

# Module de sûreté de fonctionnement

Claire Pagetti - ENSEEIHT  
3<sup>ème</sup> TR - option SE

10 décembre 2012

## Table des matières

<b>1 Principaux concepts</b>	<b>3</b>
1.1 Qu'est-ce que la sûreté de fonctionnement	3
1.1.1 Bref historique	4
1.1.2 Coût de la sûreté de fonctionnement	4
1.2 Etude des systèmes	5
1.3 Taxonomie	7
1.3.1 Entraves	8
1.3.2 Attributs	11
1.3.3 Les moyens	12
1.4 Conception de systèmes à haut niveau de sûreté de fonctionnement	13
<b>2 Méthodes d'analyse de sûreté de fonctionnement</b>	<b>13</b>
2.1 Analyse préliminaire des dangers	14
2.2 AMDE	15
2.3 Diagramme de fiabilité	17
2.4 Méthodes quantitatives et qualitatives	18
2.4.1 Evaluation qualitative	18
2.4.2 Evaluation quantitative	18
2.4.3 Système multicomposants	22
2.5 TP sur les diagrammes de fiabilité	25
<b>3 Arbres de défaillances</b>	<b>26</b>
3.1 Construction d'un arbre de défaillance	26
3.2 TP arbres de défaillance	30
3.3 Codage des arbres de défaillance sous forme de DDB	32
<b>4 Modèles à états transitions</b>	<b>34</b>
4.1 Chaînes de Markov	34
4.1.1 Construction d'un modèle	36
4.2 Evaluation de la fiabilité, de la disponibilité et du MTTF	39
4.3 Réseaux de Petri stochastiques	41
4.3.1 Modélisation des systèmes avec des réseaux de Petri	41
4.4 AltaRica	44
4.4.1 Modélisation des système avec AltaRica	44

4.4.2	Codage avec l'outil OCAS . . . . .	46
4.4.3	TP d'AltaRica . . . . .	47

# Organisation du cours

Le module de sûreté de fonctionnement comporte 5 séances, format : cours/td/tp.

**Séance 1** : cours / exercices ;

**Séance 2** : cours / exercices / TP sur les diagrammes de fiabilité et les arbres de défaillances ;

**Séance 3** : cours / exercices / TP sur les arbres de défaillances et chaînes de Markov ;

**Séance 4** : cours / TP sur AltaRica ;

**Séance 5** : TP sur AltaRica / synthèse ;

**Examen** : à la fin du mois de janvier (exercices papier) ;

**Supports de cours** : polycopié, nombreuses références.

## 1 Principaux concepts

La sûreté de fonctionnement est apparue comme une nécessité au cours du XX<sup>ème</sup>, notamment avec la révolution industrielle. Le terme *dependability* est apparu dans une publicité sur des moteurs Dodge Brothers dans les années 1930. L'objectif de la sûreté de fonctionnement est d'atteindre le Graal de la conception de système : zéro accident, zéro arrêt, zéro défaut (et même zéro maintenance). Pour pouvoir y arriver, il faudrait tester toutes les utilisations possibles d'un produit pendant une grande période ce qui est impensable dans le contexte industriel voire même impossible à réaliser tout court. La sûreté de fonctionnement est un domaine d'activité qui propose des moyens pour augmenter la fiabilité et la sûreté des systèmes dans des délais et avec des coûts raisonnables.

### 1.1 Qu'est-ce que la sûreté de fonctionnement

La sûreté de fonctionnement est souvent appelée la *science des défaillances* ; elle inclut leur connaissance, leur évaluation, leur prévision, leur mesure et leur maîtrise. Il s'agit d'un domaine transverse qui nécessite une connaissance globale du système comme les conditions d'utilisation, les risques extérieurs, les architectures fonctionnelle et matérielle, la structure et fatigue des matériaux. Beaucoup d'avancées sont le fruit du retour d'expérience et des rapports d'analyse d'accidents.

**Définition 1 (SdF)** *La sûreté de fonctionnement (dependability, SdF) consiste à évaluer les risques potentiels, prévoir l'occurrence des défaillances et tenter de minimiser les conséquences des situations catastrophiques lorsqu'elles se présentent.*

**Définition 2 (Laprie96)** *La sûreté de fonctionnement d'un système informatique est la propriété qui permet de placer une confiance justifiée dans le service qu'il délivre.*

Il existe de nombreuses définitions, de standards (qui peuvent varier selon les domaines d'application - nucléaire, spatial, avionique, automobile, rail ...). On peut néanmoins considérer que le *Technical Committee 56 Dependability* de l'International Electrotechnical Commission (IEC) développe et maintient des standards internationaux reconnus dans le domaine de la sûreté de fonctionnement. Ces standards fournissent les méthodes et outils d'analyse, d'évaluation, de gestion des équipements, services et systèmes tout au long du cycle de développement.

### 1.1.1 Bref historique

Le tableau ci-dessous présente un bref historique de la sûreté de fonctionnement.

Période	Accidents
<a href="#">Jusqu'aux années 30</a> Approche intuitive : renforcer l'élément le plus faible Premiers systèmes parallèles et redondants Approche statistique, taux de défaillance Premières estimation de probabilité d'accidents d'avion Pugsley : premier objectif de safety $\text{taux d'accident d'avion} \leq 10^{-5}$ per flight hour	Explosion poudrière (1794) Accident chemin de fer (1842) Titanic (1912)...
<a href="#">Années 40</a> Analyse des missiles allemands V1 (Robert Lusser) Loi de Murphy " <i>If anything can go wrong, it will</i> " Quantification de la disponibilité	
<a href="#">Années 50</a> Advisory Group on Reliability of Electronic Equipment (AGREE) - Réduction des coûts de maintenance - Augmentation de la fiabilité - MTBF	Tcheliabinsk 40 (1957)
<a href="#">Années 60</a> Analyses des modes de défaillance et de leurs effets Programmes de recherche spatiaux Arbre de défaillance (missile Minuteman) Arbres des causes (Boeing - NASA) Livres sur la fiabilité (ex. Barlow and Proschan)	Torrey Canyon (1967)
<a href="#">Années 70</a> Analyse des risques Collecte de données REX	
<a href="#">Années 80 à nos jours</a> Nouvelles techniques (simulation, réseaux de Petri,..) Modélisation	Tchernobyl (1986) Ariane V (1996) DART (NASA, 2005) Vol Rio Janeiro...

### 1.1.2 Coût de la sûreté de fonctionnement

Le coût d'un haut niveau de sûreté de fonctionnement est très onéreux. Le concepteur doit faire des compromis entre les mécanismes de sûreté de fonctionnement nécessaires et les coûts économiques. Les systèmes qui ne sont pas sûrs, pas fiables ou pas sécurisés peuvent être rejetés par les utilisateurs. Le coût d'une défaillance peut être extrêmement élevé. Le coût de systèmes avec un faible niveau de sûreté de fonctionnement est illustré dans les figures ci-dessous.

### Coût moyen d'indisponibilité

Secteur industriel	Production et distribution d'énergie	2,8	Millions d'Euros par heure perdue
	Production manufacturière	1,6	
	Institutions financières	1,4	
	Assurances	1,2	
	Commerce	1,1	
	Banques	1	

### Coût annuel des défaillances informatiques

Estimation compagnies d'assurance (2002)	France (secteur privé)	USA	Royaume Uni
Fautes accidentelles	1,1 G€	4 G\$	
Malveillances	1,3 G€		1,25 G£
Estimation globale	USA : 80 G\$	EU : 60 G€	

### Coûts de maintenance

Logiciel embarqué de la navette spatiale : 100 M \$ / an

### Coût logiciels abandonnés (défaillance du processus de développement)

USA [Standish Group, 2002, 13522 projets]	Succès	Remise en question	Abandon
	34%	51%	15%
~ 38 G\$ de pertes (sur total 225 G\$)			

FIGURE 1 – Quelques chiffres [Laprie07]

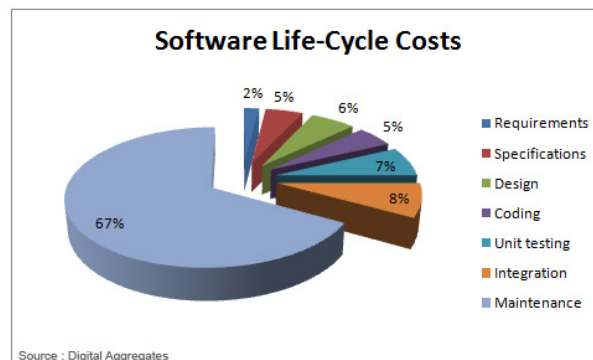


FIGURE 2 – Coût de la maintenance

## 1.2 Etude des systèmes

L'objet sous étude est le système et les fonctions qu'il fournit. Il existe de nombreuses définitions de système dans le domaine des systèmes d'ingénierie.

**Définition 3 (Un système)** *Un système peut être décrit comme un ensemble d'éléments en interaction entre eux et avec l'environnement dont le comportement dépend :*

- des comportements individuels des éléments qui le composent,
- des règles d'interaction entre éléments (interfaces, algorithmes, protocoles),
- de l'organisation topologique des éléments (architectures).

Le fait que les sous-systèmes sont en interaction implique que le système n'est pas simplement la somme de ses composants. En toute rigueur, un système dans lequel un élément est défaillant devient un nouveau système, différent du système initial.

**Exemple 1** Une installation chimique, une centrale nucléaire ou un avion sont des systèmes. Le contrôle-commande est un sous-système, une vanne ou un relais sont des composants. La nature technologique d'un système est variée : électrique, thermo-hydraulique, mécanique ou informatique.

**Assurer les fonctions** Tout système se définit par une ou plusieurs fonctions (ou missions) qu'il doit accomplir dans des conditions et dans un environnement donnés. L'objet d'étude de la sûreté de fonctionnement est la *fonction*. Une fonction peut être définie comme l'action d'une entité ou de l'un de ses composants exprimée en terme de finalité. Il convient de distinguer les fonctions et la structure (ou encore architecture matérielle support).

- fonction principale : raison d'être d'un système (pour un téléphone portable, la fonction principale est la communication entre 2 entités) ;
- fonctions secondaires : fonctions assurées en plus de la fonction principale (sms, horloge, réveil, jeux . . . ) ;
- fonctions de protection : moyens pour assurer la sécurité des biens, des personnes et environnement ;
- fonctions redondantes : plusieurs composants assurent la même fonction.

Une description fonctionnelle peut généralement se faire soit par niveau soit pour un niveau donné. Une description par niveau est une arborescence hiérarchisée. On donne l'exemple d'une description fonctionnelle d'une machine à laver dans la figure 3.

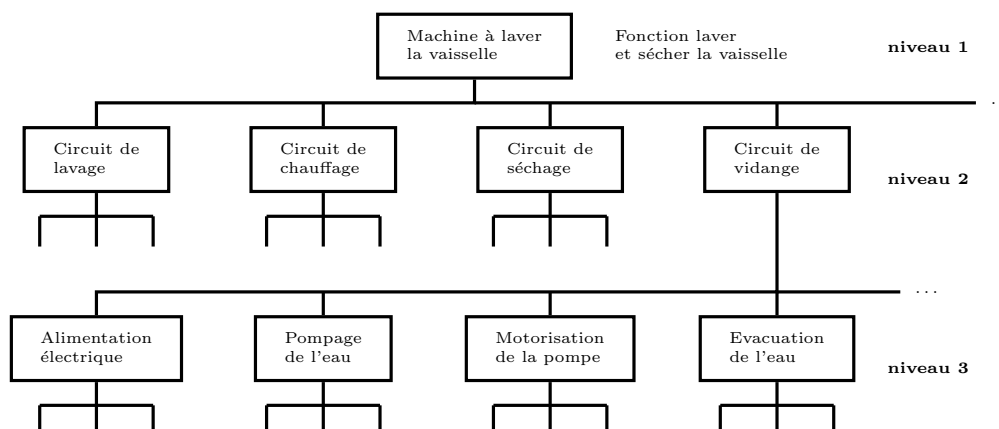


FIGURE 3 – Description fonctionnelle d'une machine à laver la vaisselle

On peut également désirer représenter les échanges de données entre fonctions, pour un niveau de granularité donné. On parle alors d'*architecture fonctionnelle*. Formellement, l'architecture fonctionnelle est constituée d'un graphe  $(\mathcal{F}, \mathcal{C}_{\mathcal{F}})$  orienté pour lequel l'ensemble des nœuds  $\mathcal{F} = \{f_1, \dots, f_m\}$  désigne les fonctions et l'ensemble des arcs,  $\mathcal{C}_{\mathcal{F}} \subseteq \mathcal{F} \times \mathcal{F}$ , représente les échanges de données entre les fonctions. Un arc  $(f_i, f_j) \in \mathcal{C}_{\mathcal{F}}$  modélise un flux de données  $f_i$  vers  $f_j$ . La figure 4 illustre l'architecture fonctionnelle de niveau 3 de l'exemple de la machine à laver.

