_i^1 is true
H_i^0 is accepted : $r_i \le J_i$	Ok	MD_i
H_i^0 is rejected : $r_i > J_i$	FA_i	D_i

Table 1: Definition of FA and MD events

The probability of the events FA_i and MD_i is:

$$P(FA_i) = P(r_i > J_i \mid H_i^0 \; true)$$

$$P(MD_i) = P(r_i \leq J_i \mid H_i^0 \; false)$$

A typical example of the evolution of FA and MD probabilities according to the threshold is given in Figure 4.

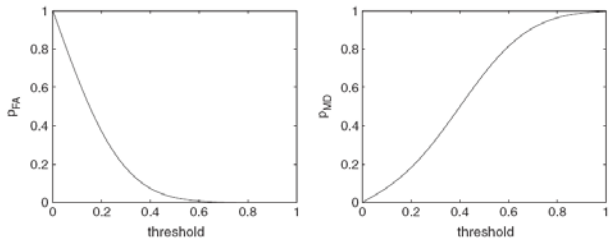


Figure 4: The probabilities of FA and MD as functions of the threshold size

The choice of threshold adjusts the compromise between a small FA probability and a small MD probability.

4 DEPENDABILITY OF FAULT TOLERANT SYSTEMS INCLUDING DIAGNOSIS PERFORMANCE

From the preceding section, it appears clearly that the diagnosis performances should be considered when evaluating the system's dependability parameters since decisions are taken according to diagnosis resulting events (reconfiguration when D and FA, corrective maintenance actions, etc.)

4.1 Static modeling

In (Aslund et al., 2007), the authors propose a method to include the diagnosis performances when evaluating the safety of a fault-tolerant system. They propose the use of fault-tree analysis, which is a systematic way to investigate credible causes for an undesired event in a system to happen. As said before, a fault-tree presents the logical relationships between the undesired event and the basic events leading to it (Figure 5).

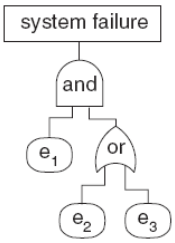


Figure 5: An example of a fault-tree of a three components system

In the example of Figure 5, the system fails if its component C1 and one of the components C2 or C3 fail together. The probability of the top event (system failure) to occur can be computed if the probabilities of the basic events are known. Here, events e_i (i=1,3) represent the failure of component C_i.

In order to improve system's reliability by decreasing the probability of system's failure, diagnostic and backup systems are always added. Hence, the system is OK if the original one is OK, or if the backup system is switched on and is OK. The switching depends on the diagnosis results. Hence, the probabilities of D, FA and MD, the same as the failure probability of the backup system could be added into the fault-tree of the overall system as depicted in the example of Figure 6. In this example, the component under diagnosis study is C_2 and the backup system is C_4. When an alarm A occurs, the system switches from the original one to C_4.

An alarm will occur if C_2 fails and the diagnosis will succeed in detecting it (i.e. we have $(e_2 \wedge \overline{MD})$), or C_2 is okay and the diagnosis algorithm detects a failure (i.e. we have $(\overline{e_2} \wedge FA)$).

The global system will fail if the original one fails and no alarm is generated, or when an alarm is generated, the backup system is switched on but it is out of service (Figure 6).

The probability of the top event depends now on the diagnosis performances. When evaluating this probability, the repeated events in the tree should be considered carefully.