**Structure du système** Les fonctions sont réalisées par le système à partir de ses composants. La structure du système doit être prise en compte pour les analyses de sûreté de fonctionnement. Pour cela, il faut décrire les composants matériels, leur rôle, leurs caractéristiques et leurs

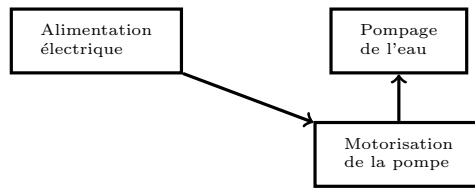


FIGURE 4 – Architecture fonctionnelle de la machine à laver

performances. On peut à nouveau utiliser une description en niveau. La figure 5 identifie les composants intervenant dans la structure de la machine à laver.

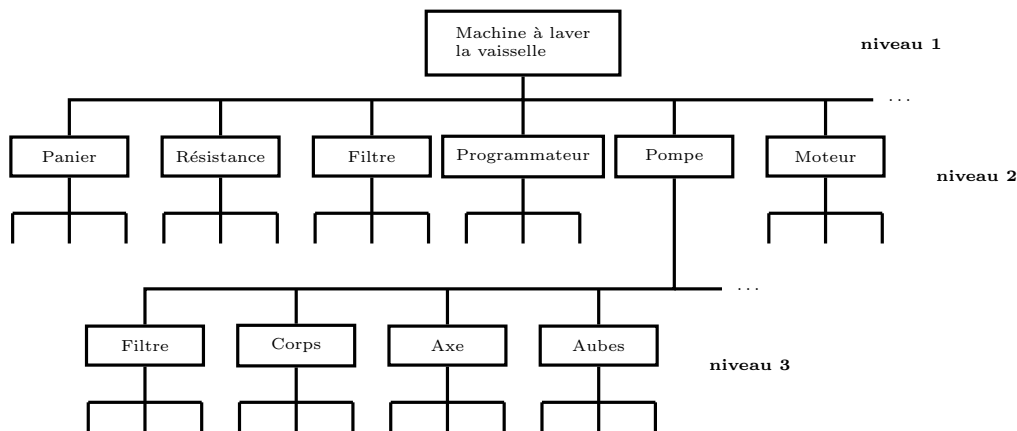


FIGURE 5 – Décomposition matérielle d'une machine à laver la vaisselle

Il faut également décrire les connexions entre composants, ce qui peut être fait par un graphe orienté pour lequel l'ensemble des nœuds désigne l'ensemble de  $n$  ressources connectées entre elles par des liaisons représentées par les arcs.

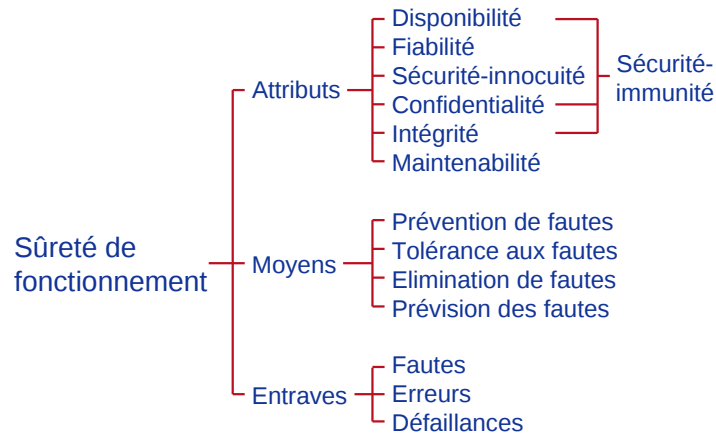
Enfin, il est également important dans certains cas de préciser la localisation des composants. Les analyses de sûreté de fonctionnement reposent sur des hypothèses au sujet de l'indépendance des défaillances des fonctions élémentaires. Le partage de ressources et l'installation de ces ressources dans une même zone risquent de violer les exigences d'indépendances. Par exemple, un éclatement pneu dans un avion peut entraîner la défaillance de plusieurs composants.

### 1.3 Taxonomie

La sûreté de fonctionnement manipule un certain nombre de concepts que nous précisons dans cette partie en donnant des définitions précises. La sûreté de fonctionnement peut être vue comme étant composée des trois éléments suivants :

- Attributs : points de vue pour évaluer la sûreté de fonctionnement ;
- Entraves : événements qui peuvent affecter la sûreté de fonctionnement du système ;
- Moyens : moyens pour améliorer la sûreté de fonctionnement.

Ces notions sont résumées dans la figure 6.



1

FIGURE 6 – Arbre de la sûreté de fonctionnement [Laprie]

### 1.3.1 Entraves

Commençons par détailler les entraves qui peuvent affecter le système et dégrader la sûreté de fonctionnement. Les entraves sont réparties en 3 notions : les fautes, les erreurs et les défaillances qui s'enchaînent comme illustré dans la figure 7. Les définitions sont récursives car la défaillance d'un composant est une faute pour le système qui le contient.

**Définition 4 (Faute / Fault)** *La cause de l'erreur est une faute (par exemple un court-circuit sur un composant, une perturbation électromagnétique ou une faute de développement logiciel).*

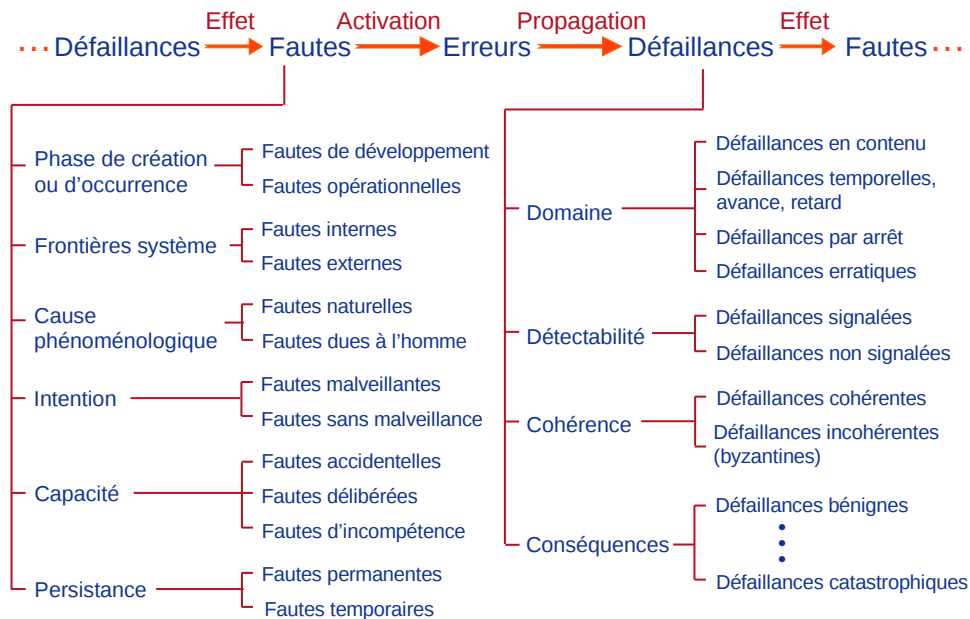
**Définition 5 (Erreur / Defect)** *La cause de la défaillance est une erreur affectant une partie de l'état du système (par exemple, une variable erronée).*

**Définition 6 (Défaillance / Failure)** *Une défaillance est la cessation de l'aptitude d'une entité à accomplir une fonction requise.*

**Définition 7 (Panne)** *La panne est l'inaptitude d'une entité à accomplir une mission. Une panne résulte toujours d'une défaillance.*

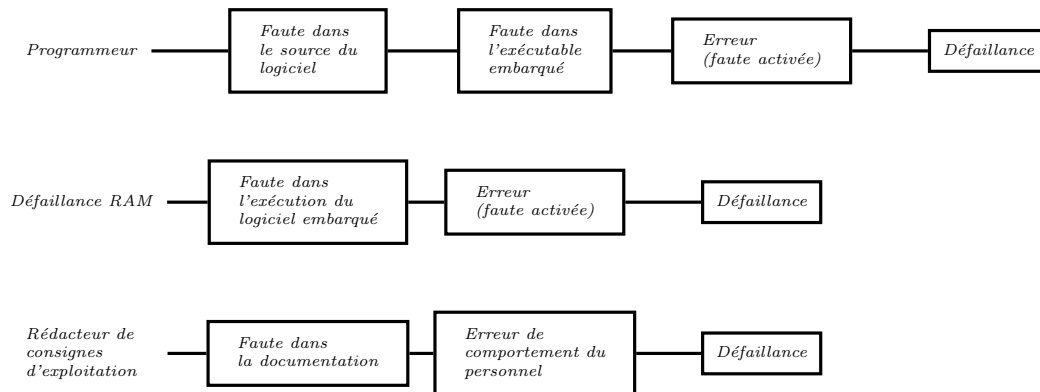
**Exemple 2** *Voici trois exemples d'occurrence de défaillances.*



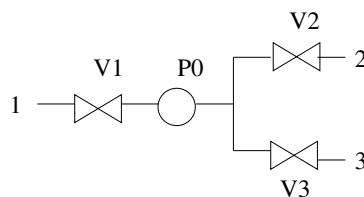


1

FIGURE 7 – Enchaînement et propagation des erreurs [Laprie96]



**Exercice 1** On considère le système hydraulique suivant. Il est destiné au transport de l'eau du point 1 aux lieux de consommation 2 et 3. Il contient les vannes  $V_1$ ,  $V_2$  et  $V_3$ , la pompe centrifugeuse  $P_0$  et les tuyaux adjacents aux composants hydrauliques. Identifier des fautes, des erreurs et des défaillances possibles.



Les défaillances dans un système peuvent avoir des effets différents. Certaines défaillances n'affectent pas directement les fonctions du système et ne nécessitent qu'une action corrective ;

d'autres, en revanche, affectent la disponibilité ou la sécurité. On utilise généralement une échelle de gravité des effets et on considère traditionnellement 4 catégories de défaillances. Ces catégories sont représentées dans la table 1.

Défaillance mineure ( <i>minor</i> )	Défaillance qui nuit au bon fonctionnement d'un système en causant un dommage négligeable au système ou à son environnement sans présenter de risque pour l'homme
Défaillance significative ( <i>major</i> )	Défaillance qui nuit au bon fonctionnement sans causer de dommage notable ni présenter de risque important pour l'homme
Défaillance critique ( <i>hazardous</i> )	Défaillance qui entraîne la perte d'une (ou des) fonction(s) essentielle(s) du système et cause des dommages importants au système en ne présentant qu'un risque négligeable de mort ou de blessure
Défaillance catastrophique ( <i>catastrophic</i> )	Défaillance qui occasionne la perte d'une (ou des) fonction(s) essentielle(s) du système en causant des dommages importants au système ou à son environnement et/ou entraîne la mort ou des dommages corporels

TABLE 1 – Classification des défaillances en fonction des effets

**Définition 8 (Mode de défaillance / Failure mode)** *Un mode de défaillance est l'effet par lequel une défaillance est observée. Plus, précisément, il s'agit d'un des états possibles d'une entité en panne pour une fonction requise donnée.*

On classe généralement les modes de défaillance en 4 catégories représentées dans la table 2.

Mode de défaillance	Explication
Fonctionnement prématuré (ou intempestif)	Fonctionne alors que ce n'est pas prévu à cet instant
ne fonctionne pas au moment prévu	ne démarre pas lors de la sollicitation
ne s'arrête pas au moment prévu	continue à fonctionner alors que ce n'est pas prévu
défaillance en fonctionnement	

TABLE 2 – Classification des modes de défaillance

**Exercice 2** *On reprend l'exercice 1. Identifier les modes de défaillance de chaque composant. On constate que la frontière entre causes de défaillance et modes de défaillance est faible. Chaque erreur doit pouvoir être rattaché à un mode. Si ce n'est pas le cas, c'est qu'il manque des modes de défaillance.*

**Définition 9 (Système cohérent)** *Un système est dit cohérent si :*

- *la panne de tous les composants entraîne la panne du système,*
- *le fonctionnement de tous les composants entraîne le fonctionnement du système,*
- *lorsque le système est en panne, aucune défaillance supplémentaire ne rétablit le fonctionnement du système,*
- *lorsque le système est en fonctionnement, aucune réparation n'induit la panne du système.*

Nous ne considérons dans la suite que des systèmes cohérents.

### 1.3.2 Attributs

Les attributs de la sûreté de fonctionnement sont parfois appelés FDMS pour Fiabilité, Disponibilité, Maintenabilité et Sécurité (RAMSS pour Reliability, Availability, Maintainability, Safety, Security).

La disponibilité est le fait d'être prêt au service.

**Définition 10 (Disponibilité / Availability)** *La disponibilité est l'aptitude d'une entité à être en état d'accomplir une fonction requise dans des conditions données, à un instant donné ou pendant un intervalle de temps donné, en supposant que la fourniture des moyens extérieurs nécessaires soit assurée.*

La fiabilité est la continuité de service.

**Définition 11 (Fiabilité / Reliability)** *La fiabilité (reliability) est l'aptitude d'un dispositif à accomplir une fonction requise dans des conditions données pendant une durée donnée.*

La sécurité est l'aptitude à ne pas provoquer d'accidents catastrophiques.