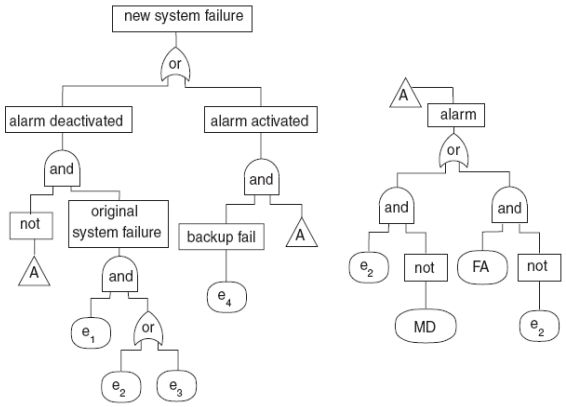


Figure 6: The fault-tree of the preceding example including diagnosis performances

4.2 Dynamic modelling

The major drawback of the preceding method is that it is suitable only for un-repairable systems and uses a static model. Also, it is only suitable for systems whose backups are in active or hot redundancy, i.e. they operate simultaneously with the original ones from time zero. Static fault-trees are not suitable to model components that are in cold standby redundancy since they are sequentially used in the system at failure times (Kuo et al, 2001). Sequential or dynamic models should then be used. In what follows, we propose a dynamic model based on stochastic activity networks, to model the

system's behavior including the diagnosis and backup systems. The SAN have the advantage of being able to model dynamic behaviors, various redundancy arrangements and maintenance actions when the considered systems are repairable.
The proposed method will be explained on the example presented previously in Figures 5 and 6.
Basically, a Petri net or a SAN can be built based on the system's fault-tree. The following guidelines allow the systematic construction of a SAN model based on the logical relationship between the failure of a system and the failure of its components:
Step 1: each component Cj can be represented with a set of two places: {Cj_OK, Cj_KO}, where a token on place "Cj_OK" (resp. "Cj_KO") means that component Cj is OK (resp. not okay or KO). These two places are related to each other by a timed transition or activity called "Panne j" representing the failure of component j. "Panne j" is represented in Figure 7 with a bold line. The duration could be stochastic (according to an exponential distribution function for example).

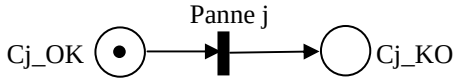


Figure 7: the modeling of a physical component

The marking of those places is binary (i.e. one token at most) and are mutually exclusive since a component is in one state or another (i.e. okay or failed).
Step 2: when the failure of two components or more is necessary to conduct the system S to fail, which could be represented by an AND logic gate, an activity with several input places will be used as depicted in Figure 8.

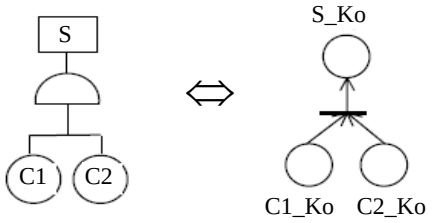


Figure 8: Petri net modeling the logic operator AND

Step 3: when the failure of one component in a set of components is sufficient for the system to fail, which could be represented by an OR logic gate, a place modeling the failure of a system with many input transitions is used as shown in Figure 9.

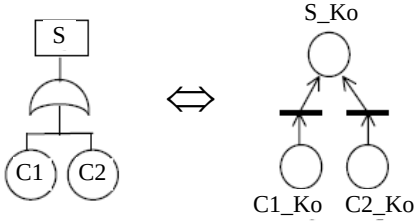


Figure 9: Petri net modeling the logic operator OR

Step 4: the diagnostic system is considered as a system which will deliver three possible events: D for good detection of faults, FA for false alarm and MD for missed detection. It could be modeled by a place, called "ALGO", having an initial marking of one token, and three output transitions. Each transition will be given some firing rate based on the probabilities of D, FA and MD events. Only one transition should be enabled and can fire at a time since all these transitions are in a conflict. The autonomous Petri nets and their corresponding software tools cannot handle conflicts but fortunately the SAN does. In fact, when a structural conflict is modeled in a Petri net, this latter must be assorted with a policy to handle this conflict. However, with stochastic Petri nets, such policy is not necessary since the conflict is cleared up thanks to the stochastic aspect. Indeed, in stochastic Petri nets, the events that are associated with the conflicting transitions form a complete system of events, i.e. they are mutually exclusive (*Cf.* law of total probability). Thus, when enabled only one of them will be fired. With SANs, the three previous transitions associated to the mutual exclusive events D, FA and MD, can be replaced by one activity having several cases and thus, several output arcs (see section 2). Each output arc will be related to an output place representing the fact that: a default is well detected (place *Detect*), or missed (place *MD*) or detected however it doesn't exist (place *Alarm*). Each case is given a probability to be chosen to fire (and the corresponding output arc to be selected) such that the sum of all cases probabilities is 1 (see Figure 10).