**Définition 12 (Sécurité innocuité / Safety)** *La sécurité innocuité est l'aptitude d'une entité à éviter de faire apparaître, dans des conditions données, des événements critiques ou catastrophiques.*

La maintenabilité est la capacité d'un système à revenir dans un état de fonctionnement correct après modifications et réparations.

**Définition 13 (Maintenabilité / Maintainability)** *Dans les conditions données d'utilisation, la maintenabilité est l'aptitude d'une entité à être maintenue ou rétablie, sur un intervalle de temps donné dans un état dans lequel elle peut accomplir une fonction requise, lorsque la maintenance est accomplie dans des conditions données avec des procédures et des moyens prescrits.*

D'autres attributs de sûreté de fonctionnement ont été identifiés comme par exemple la testabilité (le degré d'un composant ou d'un système à fournir des informations sur son état et ses performances), ou la *diagnosticabilité* (capacité d'un système à exhiber des symptômes pour des situations d'erreur) *survivabilité* (capacité d'un système à continuer sa mission après perturbation humaine ou environnementale) et ainsi de suite.

### 1.3.3 Les moyens

Les moyens sont des solutions éprouvées pour casser les enchaînements  $Faute \rightarrow Erreur \rightarrow Défaillance$  et donc améliorer la fiabilité du système.

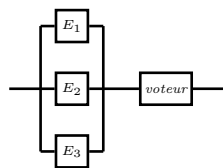
- la prévention de faute consiste à éviter des fautes qui auraient pu être introduites pendant le développement du système. Cela peut être accompli en utilisant des méthodologies de développement (cf DO 178 B/C) et de bonnes techniques d'implantation.
- L'élimination de faute peut être divisée en 2 catégories : élimination pendant la phase de développement et élimination pendant la phase d'utilisation. Pendant la phase de développement, l'idée est d'utiliser des techniques de vérification avancées de façon à détecter les fautes et les enlever avant envoi à la production. Pendant l'utilisation, il faut tenir à jour les défaillances rencontrées et les retirer pendant les cycles de maintenance.
- La prévision de faute consiste à anticiper les fautes (de manière qualitative ou probabiliste) et leur impact sur le système.
- La tolérance aux fautes consiste à mettre en place des mécanismes qui maintiennent le service fourni par le système, même en présence de fautes. On accepte dans ce cas un fonctionnement dégradé.

La tolérance aux fautes repose sur l'utilisation de mécanismes de *redondance*, l'idée est de réaliser la même fonction par des moyens différents. On distingue plusieurs types de redondance :

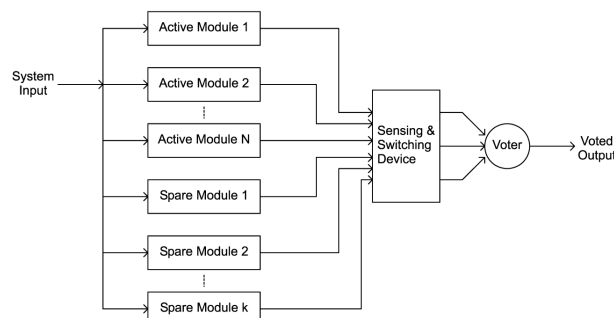
- Redondance homogène : on réplique plusieurs composants identiques
- Redondance avec dissemblance : les sous-systèmes réalisent les mêmes fonctions mais sont différents (par exemple, plusieurs équipes de conception, matériel différent).
- Redondance froide : les composants sont activés quand ceux déjà actifs tombent en panne.
- Redondance chaude : les composants tournent en parallèle et politique de prise de main.
- Redondance tiède : les composants sont idle avant de prendre la main.

D'autres mécanismes existent comme les comparateurs ou les voteurs. L'idée est de récupérer plusieurs valeurs calculées par redondance et de déterminer quelle est la plus proche de la réalité.

**Exemple 3** On considère un voteur à 3 entrées. On suppose que le voteur ne tombe jamais en panne. Il y a plusieurs types de voteur : le voteur 1/3, 2/3 et 3/3. Dans le premier cas, il prend une valeur sur 3, dans le deuxième, il faut que deux composants au moins s'accordent et dans le troisième tous les composants doit avoir la même valeur.



On considère un système composé de composants actifs et passifs.



On peut également utiliser un chien de garde (watchdog en anglais) : il s'agit d'un mécanisme destiné à s'assurer qu'un automate ou un ordinateur ne reste pas bloqué à une étape particulière du traitement qu'il effectue. C'est une protection destinée généralement à redémarrer le système, si une action définie n'est pas exécutée dans un délai imparti.

## 1.4 Conception de systèmes à haut niveau de sûreté de fonctionnement

Le problème général est le suivant :

**Données :** – une exigence de fiabilité sur le système complet (par exemple une valeur maximale de  $\lambda$ , une valeur minimale de  $A$ )  
– des composants utilisables pour construire le système avec leur fiabilité (normalement plus faible que la fiabilité globale requise)

**Problème :** Trouver une architecture du système complet qui répond aux exigences de fiabilité

La solution à ce problème est de :

1. Construire une architecture candidate
2. Analyser la fiabilité de cette architecture
3. Si le résultat est suffisant, la solution est correcte sinon retourner en 1.

Trouver des architectures candidates peut rapidement devenir un casse tête donc on utilise généralement des patterns. Nous nous concentrerons dans les parties suivantes sur le point 2.

## 2 Méthodes d'analyse de sûreté de fonctionnement

Une analyse prévisionnelle de sûreté de fonctionnement est un processus d'étude d'un système réel de façon à produire un modèle abstrait du système relatif à une caractéristique de sûreté de fonctionnement (fiabilité, disponibilité, maintenabilité, sécurité). Les éléments de ce modèle seront des événements susceptibles de se produire dans le système et son environnement, tels par exemple :

- des défaillances et des pannes des composants du système,
- des événements liés à l'environnement,
- des erreurs humaines en phase d'exploitation.

Le modèle permet ainsi de représenter toutes les défaillances et les pannes des composants du système qui compromettent une des caractéristiques de SdF.

Afin d'aider l'analyste, plusieurs méthodes d'analyse ont été mises au point. Les principales sont :

**APD** Analyse Préliminaire des Dangers,

**AMDE** Analyse des Modes de Défaillances et de leurs Effets,

**MDS** Méthode du Diagramme de Succès,

**MTV** Méthode de la Table de Vérité,

**MAC** Méthode de l'Arbre des Causes,

**MCPR** Méthode des Combinaisons de Pannes Résumées,

**MACQ** Méthode de l'Arbre des Conséquences,

**MDCC** Méthode du Diagramme Causes-Conséquences,

**MEE** Méthode de l'Espace des Etats.

Nous ne verrons dans la suite que quelques unes de ces méthodes.

## 2.1 Analyse préliminaire des dangers

L'analyse préliminaire des dangers a été utilisée la première fois aux Etats-Unis dans les années 60 dans le cadre d'une analyse de sécurité de missiles à propergol liquide. Elle a ensuite été formalisée par l'industrie aéronautique et notamment pas le société Boeing. L'analyse se fait en phase amont de conception. L'objectif est d'identifier les dangers d'un système et leurs causes puis d'évaluer la gravité des conséquences liées aux situations dangereuses. L'identification des dangers est effectuée à l'aide de l'expérience et du jugement des ingénieurs, aidés de liste-guides élaborées dans des domaines précis et régulièrement enrichies. Les grandes étapes de cette analyse sont :

1. Identification du contexte opérationnel dans lequel évolue le système.
2. Identification des dangers potentiels et de la sévérité de leurs conséquences.
3. Définition d'actions correctives.
4. Vérification de la complétude de la liste des conditions de panne issue de l'analyse de risque et de compléter les exigences de sécurité. Fournit également les premières indications pour l'architecture du système afin de mitiger les conséquences.
5. Evaluation de l'atteinte des objectifs de SdF.

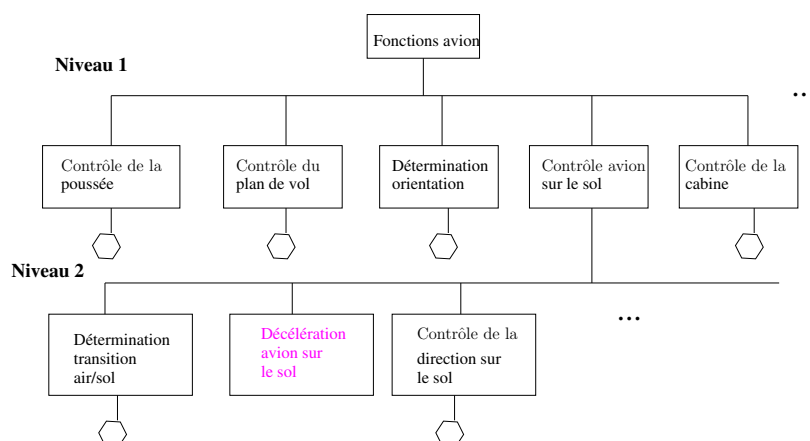
On illustre cette analyse avec un exemple extrait de l'ARP4761 (Aerospace Recommended Practice). On cherche à mettre des objectifs de sécurité sur le système de freinage.

**Que doit-on évaluer ?** Il faut d'abord identifier le système à évaluer, le contexte opérationnel, et les fonctions.



avion, piste, contrôle de l'avion au sol

La description fonctionnelle de l'avion est la suivante.

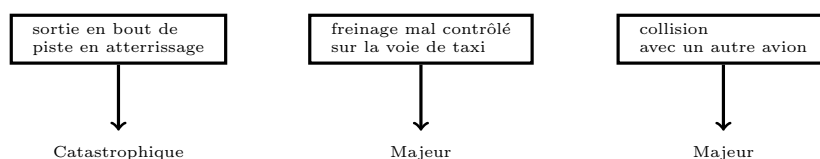


**Que peut-il arriver ?** On identifie ensuite les dangers fonctionnels (dangers dus à de mauvaises performances des fonctions, ou à des problèmes contextuels)



Perte de la décélération sur la piste

**A quel point cela est-il mauvais ?** On évalue les conséquences des dangers dans le pire cas.



**A quelle fréquence est-ce acceptable ?** On détermine des objectifs de safety en terme de fréquence acceptable pour événements ayant une certaine sévérité. (fh = flight hour)

catastrophique	$< 10^{-9}/fh$
hasardeux	$< 10^{-7}/fh$
majeur	$< 10^{-5}/fh$

**Et on recommence** Il faut faire cette étude pour toutes les fonctions et tous les risques. Le résultat partiel de l'analyse est résumé dans la figure 8.

## 2.2 AMDE

La méthode de Analyse des Modes de Défaillances et de leurs Effets (AMDE) est une des premières méthodes systématiques permettant d'analyser les défaillances. Elle a été développée par l'armée américaine et se trouve dans la première guideline Military Procedure MIL-P-1629 "Procedures for performing a failure mode, effects and criticality analysis" du 9 novembre 1949. Cette analyse est largement utilisée pendant les phases initiales de développement. Une AMDE (FMEA pour Failure Mode and Effects Analysis) est une analyse détaillée de toutes les défaillances simples, de leurs conséquences (ainsi qu'un chiffrage préliminaire de probabilité d'occurrence). Elle permet d'identifier les éléments critiques de sécurité (provoquant des événements critiques ou catastrophiques) ainsi que les fautes dormantes. En résumé, une AMDE permet :

1. identifier les modes de défaillances des différentes parties du système,
2. d'évaluer les effets de chaque mode de défaillance des composants sur les fonctions du système,
3. d'identifier les modes de défaillances qui auront un effet important sur la sécurité, la fiabilité, la sécurité ...

1. Function	2. Failure Conditions, Hazards	3. Phase, State, Mode	4. Effects on Aircraft	5. Classification	6 support	7. verificatio n
Decelerate aircraft on the ground	<b>Loss of deceleration capabilities</b>	Landing/ RTO/ Taxi	See below			
	a) unannunciated	Landing/ RTO	Crew unable to decelerate, resulting in a high speed overrun	Catastrophic		FT
	b) annunciated	Landing	Crew selects a more suitable airport, prepares occupants for overrun	Hazardous	Emergency landing procedures	FT
	c) unannunciated	Taxi	Crew unable to stop the aircraft, resulting low speed contact with obstacles	Major		
	d) annunciated	Taxi	Crew steers the aircraft clear for any obstacles and calls for a tug	No safety effect		
	<b>Asymmetric deceleration</b>	Landing/ RTO	See below			
	a) unannunciated	Landing/ RTO	Offside excursion from the runway	Major		
	b) annunciated	Landing	Crew is prepared and counters with rudder & nose wheel steering input	Minor		
	c) Asymmetric deceleration	Taxi	Slightly diverts from intended course	No safety effects		

FIGURE 8 – Exemple d'analyse de risque

## Procédure

1. Définir le système à analyser.
  - (a) Les missions et fonctions principales du système
  - (b) Conditions opérationnelles et environnementale
2. Collecter les informations disponibles qui décrivent le système (incluant schémas, spécifications, listes de composants, ...) et collecter les informations sur les précédents conceptions similaires.
3. Préparer les rapports AMDE.