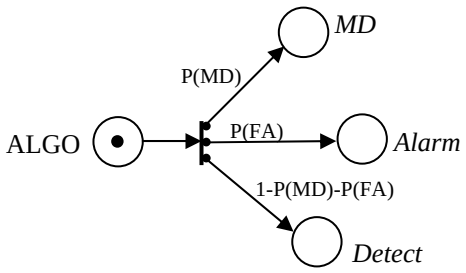


Figure 10: The modeling of the diagnosis system and its performance

Each of the places {MD, Alarm, Detect} are related to some activities of some sub-models to contribute to their enabling.
Step 5: the backup system BS is also modeled by a set of two places {BS_OK, BS_KO} modeling the fact that it could be okay or failed, as described in step 1. The only difference with the components of step 1, is that the initial marking of the preceding places depends on the redundancy policy of the backup system.
If the backup system is used in an active redundancy with the original system, then the place BS_OK will have an initial marking of one token (see step 1). Otherwise, if passive redundancy is used, the initial marking of the both places is zero. A token will be added to BS_OK only when the diagnosis system detect a fault

whether the detection is good or not (i.e. the place *Detect* or *Alarm* is marked). Then, the marking of "BS_OK" could change to zero if the backup system fails, or if some corrective maintenance actions are undertaken. Indeed, if the component which is considered by the diagnosis system is repaired after fault detection, then the backup system could be switched off.
These last remarks, define in some way how the place BS_OK should be related to some output activities other than the failure transition "Panne BS". This situation is illustrated in the simulation model of the next section.
Step 6: The models of components, including the diagnosis system, are combined with each other according to steps 2 and 3, in order to form the overall system.

Our procedure is a generic one and can be used to derive a a SAN model of any fault tolerant system including a diagnosis subsystem. This is possible for example using FMEA method to describe the way that the elementary events like components failures propagate through the system to give a rise to the undesired event: the system's failure. However, the dynamic aspect of these events can be modeled with Petri nets thanks to the markings and the temporal dimension.

5 SIMULATION STUDY

To drive our simulation study, we used Möbius™ software tool developed at university of Illinois for modeling complex systems behavior. It is a successor of *UltraSAN* tool and was originally developed for studying the reliability, availability and performance of computer and network systems. After that, its use has expanded rapidly. The Möbius tool is an environment that supports multiple modeling formalisms such as fault-trees and stochastic activity networks. All our models were developed using Möbius tool.