**Exemple 4** Considérons l'exemple extrait du livre d'A. Villemeur et représenté dans la figure 9. Le système permet à un opérateur de commander le fonctionnement du moteur à courant continu; pour cela l'opérateur appuie sur un bouton presseur (B.P.) provoquant ainsi l'excitation d'un relais, la fermeture du contact associé et l'alimentation électrique du moteur. Lorsque l'opérateur relâche la pression, le moteur s'arrête. Un fusible permet de protéger le circuit électrique contre tout court-circuit.

On suppose que le fil AB traverse une zone où se trouvent des vapeurs inflammables : on admet que l'analyse préliminaire des dangers a montré que l'événement indésirable est la surchauffe du fil AB.

Le système est conçu pour faire fonctionner le moteur électrique pendant un temps très court; on admet qu'un fonctionnement prolongé peut entraîner sa destruction par suite d'un échauffement du moteur et de l'apparition d'un court-circuit. On admet également qu'après l'apparition d'un courant élevé dans le circuit dû à un court-circuit, le contact du relais reste collé, même après la désexcitation du relais.



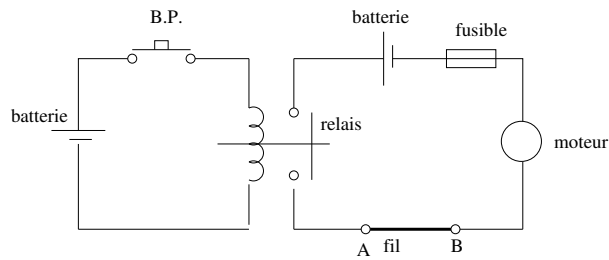


FIGURE 9 – Exemple d'illustration

*L'analyse ne porte que sur le bouton-presseur et le relais ; et on ne considère pour ces composants qu'un ou deux modes de défaillances.*

<i>Composant</i>	<i>Modes de défaillance</i>	<i>Causes possibles</i>	<i>Effets sur le système</i>
<i>B.P.</i>	<ul style="list-style-type: none"> <li>- le B.P. est bloqué</li> <li>- contact du B.P. bloqué</li> </ul>	<ul style="list-style-type: none"> <li>- défaillance mécanique</li> <li>- défaillance mécanique</li> <li>- erreur humaine</li> </ul>	<ul style="list-style-type: none"> <li>- moteur ne tourne pas</li> <li>- moteur tourne trop longtemps : d'où court-circuit moteur et fusion fusible</li> </ul>
<i>Relais</i>	<ul style="list-style-type: none"> <li>- contact bloqué ouvert</li> <li>- contact collé</li> </ul>	<ul style="list-style-type: none"> <li>- défaillance mécanique</li> <li>- défaillance mécanique</li> <li>- courant élevé dans circuit</li> </ul>	<ul style="list-style-type: none"> <li>- moteur ne tourne pas</li> <li>- moteur tourne trop longtemps (idem)</li> </ul>

## 2.3 Diagramme de fiabilité

Historiquement, la méthode du diagramme de fiabilité ou de succès est la première à avoir été utilisée pour analyser les systèmes. Ses limites sont rapidement apparues, néanmoins elle permet de modéliser rapidement le système. On suppose dans la suite que le système n'est pas réparable.

**Définition 14** *Un diagramme de fiabilité est défini par :*

- Une entrée  $E$ , un corps de diagramme et une sortie  $S$
- Un flux est transmis de  $E$  jusqu'à  $S$  en passant par les différents chemins.
- Défaillance d'une entité arrête le flux au niveau du composant.
- S'il n'existe pas de chemin jusqu'à  $S$ , le système est défaillant, sinon il fonctionne.
- Configuration série ou/et parallèle.

**Exercice 3** *On reprend l'exercice 1. Donner le diagramme de fiabilité associé.*

**Exercice 4** *Un système contient 5 blocs. Deux blocs lisent deux entrées capteur, un bloc réalise le traitement et les deux derniers blocs contrôlent deux actionneurs identiques. Les entrées et le calculateur sont nécessaires à l'exécution de la fonction, en revanche un seul actionneur est suffisant. Donner le bloc diagramme associé à ce système.*

## 2.4 Méthodes quantitatives et qualitatives

### 2.4.1 Evaluation qualitative

Une analyse qualitative de sûreté de fonctionnement consiste à :

**Donnée :** structure du système (par exemple un diagramme de fiabilité)

**Résultat :** calculer la combinaison des composants qui amène à la défaillance du système.

Pour exprimer la combinaison des blocs entraînant la défaillance, les analystes utilisent deux concepts : les chemins à succès et les coupes.

**Définition 15 (Chemin à succès)** *Un chemin à succès est un ensemble de blocs de base qui réalisent la fonction. Un chemin est dit minimal s'il ne contient aucun sous chemin.*

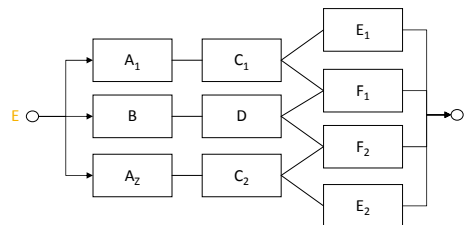
**Définition 16 (Coupe)** *Une coupe décrit un ensemble de blocs de base dont la défaillance entraîne la défaillance du système. Une coupe est dite minimale si en retirant n'importe quel bloc de la liste, le système n'est plus défaillant. La taille (ou l'ordre) de la coupe est le nombre d'éléments dans la liste.*

La connaissance des coupes minimale permet d'établir *qualitativement* la liste des composants critiques d'après l'organisation fonctionnelle du système.

**Exercice 5** *Combien y a-t-il de coupes d'ordre 1 sur un système ayant  $n$  éléments en série ? sur un système ayant  $n$  éléments en redondance ?*

**Exercice 6** *On reprend l'exercice 1. Calculer les chemins minimaux et les coupes minimales.*

**Exercice 7** *Calculez les coupes et les chemins minimaux du système suivant :*



### 2.4.2 Evaluation quantitative

Une analyse quantitative de sûreté de fonctionnement consiste à :

**Données :** – structure du système (par exemple un diagramme de fiabilité)

– probabilité d'occurrence des défaillances des blocs de base

**Résultat :** Evaluer la probabilité de défaillance du système complet.

**Système à composant unique** Dans cette partie, nous allons étudier le comportement stochastique des systèmes à composant unique subissant des défaillances en les observant dans le temps. On suppose que le système se met en fonctionnement à l'instant  $t=0$  et ne présente qu'un seul mode de défaillance. Le composant fonctionne pendant un temps aléatoire  $X_1$  au bout duquel il tombe en panne. Il y reste pendant un temps aléatoire  $Y_1$  durant son remplacement puis est remis en fonctionnement et ainsi de suite. On dit que le système est *réparable*. Lorsque le composant n'est jamais réparé, on dit qu'il est *non réparable*. La description graphique du comportement du système est donnée dans la figure 10.

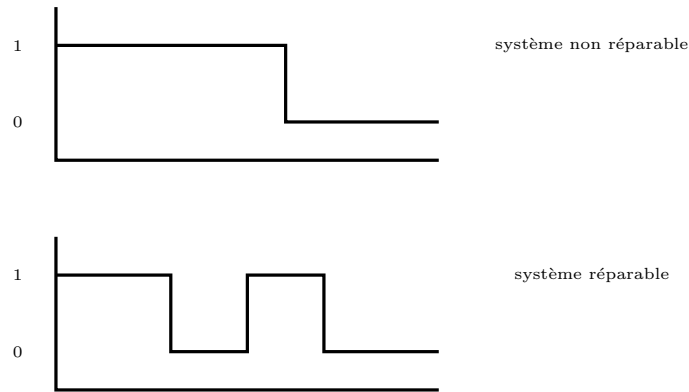


FIGURE 10 – Diagramme des phases

**Définition 17 (Taux de défaillance / Failure rate)** *Considérons un ensemble d'entités identiques et en fonctionnement à l'instant initial. On définit le taux de défaillance comme la proportion, ramenée à l'unité de temps, des entités qui ayant survécu à un temps arbitraire  $t$  ne sont plus en vie à l'instant  $t + dt$ .*

Il est fréquent que les entités présentent des taux de défaillance en fonction du temps suivant une courbe dite *en baignoire*.

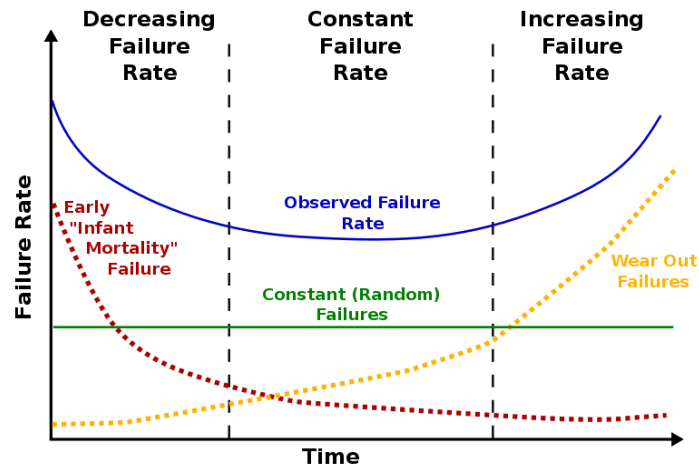


FIGURE 11 – Taux de défaillance en fonction du temps [Wikipedia]

**Rappels sur les variables aléatoires** Une variable aléatoire est une fonction définie sur l'ensemble des résultats possibles d'une expérience aléatoire. Ainsi, dans le jeu de jet d'un seul dé, le résultat est une variable aléatoire  $X$  pour laquelle nous lisons sur la face du dé les valeurs possibles de 1 à 6. En sûreté de fonctionnement, on rencontre des lois discrètes lorsque les événements qui se produisent sur un intervalle de temps donné sont des nombres entiers. C'est, par exemple, le nombre de cartes électroniques défaillantes par mois dans un système de contrôle-commande. Les variables aléatoires continues sont à valeurs réelles positives. Par

exemple, l'instant d'apparition d'une défaillance dans une structure métallique est une variable continue.

Pour les évaluations probabilistes, on utilise deux variables aléatoires :

$$\begin{array}{ll} \text{variable d'état} & X(t) = \begin{cases} 1 & \text{si l'entité fonctionne à l'instant } t \\ 0 & \text{si l'entité est défaillante à l'instant } t \end{cases} \\ \text{instant de la défaillance } T & \\ \text{(time to failure)} & \\ \text{variable continue} & \end{array}$$

En sûreté de fonctionnement, il faut faire la distinction entre l'occurrence d'un événement et son existence à l'instant  $t$ . L'assertion *l'occurrence de la défaillance du composant au temps  $t$*  signifie que la défaillance a eu lieu dans l'intervalle  $[t, t + dt]$ . En revanche, l'assertion *existence de la défaillance au temps  $t$*  signifie que la défaillance a eu lieu entre  $[0, t]$ .

La fonction de répartition d'une variable aléatoire  $X$  est la fonction  $F(x) = P(X \leq x)$ .  $F$  tend vers 0 en  $-\infty$  et vers 1 en  $+\infty$ . La densité de probabilité lorsque  $F$  est dérivable est  $f(x) = dF(x)/dx$ . Le moment d'ordre  $k$  de la variable  $X$  est  $E[X^k] = \int_{-\infty}^{\infty} x^k f(x) dx$ .  $E[X]$  est l'espérance mathématique ou la moyenne. La variance est le réel lorsqu'il existe  $V[X] = \int_{-\infty}^{\infty} (x - E[X])^2 f(x) dx$ .  $\sqrt{V[X]}$  est appelé l'écart type. On rappelle quelques lois classiques que nous utiliserons dans la suite.

**Loi exponentielle :** utilisée fréquemment car les calculs sont simples. La densité de probabilité est  $f(t) = \lambda e^{-\lambda t}$  où  $\lambda$  est une constante. La fonction de répartition est  $F(t) = \int_{-\infty}^t f(t) dt = 1 - e^{-\lambda t}$ . La moyenne est de  $1/\lambda$  et la variance de  $1/\lambda^2$ . Cette loi s'applique pour la durée de vie utile représentée dans la courbe en baignoire.

**Loi normale :** la loi normale ou de Gauss a deux paramètres  $N(m, \sigma)$  avec  $m$  la moyenne et  $\sigma$  l'écart type. La densité de probabilité est  $f(t) = \frac{1}{\sigma\sqrt{2\pi}} e^{-1/2(\frac{t-m}{\sigma})^2}$ . La fonction de répartition est  $\frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^t e^{-1/2(\frac{t-m}{\sigma})^2} dx$ . Cette loi s'applique à de nombreux phénomènes (incertitude sur des mesures ou des fabrications par exemple).