5.1 Description of the SAN model

The system considered in our simulation study is the one used in (Aslund et al., 2007) and presented in section 3.
We developed a stochastic activity network associated to this system according to our algorithm explained in the paragraph (3.2). Each couple of places (Ci, CiKo) is associated to component Ci to describe its two states: okay and failed (i=1, 4). In this example, C2 is the component supervised by the diagnosis system. The diagnosis system is modeled here by 4-tuple of places (Diagnostic, MD_C2, Alarme, Detect) constructed as previously explained (§3.2). the performances of the diagnosis algorithm (i.e. P(MD), P(FA) and P(D)) are defined in the Case probabilities of the instantaneous activity called "algo". The marking of the place "Detect" models the fact the C2 fails and the failure is well detected by the diagnosis system. Marking the place "Alarme" implies that a failure on C2 was detected however it is okay. Place "MD_C2" marked implies that C2 is failed and it wasn't detected by the diagnosis system. But the failure of C2, even if it's not detected, will contribute to the failure of the original system if C1 fails too. This is why the place "MD_C2" is related to the activity named "Panne12". This activity has also the place "C1Ko" as an input place. The place "Sys_Orig" models the failure of the original system as shown in the fault-tree of Figure 5. It will be marked when "Panne12" or "Panne13" completes. The notation "Panneij" means that "Ci" and "Cj" are both failed.
When an alarm is produced by the diagnosis system, whether the system is okay or not, will conduct the original system to switch into its backup system C4. In that case, there are two possibilities: C4 is in an active redundancy with the original system, or in passive redundancy. Also, when an alarm occurs and the backup system is switched-on, corrective maintenance actions could be undertaken on the component C2 considered as failed. So there are 4 possible solutions according to whether C2 is maintained or not, and C4 is in active or passive redundancy. We have derived 4 different SAN models to describe each solution. The model of Figure 11 is the one where C4 is in passive redundancy and C2 is repaired. This is modeled by the place "maint_C2" and there are two ways for a token to arrive there: by the firing of the activities "maintenir1" if good detection, or "maintenir2" if false alarm.
Since C4 is considered in passive redundancy, no one of its associated places "C4" and "C4Ko" is initially marked. Place "C4" will be marked to model the switching to the backup system when an alarm is produced, that is by firing "maintenir1" or "maintenir2". Therefore, "C4" is also an output place for those two activities. Place "C4" has two output activities: "Panne4" modeling the failure of C4 and "maintC2" modeling the end of the reparation procedure on C2. If "maintC2" fires before "Panne4", then the token in "C4" will be removed to model the switching-off of component C4. Otherwise, this token is removed from "C4" and is added to "C4Ko" modeling the failure of C4. This will lead obviously to the failure of the system (firing of "Panne24"). Notice that there are two input gates "I1" and "I2" on arcs relating places "C1" and "C3" to their output activities "Panne1" and "Panne3". These input gates allow the enabling of their associated activities only when the backup system is switched off (i.e. the currently running system is the original one). The predicate (i.e. the logical condition) defined in those input gates is that the marking of "C4" and "C4Ko" is null. Since we are interested in repairable systems, a place modeling the maintenance actions on the global system called "Panne" is defined in the SAN model (see Figure 11). It has a timed output activity called "Reparation" which models actually the maintenance actions. This activity is related to an output gate called "alimentation_en_jetons" which defines the way the tokens are added to all the output places. When place "Panne" is marked, it means that the system or its backup is out of service. Then, maintenance actions are undertaken to repair the overall system, by repairing all its failed components.

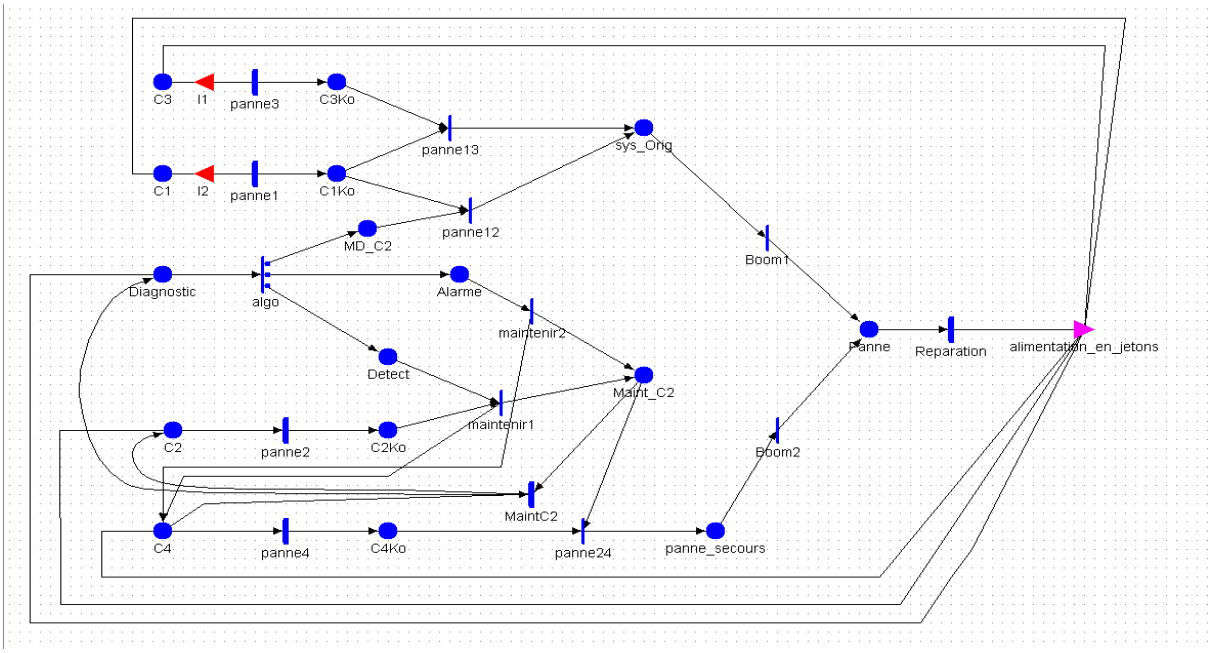


Figure 11: The SAN model of the fault tolerant system of [Aslund] including maintenance actions

To model that, tokens will be added only to places "Ci" where the couple (Ci, CiKo) is not marked (i=1, 4). This is the output condition defined in the preceding output gate.
Notice that activities "Pannej" (j=1, 4), "maintC2" and "reparation" are all timed. We choose exponential distribution for the delays of "Pannej" of a constant rate λ_j, and a uniform distribution for "maintC2" and "reparation".

5.2 Simulation results

To evaluate the impact of diagnosis performance on some performance amounts of the system such as its availability, we conducted a Monte Carlo simulation on the SAN model described before and another model where C2 is not maintained if an alarm is generated. This last one is not presented here. Only its simulation results will be given.
Planning Monte Carlo simulations with Möbius is easy. There is a solving interface which allows the definition of the simulation parameters such that the number of histories to be simulated, the stopping criterion…
The stopping criterion that we used is an upper bound on the discrepancy between the results of the simulations (in our study, it is fixed to 10^{-3}). We also chose to simulate at least 50000 histories and at most 800000 histories. This means that the simulator will stop if one of the two preceding conditions is fulfilled. The time duration of each history T_h is 20000 time units.
Table 2 shows the values of the timed-activities distribution function parameters used in our simulation study.

Exponential distribution				Uniform distribution			
λ_1	λ_2	λ_3	λ_4	α_1	β_1	α_2	β_2
0.0009	0.009	0.0005	0.004	5	20	10	50

Table 2: Distribution functions of the timed activities

In table 2, α_1 and β_1 (resp. α_2 and β_2) denote the lower and upper bound of the uniform distribution associated with the timed activity "maintC2" (resp. "reparation").

Many simulations were conducted with different values of the probabilities FA, MD and D (good detection). For that, we fixed the probability of detecting successfully an alarm (event D) to 80% and we varied the values of P(FA) and P(MD). We also considered the case where the hypothesis that the failure detection is done with certainty (i.e. P(D)=1). Even if it's false, this hypothesis is still widely used in the literature.
The simulation results are reported in table 3.

The statistics collected in table 3 are:
- The average of the time occupation T_{panne} of place "Panne" in the SAN model, calculated over 50000 histories. It is reported in each cellule of the table in black thin line.
- The system's mean availability A, calculated according to the relation: $A = (1 - \frac{T_{panne}}{T_h}) \times 100$, where T_h is the simulation duration (equal to 20000). The system availability is reported in each cellule of the table in red bold line.