**Loi log-normale :** une variable aléatoire est distribuée suivant une loi log-normale si son logarithme est distribué selon une loi normale. La densité de probabilité est  $f(t) = \frac{1}{\sigma t\sqrt{2\pi}} e^{-1/2(\frac{\log(t)-m}{\sigma})^2}$ . La fonction de répartition est  $F(t) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^t \frac{1}{x} e^{-1/2(\frac{\log(x)-m}{\sigma})^2} dx$ . La moyenne est  $e^{m+\sigma^2/2}$  et la variance  $e^{2m+\sigma^2}(e^{\sigma^2} - 1)$ . Cette loi est souvent utilisée pour représenter les durées de réparation des composants ou les incertitudes dans la connaissance d'une donnée de sûreté de fonctionnement.

**Loi de Weibull :** cette loi dépend de 3 paramètres et permet de représenter un grand nombre de distributions expérimentales. La densité de probabilité est  $f(t) = \frac{\beta(t-\gamma)^{\beta-1}}{\sigma^\beta} e^{-(\frac{t-\gamma}{\sigma})^\beta}$  avec  $\beta > 0$ ,  $\sigma > 0$  et  $t > \gamma$ . La fonction de répartition est  $F(t) = 1 - e^{-(\frac{t-\gamma}{\sigma})^\beta}$ . La moyenne est  $\gamma + \sigma\Gamma(\frac{1+\beta}{\beta})$  et la variance  $\sigma^2(\Gamma(\frac{2}{\beta} + 1) - \Gamma^2(\frac{1}{\beta} + 1))$  avec  $\Gamma(x) = \int_0^\infty x^{\beta-1} e^{-x}$ .

### Définition probabiliste des attributs

**Définition 18 (Fiabilité)** La fiabilité d'une entité  $E$  peut s'exprimer par

$$R(t) = P(E \text{ non défaillante sur la durée } [0, t]) = R(t) = P[T > t]$$

où  $T$  est le temps de la défaillance.

On peut expérimentalement calculer cette probabilité. On prend  $n$  composants identiques qui fonctionnent à  $t=0$ , et on compte à chaque instant  $t_i$  combien sont toujours en marche  $v(t)$ . Alors,  $R(t) = v(t)/n$ .

La défiabilité  $\bar{R}(t) = 1 - R(t)$  est donc égale à la fonction de répartition de  $T$ ,  $\bar{R}(t) = F(t)$ . Il y a une relation forte entre fiabilité et le taux de défaillance  $\lambda$ .

$$R(t) = e^{-\int_0^t \lambda(x) dx}$$

Le MTTF (Mean Time to Failure) est le temps moyen de fonctionnement avant la 1<sup>ère</sup> panne.

$$MTTF = E[T] = \int_0^\infty R(t) dt$$

On peut également l'obtenir de manière expérimentale puisque  $MTTF = \lim_{n \rightarrow \infty} \Sigma t_i / n$ .

**Exercice 8** Le taux de défaillance d'un module est constant et vaut  $\lambda = 10^{-4} h^{-1}$ . Calculez le temps moyen de défaillance MTTF.

**Définition 19 (Maintenabilité)** La maintenabilité d'une entité  $E$  peut s'exprimer par

$$M(t) = P(E \text{ en panne en } 0 \text{ et réparé sur } [0, t])$$

Soit  $Y$  la variable aléatoire désignant la durée de la panne du composant, alors  $M(t) = P(Y \leq t)$ .

MUT (Mean Up Time) : durée moyenne de fonctionnement du système après réparation

MDT (Mean Down Time) : durée moyenne de non fonctionnement du système.

MTBF (Mean Time Between Failure) : durée moyenne entre 2 pannes. On a  $MTBF = MUT + MDT$ .

MTTR (Mean Time To Restoration) : durée moyenne avant remise en service.  $MTTR = E[Y] = \int_0^\infty tG(t)dt = \int_0^\infty [1 - M(t)]dt$ . Les différents temps moyens sont représentés dans la figure 12.

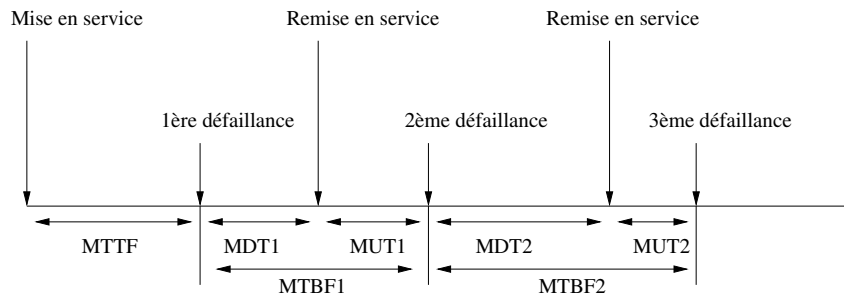


FIGURE 12 – Représentation des temps moyens dans la vie en opération

**Exercice 9** Un moteur peut être vu comme un système réparable, les brosses en carbone doivent être changées après un certain nombre d'heures en opération. Durant une année, le moteur doit être réparé 3 fois. La première réparation a lieu après 98 jours et dure 10h, la deuxième après 100 autres jours et dure 9h, la troisième après 105 autres jours et ce pendant 11 heures.

Calculer le temps moyen en opération MUT et le temps moyen de réparation MDT du moteur.

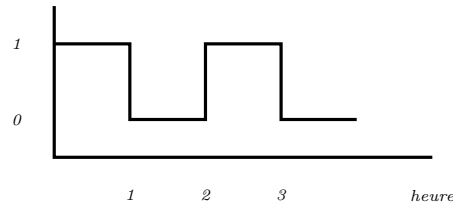
**Définition 20 (Disponibilité)** La disponibilité d'une entité  $E$  peut s'exprimer par

$$A(t) = P(E \text{ non défaillante à l'instant } t)$$

C'est une grandeur instantanée, le système peut avoir subi plusieurs pannes et réparations avant  $t$ . Lorsque l'entité n'est pas réparable, on a  $A(t) = R(t)$ . En particulier dans le cas où le taux de défaillance est constant, on  $A(t) = e^{-\lambda t}$ ;

On utilise parfois la notion de disponibilité moyenne, elle vaut  $MUT / (MUT + MDT)$ .

**Exemple 5** Dans le cas suivant, la disponibilité moyenne est de 1/2.



On résume dans la table 3 les attributs de sûreté de fonctionnement pour les lois usuelles.

loi aléatoire	application fiabilité	loi exponentielle	loi normale	loi de Weibull
fonction répartition	$\overline{R}(t)$	$1 - \exp^{-\lambda t}$	$\frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^t e^{-1/2(\frac{t-m}{\sigma})^2}$	$1 - e^{-(\frac{t-\gamma}{\sigma})^\beta}$
taux défaillance		$\lambda$	$U(t)/R(t)$	$\frac{\beta(t-\gamma)t^{\beta-1}}{\sigma^\beta}$
$F(t) = \overline{R}(t)$				
$\lambda(t)$				

TABLE 3 – Attributs de sûreté de fonctionnement pour les lois usuelles

### 2.4.3 Système multicomposants

Dans la partie précédente, nous avons étudié la défaillance d'un système à composant unique. Dans ce cas, la défaillance du composant implique la défaillance du système. Dans un système multicomposants, la défaillance du système survient à la suite de défaillance de sous-ensembles de composants. Par exemple, la défaillance d'un frein sur un vélo ne remet pas en cause le fonctionnement global même s'il faut adapter son mode d'utilisation. Par contre, la défaillance d'une roue conduit à l'immobilisation du vélo.

Soit un système à  $n$  composants, on note par  $x_i$  l'état du  $i$ -ème composant.  $x_i = 1$  si le composant fonctionne et 0 sinon. On note  $\phi(x)$  l'état du système complet. On rappelle également

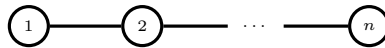
que :

$$\begin{aligned} P(A \wedge B) &= P(A) \times P(B|A) \\ &= P(A).P(B) \text{ si } A \text{ et } B \text{ sont indépendants} \end{aligned}$$

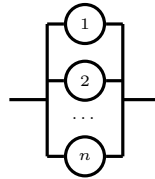
$$\begin{aligned} P(A \vee B) &= P(A) + P(B) - P(A \wedge B) \\ &\sim P(A) + P(B) \end{aligned}$$

**Exemple 6** L'état du système de freinage d'un vélo dépend de l'état des deux freins  $x_1$  et  $x_2$ . On a  $\phi(x) = 1 - (1 - x_1)(1 - x_2)$ .

**Système série** Un système est dit en série si son fonctionnement est assujéti au fonctionnement simultané de tous ses composants. Si un seul composant est en panne, alors tout le système est en panne. On a alors  $\phi(x) = \prod_i x_i$ . Si le système n'est pas réparable, on a  $A(t) = \prod_i A_i(t)$ .



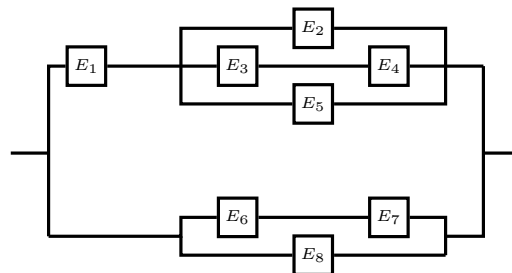
**Système parallèle** Un système est dit en parallèle si son fonctionnement est assuré si au moins un des composants est en bon état. Le système est en panne si et seulement si tous les composants sont en panne. On a alors  $\phi(x) = 1 - \prod_i (1 - x_i)$ . Si de plus, le système n'est pas réparable, alors  $A(t) = 1 - \prod_i (1 - A_i(t))$ .



**Exercice 10** Un système contient 7 modules identiques dont le taux de défaillance est  $\lambda = 10^{-4}h^{-1}$ . Calculer la fiabilité du système sur un temps de  $t = 1000$  heures lorsque :

1. tous les modules sont nécessaires à la réalisation de la fonction ;
2. au moins un module doit fonctionner.

**Exercice 11** Calculez la fiabilité du système suivant.



**Analyse d'un diagramme de fiabilité** Dans la partie 2.4.3 p. 23, nous avons illustré sur les diagrammes séries ou parallèles le calcul de la fiabilité.

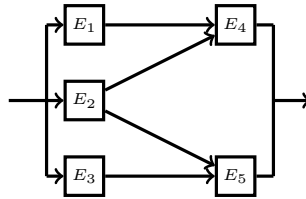
**Exercice 12** On considère  $n$  composants identiques de taux de défaillance constant  $\lambda$ . Si l'assemblage est en série, quel est le taux de défaillance du système et quel est son MTTF ?

**Analyse de système complexe** Si le système n'est pas un assemblage de structures séries/parallèles, il faut adapter le calcul.

**Théorème 1 (Bayes)** Soit  $S$  un système tel que  $A$  est un composant, alors

$$R(S) = R(S/A \text{ est OK})R(A) + R(S/A \text{ est KO})\bar{R}(A)$$

**Exemple 7** Considérons l'exemple décrit ci-dessous.



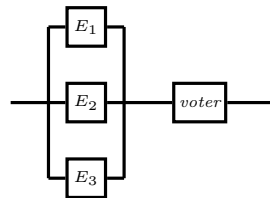
Considérons les deux événements le composant  $E2$  fonctionne à l'instant  $t$  et le composant  $E2$  est défaillant à l'instant  $t$  de probabilité respectivement  $R_2$  et  $1 - R_2$ . La fiabilité du système s'écrit alors :

$$R = P[S \text{ fonctionne à l'instant } t / E2 \text{ fonctionne à l'instant } t] \cdot R_2 + P[S \text{ fonctionne à l'instant } t / E2 \text{ est défaillant à l'instant } t] \cdot (1 - R_2)$$

Si  $E2$  fonctionne,  $S$  fonctionne si et seulement si  $E4$  ou  $E5$  fonctionne. Donc  $P[S \text{ fonctionne à l'instant } t / E2 \text{ fonctionne à l'instant } t] = 1 - (1 - R_4)(1 - R_5)$ . Si  $E2$  est défaillant,  $S$  fonctionne si et seulement si l'une des deux séries  $E1.E4$  ou  $E3.E5$  fonctionne. Donc  $P[S \text{ fonctionne à l'instant } t / E2 \text{ est défaillant à l'instant } t] = 1 - (1 - R_1.R_4)(1 - R_3.R_5)$ . Finalement,  $R = [1 - (1 - R_4)(1 - R_5)]R_2 + [1 - (1 - R_1.R_4)(1 - R_3.R_5)](1 - R_2)$ .

Certains systèmes utilisent des mécanismes plus complexes comme des voteurs.

**Exemple 8** Considérons un système composé d'un voteur et de 3 composants redondants. ON suppose que le voteur est parfait et ne tombe jamais en panne. Il y a plusieurs types de voteur : 1/3, 2/3 et 3/3. Un voteur 1/3 choisit une valeur parmi 3, un voteur 2/3 prend la valeur sur laquelle au moins 2 composants s'accordent et un voteur 3/3 nécessite que les 3 composants s'accordent pour produire une valeur.



Supposons que  $A(E_1) = 0.9$ ,  $A(E_2) = 0.8$  et  $A(E_3) = 0.7$ . On veut calculer la disponibilité du système avec voteur.