P(FA)	P(MD)	P(D)	C4 in passive redundancy & No maintenance on C2	C4 in passive redundancy & maintenance on C2
0%	0%	100 %	1627.90006	498.80672
			91.860 %	**97.500 %**
1%	19%	80 %	1874.97116	712.11297
			90.625 %	**96.439 %**
10%	10%	80 %	2040.36759	928.01575
			89.798 %	**95.359 %**
15%	5%	80 %	2135.33026	1404.29971
			89.323 %	**92.978 %**
19%	1%	80 %	2204.40272	2092.47972
			88.977 %	**89.537 %**

Table 3: The results of Monte-Carlo simulation for different diagnosis performance values

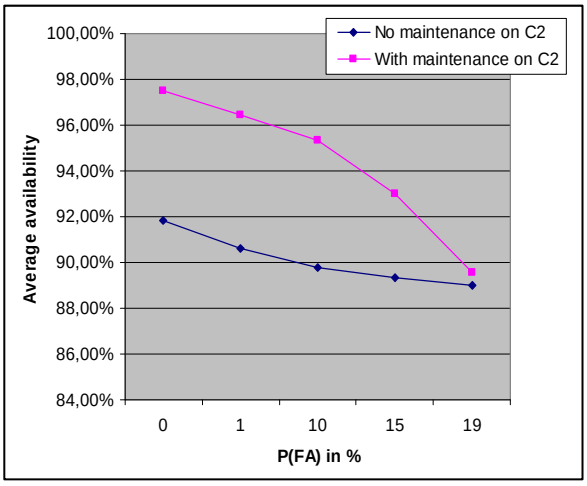


Figure 12: Availability evolution according to the false alarm rate

From table 3 and Figure 12, many remarks can be done:
First, we can clearly see that if the probability of good detection is set to a fix value (here 80%), when the probability of false alarm increases, the system's availability decreases since the time occupation of place "Panne" increases. This can be seen for both models (i.e. with and without maintenance on C2). This can be explained by the fact that a great rate of false alarms will lead the system to switch a lot to its backup system. This makes the system more vulnerable since if the backup system fails, the system does too, while the original system is switched-off but is actually okay. Notice also that the backup system is less reliable than the original system (see table 2 and the fault-tree of Figure 5).

Second, if we compare the fourth and fifth columns to each other, it can be seen that the availability of the second model is greater to the one of the first model. This is due to the fact that each time an alarm occurs, the backup system is turned-on and a corrective maintenance action on C2 is rapidly undertaken. And when C2 is repaired, the system switches from the backup system to the original one. This policy increases the system availability compared to the one of not maintaining C2. Notice however that when the false alarm rate is important (19%), the two models availabilities are quite close. In fact, when the false alarm rate is important, maintaining C2 will not improve the system's availability greatly in comparison to the first model. Indeed, the system will already switch to the backup one and much time will be spent trying to repair C2 even if it's okay.
Third, we can see clearly that the generally used hypothesis that considers that failure detection is done with certainty is too optimistic provided that the system availability is greater than in any other cases and this, for both models.
In this paper, we are not interested in the cost of maintenance, but it could be integrated in the model and may offer a good performance indicator.
The simulation run time varies from few seconds for the SAN model with no maintenance actions on C2 (model 1) to few minutes for one where C2 is maintained (model 2). For example, when P(FA)=P(MD)=10%, the run time is of 71.148 seconds for the model 1 and 171.486 seconds for model 2. The model 2 takes more time in execution since some transitions are fired much more times than the ones in model 1 and thus, these models don't behave similarly. For example, the transition "Maintenir1" is fired in average 22.48 times in model 1 and 120.36 times in model 2 that is at least five times more. The same thing can be said for transition "Maintenir2". This is due to the fact that C2 is maintained each time an alarm is generated and the more it is repaired, the more it is prone to fail again. These simulations were done on an *Intel® core™ Duo* CPU with a clock speed of 2.26 Ghz.