$E_1$	$E_2$	$E_3$	$1/3$	$2/3$	$3/3$
0.1	0.2	0.3	$1 - 0.1 \times 0.2 \times 0.3$		
0.1	0.2	0.7			
0.1	0.8	0.3			
0.1	0.8	0.7		$0.1 \times 0.8 \times 0.7 +$	
0.9	0.2	0.3			
0.9	0.2	0.7		$0.9 \times 0.2 \times 0.7 +$	
0.9	0.8	0.3		$0.9 \times 0.8 \times 0.3 +$	
0.9	0.8	0.7		$0.9 \times 0.8 \times 0.7$	$0.9 \times 0.8 \times 0.7$
			$=0.994$	$=0.904$	$=0.504$

**Lien avec les chemins à succès et les coupes** Lorsque tous les chemins minimaux  $\{L_i\}_{i=1\dots p}$  ont été identifiés, on obtient le résultat :

$$R = P(\Sigma_{i=1\dots p} L_i)$$

Lorsque toutes les coupes minimales  $\{C_i\}_{i=1\dots n}$  ont été identifiées, on obtient le résultat :

$$\bar{R} = P(\Sigma_{i=1\dots n} C_i) = \Sigma_i P(C_i) - \Sigma_k (-1)^k \Sigma_{i_1, \dots, i_k} P(C_{i_1} \wedge \dots \wedge C_{i_k})$$

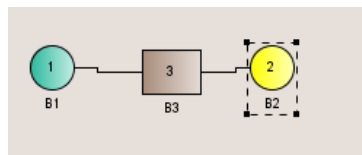
Si les probabilités sont très faibles, on approxime généralement la probabilité avec  $\Sigma_i P(C_i)$ . Sinon, on encadre

$$\Sigma_i P(C_i) - \Sigma_{i=2}^n \Sigma_{j=1}^{i-1} P(C_i.C_j) \leq \bar{R} \leq \Sigma_i P(C_i)$$

## 2.5 TP sur les diagrammes de fiabilité

On va utiliser le logiciel GRIF <http://grif-workshop.fr/tag/total/> développé par les sociétés SATODEV et TOTAL. Il existe une version gratuite installée sur les machines de la salle B302. Pour y accéder, allez dans *Démarrer → tous les programmes → GRIF 2012 → Reliability block diagramme*.

**Exercice 13** Dessinez en GRIF un bloc diagramme composé d'un unique composant. Essayez ensuite plusieurs distributions (Constant, Exponential  $-10^{-3}$ , Weibull) et calculez la fiabilité pour 1000 heures. Attention GRIF calcule la défiabilité  $\bar{R}(t)$ .



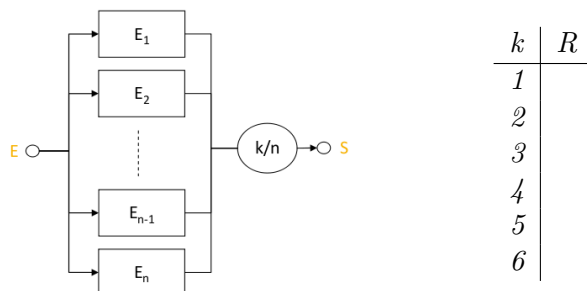
Utilisez  pour configurer les calculs et sélectionnez l'affichage des coupes.

**Exercice 14** Dessinez en GRIF un système composé de 5 éléments en série avec un taux de défaillance exponentiel  $10^{-3}$ . Calculez la fiabilité pour 1000 heures ainsi que les coupes minimales. Idem pour un système composé de 5 éléments en parallèle.

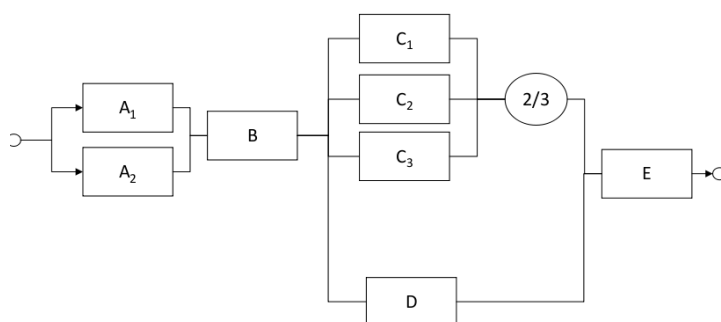
**Exercice 15** On reprend l'exemple 7, calculez les coupes minimales.

Idem avec l'exercice 11.

**Exercice 16** On considère le système suivant avec  $N=6$ . Calculez les valeurs du tableau de droite pour 1000 heures.



**Exercice 17** Calculez les chemins à succès minimaux, les coupes minimales et la fiabilité à 1000h du système suivant.



### 3 Arbres de défaillances

La méthode de l'arbre de défaillances, encore appelée arbre des causes (*fault tree*) est née en 1962 dans la société Bell Telephone grâce à Watson qui travaillait sur le projet Minuteman. Dans les années suivantes, les règles de construction ont été formalisées par Haasl en 1965, par l'University of Washington et Boeing. Dans les années 70, Vesely a jeté les bases de l'évaluation quantitative, Kinetic Tree Theory (KITTT). Enfin, en 1992, la dernière grande avancée est due à Coudert, Madre et Rauzy qui les ont codés avec des Diagrammes de décision binaires (DDB) obtenant ainsi une grande efficacité de calcul. Cette méthode a pour objectif de déterminer les combinaisons possibles d'événements qui entraînent l'occurrence d'un événement indésirable (ou redouté). L'idée est de représenter graphiquement la logique de dysfonctionnement d'un système.

#### 3.1 Construction d'un arbre de défaillance

L'analyse par l'arbre de défaillance se concentre sur un événement particulier qualifié d'*indésirable* ou de *redouté* car on ne souhaite pas le voir se réaliser. Cet événement devient le sommet de l'arbre et l'analyse a pour but d'en déterminer toutes les causes.

La syntaxe des arbres de défaillances est décrite dans la figure 13 et l'équivalence avec les diagrammes de fiabilité est donnée dans la figure 14.

On utilise généralement la convention du rond pour dénoter un événement terminal, ou une feuille. Un événement intermédiaire sera représenté par une rectangle. Quand un sous-arbre



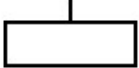





Événement / report	Dénomination	Portes	Dénomination
	Événement de base		Porte « ET »
	Événement-sommet ou événement intermédiaire		Porte « OU »
	Report (sortie)		Porte « OU exclusif »
	Le sous-arbre situé sous ce « drapeau » est à dupliquer ...		
	Report (entrée)		Porte « combinaison »
	...à l'endroit indiqué par ce second drapeau		

FIGURE 13 – Syntaxe des arbres de défaillance

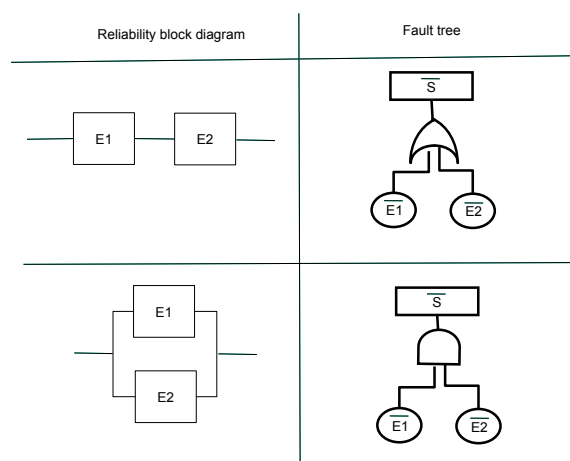
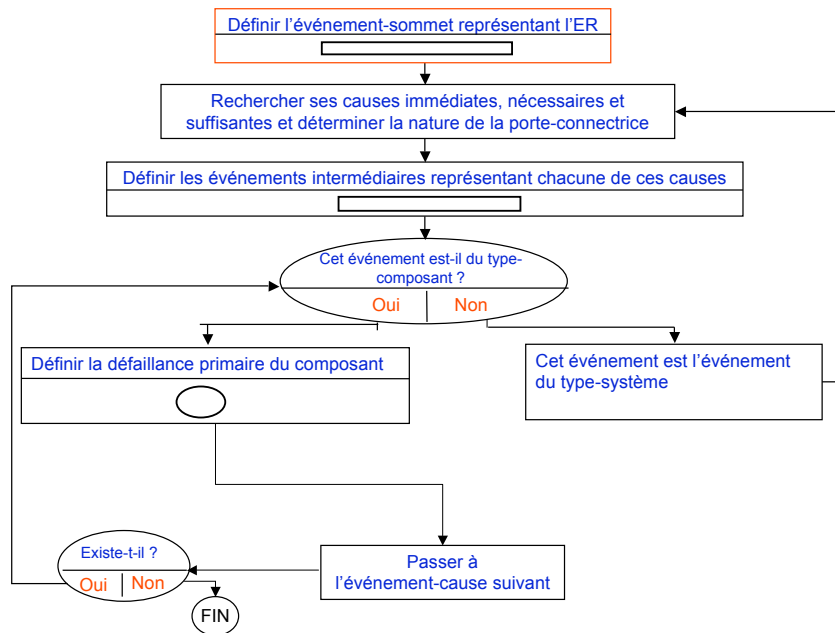


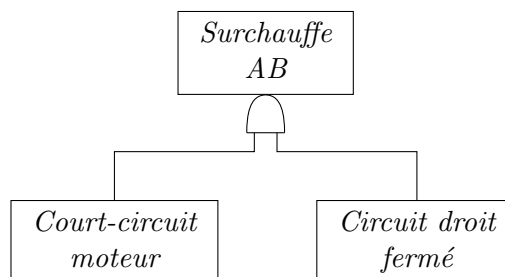
FIGURE 14 – Equivalence entre diagramme de fiabilité et arbre

apparaît plusieurs fois, on peut factoriser l'écriture en utilisant les reports symbolisés par des triangles. Dans le cas de la porte *et*, la sortie  $S$  est vraie si toutes les entrées  $E_i$  le sont. Pour la porte *ou*, la sortie  $S$  est vraie si au moins une des entrées  $E_i$  est à vrai. Dans le cas de la porte *ou exclusif*, la sortie  $S$  est vraie si une seule entrée est à vrai. Enfin, pour la porte  $k/n$ ,  $S$  est à vrai si  $k$  événements au moins sont à vrai sur les  $n$ . Il existe d'autres portes dont nous ne parlerons pas dans la suite.

Le schéma suivant est un guide permettant l'élaboration d'un arbre de défaillance. L'idée est de déterminer toutes les causes élémentaires qui mènent à l'événement redouté.



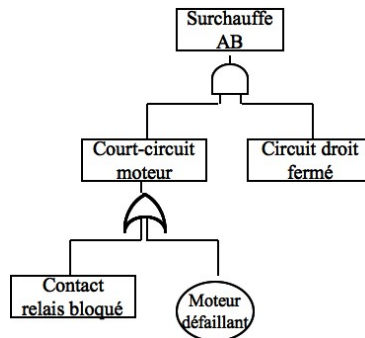
**Exemple 9** On reprend l'exemple de la figure 9. L'événement indésirable est la surchauffe du fil AB. Il ne peut résulter que de la présence d'un courant élevé dans le circuit de droite ce qui est le résultat d'un court-circuit du moteur et du fait que le circuit reste fermé. On en déduit l'arbre



On recommence pour chaque événement intermédiaire. Les causes du court-circuit moteur sont :

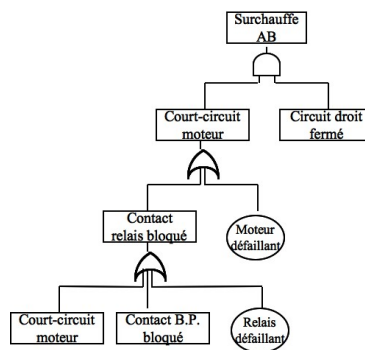
- soit une défaillance première du moteur (comme par exemple le vieillissement), c'est un événement élémentaire ;

- soit le résultat d'une cause externe, ici ce sera le contact du relais resté collé.



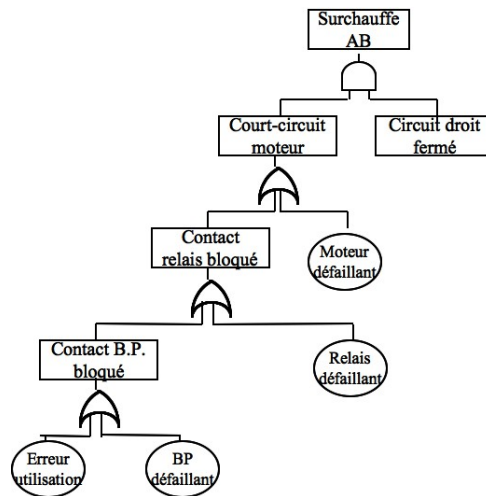
Les causes de l'événement le contact du relais resté collé sont :

- soit une défaillance première du relais (comme par exemple une défaillance d'origine mécanique), c'est un événement élémentaire ;
- soit le contact du relais reste collé si un courant élevé traverse le contact, c'est-à-dire s'il existe un court-circuit moteur ;
- soit le contact de B.P. reste collé.