6 CONCLUSION

The aim of this paper was to propose a modelling framework based on stochastic activity networks (SAN) to model a fault-tolerant system in order to evaluate its dependability parameters. We showed how the performances of diagnosis system could be taken into account when constructing a dynamic model for a fault-tolerant system. We proposed a method for model construction which systematically includes diagnosis performances when the system's structure function is known. This implies of course that a dependability analysis should be done upstream.
We presented also our simulation model for a fault-tolerant system, where two policies were applied: maintenance and no maintenance on the supervised component. Monte-Carlo simulations were conducted in

both models to evaluate the average of the system's time duration failure and the system's mean availability. Some simulation results were reported in the paper and show the impact of each policy on the overall system's availability. As predicted, corrective maintenance actions on the supervised component improve the availability of the system.

The simulation results show also that when the probability of false alarm increases, the availability of the system decreases, and thus for both models. This is due to the fact that detecting a fault falsely will conduct the system to switch to its backup system which makes the system less reliable. It also means that it is better to miss the detection of some faults than to detect them wrongly.

There are many perspectives to this work that we intend to work on. For example, since we are using the Petri nets (or SANs) formalism which is a very flexible modelling way, states other than "Ok" and "not Ok" could be considered to take into account some degrading modes of components/system.

Also, the cost analysis study could and should be considered when maintenance actions are provided. Components time degradation could be also considered. We also generally suppose that the switching on and off is done instantaneously and with certainty however, this isn't always true. Thus, a probability of successful switching could be considered in the model.

Another perspective of this work is to make a feedback from such a dependability evaluation to the diagnosis problem in order to constraint the design of the diagnosis threshold J based on its impact, via FA and MD, on the system's performance such that availability. We are also planning to evaluate with numerical simulation the impact of the threshold (instead of FA and MD events) on the dependability factors.

REFERENCES

Aslund J., J. Biteus, E. Frisk, M. Krysander, and L. Nielson, 2005. A systematic inclusion of diagnosis peroformance in fault tree analysis. *IFAC world Congress*, Czech Republic.

Aslund J., J. Biteus, E. Frisk, M. Krysander, and L. Nielson, 2007. Safety analysis of autonomous systems by extended fault tree analysis. *International Journal of Adaptive Control and Signal Processing*, 21, p.287-298.

Bonivento C., M. Capiluppi, L. Marconi, A. Paoli, and C. Rossi, 2006. Reliability evaluation for fault diagnosis in complex systems. *Safe Process.*

Cassandras C.G., and S. Lafortune, 1999. *Introduction to discrete event systems*. Kluwer Academic Publishers.

Castaneda G.A, J-F. Aubry, and N. Brinzei, 2009. Simulation de Monte Carlo par automate stochastique hybride: application à un cas test pour la fiabilité dynamique. *8ème Congrès International pluridisciplinaire Qualita*, Besançon, France.

Guenab F, W. Schön, and J-L. Boulanger, 2009. Système tolerant aux défauts : Synthèse d'une méthode de reconfiguration et/ou restructuration intégrant la fiabilité de certains composants. *Journal Européen des Systèmes Automatisés*, vol. 43 – No. 10, pp. 1149-1178.

Gustafsson F., J. Aslund, E. Frisk, M. Krysander, and L. Nielsen, 2008. On threshold optimization in fault-tolerant systems. *IFAC World Congress,* South Korea.

Kuo W., V.R. Prasad, F.A. Tillman and C-L. Hwang, 2001. Optimal reliability design, *Cambridge University Press*.

Laprie J.C., 1992. Dependability: Basic concepts and associated terminology. *Dependable Computing and Fault-tolerant Systems*, 5, Springer Verlag.

Mogavar A., and J.F. Meyer, 1984. Performability modeling with stochastic activity network. *Proceeding of real-time systems symposium*, p. 215-224, Austin TX, USA.

Villemeur A., 1988. Sûreté de fonctionnement des systèmes industriels, Fiabilité - Facteurs Humains - Informatisation, *Collection de la Direction des Études et Recherches d'Électricité de France,* Edition Eyrolles.

Weber P., D. Theilliol, and C. Aubrun, 2008. Component reliability in fault-diagnosis decision making based on dynamic Bayesian networks. *Journal of Risk and Reliability*, 222(2), 161-172.