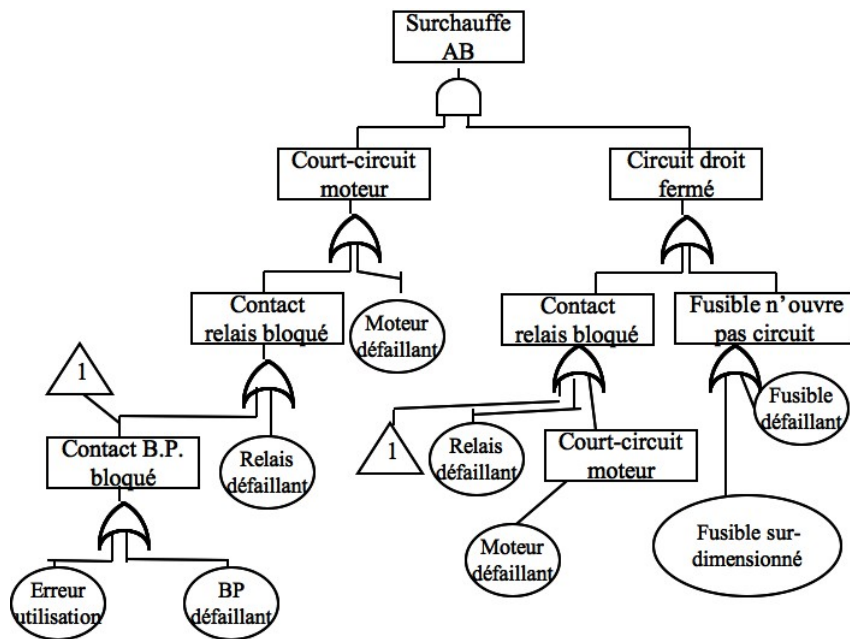


On tombe sur un problème puisque le court-circuit moteur apparaît deux fois dans la même branche. Il faut donc supprimer cet événement. Les causes de l'événement le contact du B.P. resté collé sont :

- soit une défaillance première du B.P. (comme par exemple une défaillance d'origine mécanique), c'est un événement élémentaire ;
- soit le contact du BP reste collé par suite d'une erreur humaine.



Finalement l'arbre complet est la suivant



### 3.2 TP arbres de défaillance

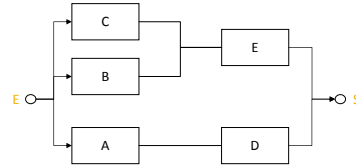
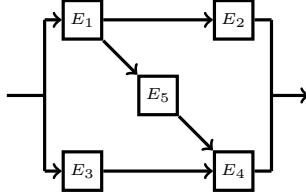
Dans ce TP, nous allons écrire des arbres de défaillance à l'aide de l'outil *GRIF Fault tree*. Ce type de logiciel est très utilisé dans l'industrie. On peut également citer les outils Arbor (Dassault), Simfia (Apsys), Aralia (Arboost Technology - maintenant inclus dans Arbor) et Fault-Tree+ (Isograph).

**Exercice 18** Dessinez deux arbres élémentaires avec une porte pour l'un et une porte ou pour l'autre, reliant deux blocs élémentaires  $e_1$  et  $e_2$ . Calculez les coupes et la fiabilité.

**Exercice 19** On reprend l'exercice 1 et on considère un mode de défaillance unique par composant (HS pour la pompe, bloquée fermée pour les vannes). L'événement redouté est : pas de débit en 2 et 3. Donner l'arbre de défaillance associé. Calculez les coupes minimales. Si Aralia

est disponible, calculez leur probabilité d'occurrence (avec lois constante et exponentielle). Un fichier est généré automatiquement à partir d'Arbor.

**Exercice 20** On considère les 2 blocs diagramme représentés ci-dessous. On suppose que tous les composants sont identiques non réparables et qu'ils ont un unique mode de défaillance (perte). Ecrivez les arbres de défaillance associés et donnez les coupes minimales.



**Exercice 21 (Etude simplifiée d'une unité de production chimique)** On considère le système représenté graphiquement dans la figure 15. L'objectif est de fabriquer un produit chimique Z à l'aide d'un réacteur alimenté par deux réactifs X et Y (ils sont tous les deux nécessaires pour obtenir le produit Z). Deux réservoirs  $R_X$  et  $R_Y$  permettent cette alimentation, sachant qu'un réservoir supplémentaire  $R_{X2}$  en produit X est également disponible en secours grâce à l'ouverture de la vanne  $V_5$ . L'unité de contrôle C est informée du débit par le détecteur DX après la vanne de régulation  $V_X$ . Cette dernière est commandée par C. La vanne de régulation  $V_Y$  le circuit en produit Y. Les vannes  $V_1$ ,  $V_2$ ,  $V_3$  et  $V_4$  sont des vannes de sécurité en cas de détection de fuites (le produit X est nocif) et aussi en cas d'arrêt d'urgence (risque d'explosion du réacteur). On ne tiendra pas compte ici des fuites des canalisations.

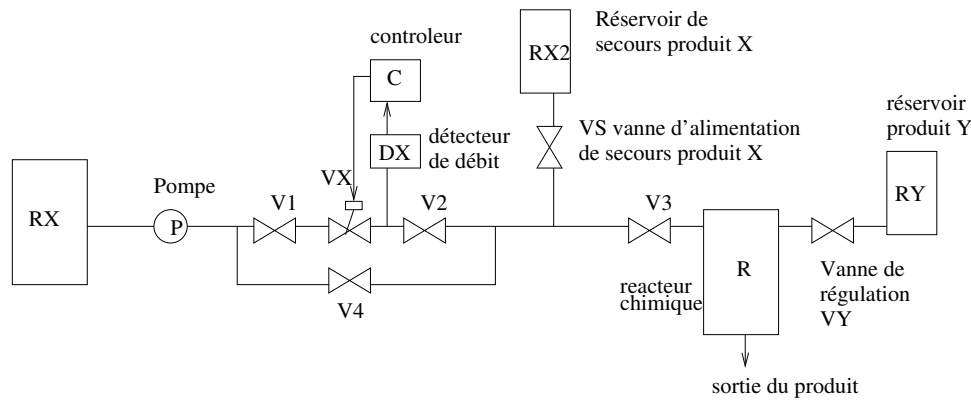


FIGURE 15 – Unité de production

On considère que tous les composants sont susceptibles d'être défaillants (on supposera qu'il n'y a qu'un mode de défaillance panne) sauf le détecteur et le réservoir  $R_{X2}$ . Les taux de défaillance sont supposés constants. On souhaite maintenir la production du produit Z quand il n'y a pas d'incidents majeurs dans le système.  $V_1$ ,  $V_2$ ,  $V_3$  sont ouvertes initialement et  $V_4$  est fermée.

composants	taux $\lambda$
$R_X$ , $R_Y$ , $R$	$5.10^{-5}/h$
$V_X$ , $V_Y$	$5.10^{-5}/h$
$P$ , $C$	$10^{-4}/h$
$V_1$ , $V_2$ , $V_3$ , $V_4$	$5.10^{-5}/h$

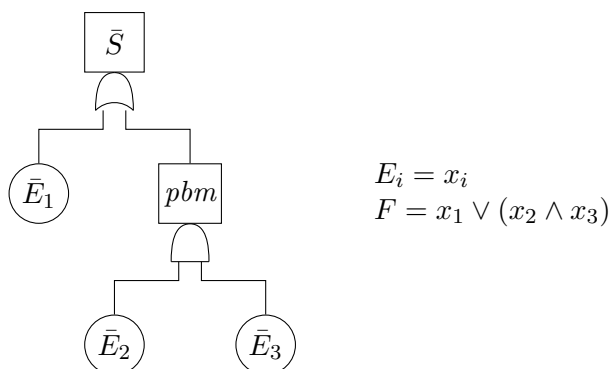
Construire le bloc diagramme associé ainsi que l'arbre de défaillance. En déduire les coupes minimales et leur probabilité.

### 3.3 Codage des arbres de défaillance sous forme de DDB

En 1992, J.-C. Coudert et O. Madre d'un côté, A. Rauzy d'un autre, ont proposé un codage efficace des arbres de défaillances. Un arbre de défaillance est équivalent à une formule binaire. En effet, un arbre est constitué d'événements  $E = \{e_1, \dots, e_n\}$  et de portes logiques  $P = \{ou, et \dots\}$ . Chaque événement est transformé en une variable booléenne  $x_i$  qui vaut 1 si l'événement s'est produit et 0 sinon. Les portes logiques sont directement traduites en connecteurs logiques.

événement $e_i$	variable booléenne $x_i$
se produit	1
ne se produit pas	0

**Exemple 10** Voici la traduction en formule booléenne d'un arbre de défaillance.

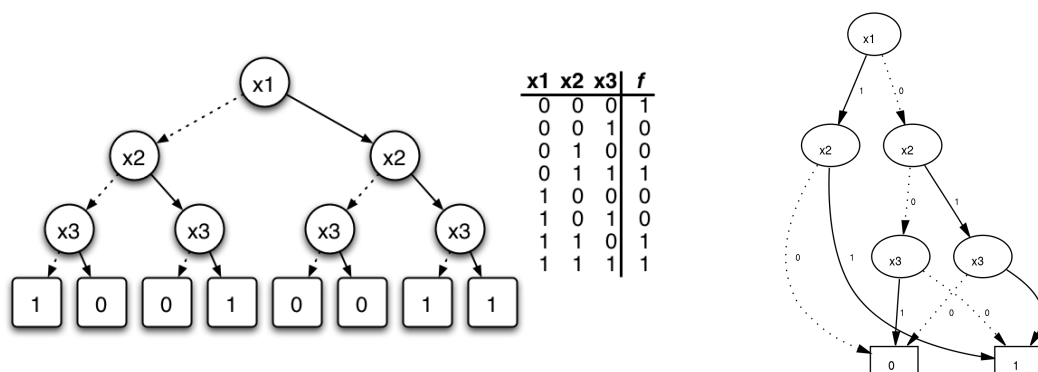


Les diagrammes de décision binaires (BDD pour *Binary Decision Diagram*) sont une structure de données qui permet de représenter de façon compacte les relations entre variables booléennes. Ils ont été introduits par Randal E. Bryant et sont devenus incontournables pour les outils de vérification.

**Exemple 11** Considérons la fonction

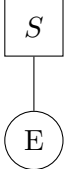
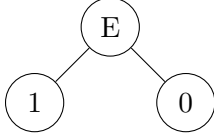
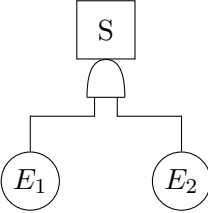
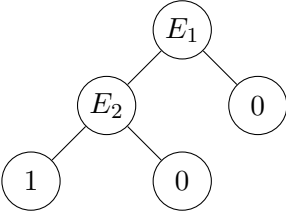
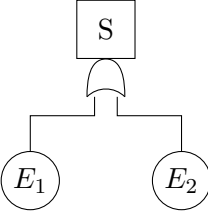
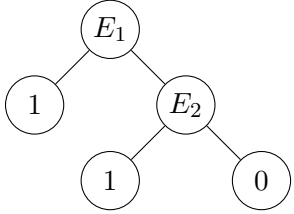
$$f(x_1, x_2, x_3) = \bar{x}_1 \bar{x}_2 \bar{x}_3 + x_1 x_2 + x_2 x_3$$

Le diagramme de décision binaire et la table de vérité sont représentés ci-dessous.

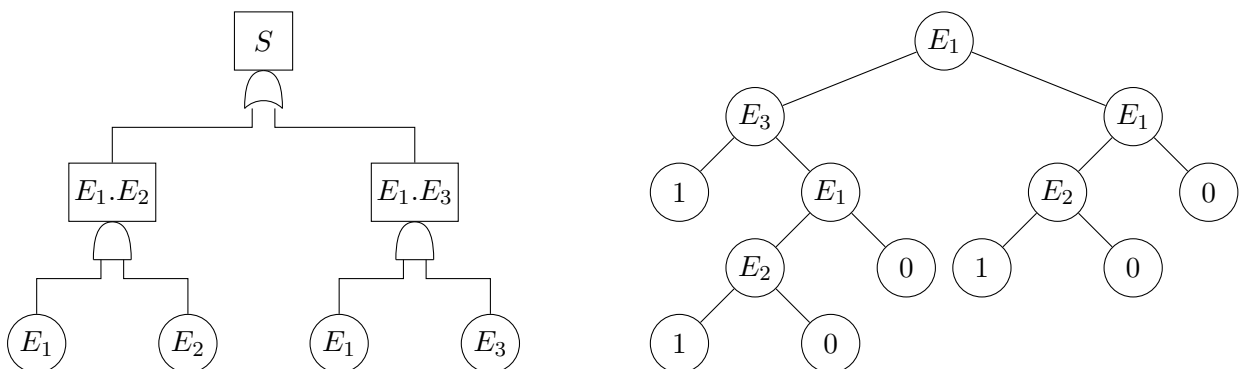




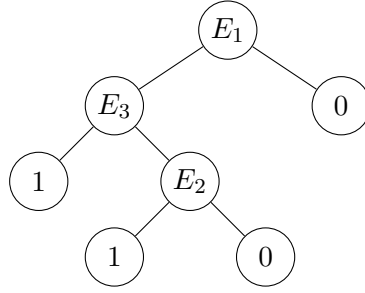
On en déduit le codage des arbres de défaillances.

Arbre de défaillance	Diagramme de décision binaire
	
	
	

On peut ensuite appliquer des règles de simplification quand une variable apparaît plusieurs fois dans une même branche. Considérons l'exemple ci-dessous :

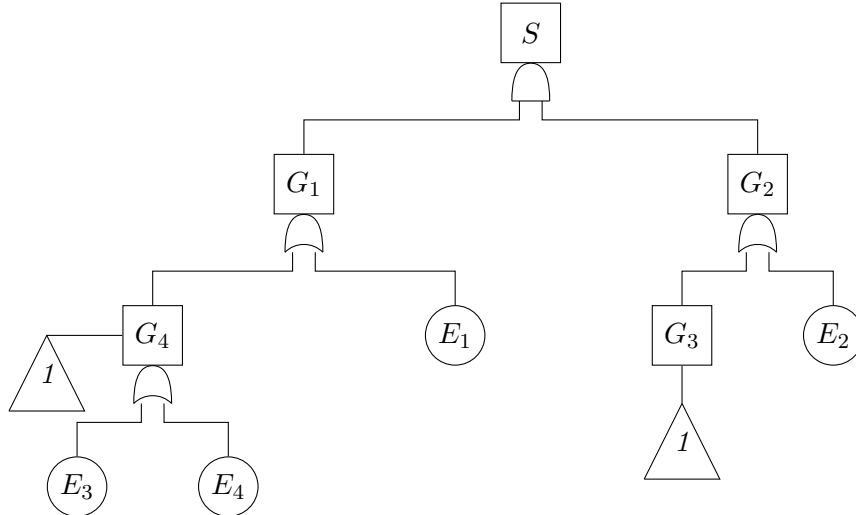


Le BDD peut se simplifier en  $E_1$  dans les deux branches et on obtient :



Grâce à cette représentation sous forme de diagramme de décision binaire, on trouve rapidement les coupes minimales d'un arbre, il s'agit en effet d'une branche menant à un 1. Dans l'exemple, il y a deux coupes minimales  $E_1.E_2$  ou  $E_1.E_3$ .

**Exercice 22** Donner le diagramme de décision binaire associé à l'arbre suivant ainsi que l'ensemble des coupes minimales.



## 4 Modèles à états transitions

### 4.1 Chaînes de Markov

La Méthode de l'Espace d'Etat (MEE) a été développée pour l'analyse de sûreté de fonctionnement de système réparable. Les arbres de défaillances, vus dans le chapitre précédent, permettent de bonnes descriptions statiques de système mais ne prennent pas en compte les reconfigurations, comme les réparations. Les premières utilisations des processus stochastiques dans les années 50 utilisaient des processus markoviens; des généralisations ont ensuite été faites. Dans cette partie nous nous concentrons sur les processus markoviens. Andrei Markov a publié ses premiers résultats en 1906, qui ont ensuite été généralisés à un espace d'états infini dénombrable par Andrei Kolmogorov en 1936.

Un processus stochastique est un ensemble de variables aléatoires  $(X_t)_{t \geq 0}$  à valeurs dans l'ensemble des observations. Un processus est markovien si la probabilité de passage de l'étape présente à la suivante ne dépend pas du passé, i.e.

$$P(X_t \in A \mid X_{t_n} \in A_n, \dots, X_1 \in A_1) = P(X_t \in A \mid X_{t_n} \in A_n)$$

**Exemple 12 (Exemple Wikipedia : Doudou le hamster)** Cet exemple est à temps discret. Doudou le hamster paresseux ne connaît que 3 endroits dans sa cage : les copeaux où il dort, la mangeoire où il mange et la roue où il fait de l'exercice. Ses journées sont assez semblables les unes aux autres, et son activité se représente aisément par une chaîne de Markov. Toutes les minutes, il peut soit changer d'activité, soit continuer celle qu'il était en train de faire. L'appellation processus sans mémoire n'est pas du tout exagérée pour parler de Doudou.

- Quand il dort, il a 9 chances sur 10 de ne pas se réveiller la minute suivante.
- Quand il se réveille, il y a 1 chance sur 2 qu'il aille manger et 1 chance sur 2 qu'il parte faire de l'exercice.
- Le repas ne dure qu'une minute, après il fait autre chose.
- Après avoir mangé, il y a 3 chances sur 10 qu'il parte courir dans sa roue, mais surtout 7 chances sur 10 qu'il retourne dormir.
- Courir est fatigant ; il a donc 80 % de chance de retourner dormir au bout d'une minute.
- Sinon il continue en oubliant qu'il est déjà un peu fatigué.

La chaîne de Markov associée au comportement du hamster est donnée dans la figure 16 et la version complète est représentée dans la figure 17.

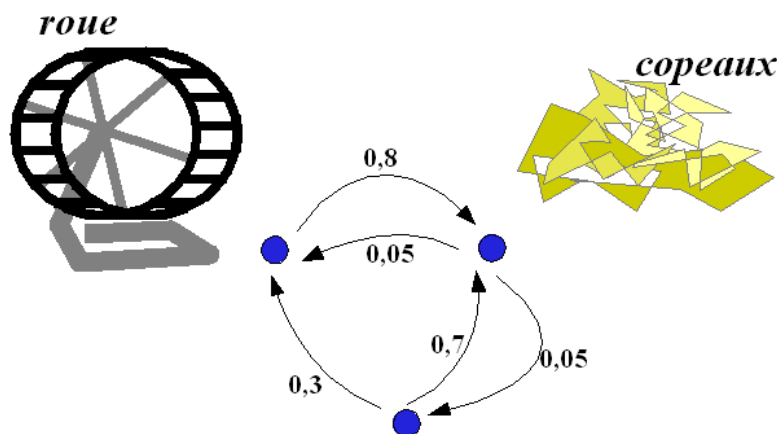


FIGURE 16 – Chaîne de Markov associée à Doudou le hamster

On peut ensuite décrire la matrice de simulation associée à la chaîne de Markov où les lignes et les colonnes correspondent dans l'ordre aux états dormir, manger, courir :

$$T = \begin{bmatrix} 0,9 & 0,05 & 0,05 \\ 0,7 & 0 & 0,3 \\ 0,8 & 0 & 0,2 \end{bmatrix}$$

Il est alors possible de simuler le comportement du hamster sachant qu'initialement il dort :  $\mathbf{x}^{(0)} = [1 \ 0 \ 0]$ . Au bout d'une minute, on peut prédire qu'il y a 90 % de chances que Doudou dorme encore, 5 % qu'il mange et 5 % qu'il court :  $\mathbf{x}^{(1)} = \mathbf{x}^{(0)}T = [0,9 \ 0,05 \ 0,05]$ . Au bout de deux minutes,  $\mathbf{x}^{(2)} = \mathbf{x}^{(1)}T = \mathbf{x}^{(0)}T^2 = [0,885 \ 0,045 \ 0,07]$  et ainsi de suite.

**Exercice 23 (Petitot)** Chaque année à Noël, les mangeurs de chocolat adoptent un type de chocolat, pour une durée d'un an renouvelable. Un sondage effectué sur un échantillon représentatif

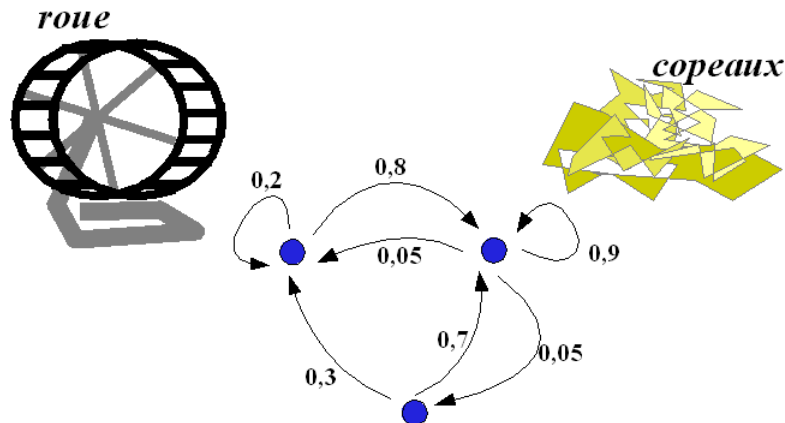


FIGURE 17 – Chaîne de Markov complète associée à Doudou le hamster

de cette population a donné les chiffres suivants : parmi les mangeurs de chocolat noir, 65% sont fidèles à leur choix, tandis que 35% préfèrent essayer le chocolat au lait. De même, parmi les mangeurs de chocolat au lait, 70% restent fidèles et 30% changent pour le noir. Initialement, il y avait 50% de mangeurs de chocolat noir et 50% de mangeurs de chocolat au lait.

Quelle sera la tendance au bout d'un an ? 10 ans ? Modélisez le problème avec une chaîne de Markov en GRIF.

#### 4.1.1 Construction d'un modèle

Considérons un système composé de  $n$  composants, chaque composant ayant un nombre fini d'états de fonctionnement et de panne ; ce système est supposé réparable et chaque composant est réparé après constatation de la panne. Le système est donc composé :

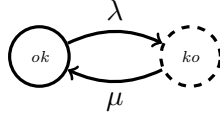
- des états de fonctionnement : un état de bon fonctionnement où tous les composants fonctionnent, et des états où certains composants sont en panne mais le système reste fonctionnel,
- des états de pannes : où suffisamment de composants sont en panne pour affecter le système globale.

La construction du modèle se fait en 3 étapes :

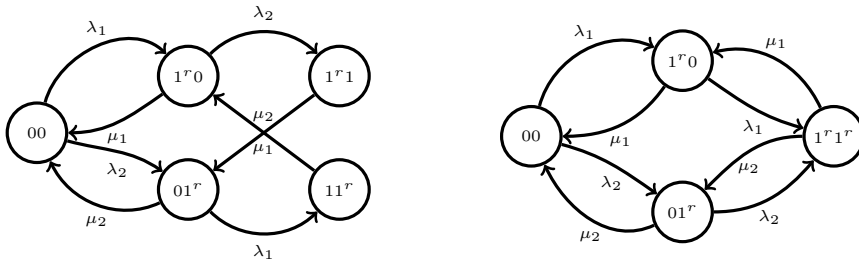
1. recensement de tous les états du système. Si chaque composant a 2 états (ok ou panne) et si le système à  $n$  composants, le nombre maximal d'états est  $2^n$ . Au cours de la vie du système, des états de panne peuvent apparaître à la suite de défaillance ou disparaître à la suite de réparation ;
2. recensement de toutes les transitions possibles entre ces différents états et l'identification de toutes les causes de ces transitions. Les causes des transitions sont généralement des défaillances des composants ou la réparation de composants ;
3. calcul des probabilités de se trouver dans les différents états au cours d'une période de vie du système, calcul des temps moyens (MTTF, MTBF, MTTR ...)

**Exemple 13 (Composant unique)** Pour un système à un composant qui n'a qu'un mode de défaillance panne, on obtient l'automate décrit ci-dessous. Initialement, on est dans l'état ok, à

tout instant le composant peut tomber en panne avec le taux de défaillance instantané  $\lambda$  puis se faire réparer avec le taux de réparation  $\mu$ .



**Exemple 14 (Deux composants)** On considère un système constitué de deux composants et on suppose qu'une seule panne à la fois peut survenir. Chaque composant a trois états possibles : l'état 0 correspond au bon fonctionnement, l'état  $1^r$  correspond au fait que le composant est indisponible mais en cours de réparation et 1 correspond à l'état de panne. L'automate associé au système est la combinaison de tous ces états, par exemple 00 est le bon fonctionnement du système alors que 11 correspond à l'arrêt total. A gauche, la chaîne de Markov avec un seul réparateur qui intervient dès la détection d'une panne ; et à droite deux réparateurs sont à disposition.



Les graphes des états avec deux réparateurs pour un système série à deux composants et un système en redondance active sont donnés ci-dessous.

Diagramme de fiabilité	chaîne de Markov
	<div style="display: flex; justify-content: space-around;"> <div style="border-right: 1px solid black; padding-right: 10px;"> <p>Fonctionnement correct</p> </div> <div style="padding-left: 10px;"> <p>Système défaillant</p> </div> </div>
	<div style="display: flex; justify-content: space-around;"> <div style="border-right: 1px solid black; padding-right: 10px;"> <p>Correct functioning</p> </div> <div style="padding-left: 10px;"> <p>Failed system</p> </div> </div>

**Exemple 15 (Signoret)** On considère un système composé de 2 composants redondants. Un seul réparateur est disponible et doit réparer en priorité  $E_1$ . La chaîne de Markov est

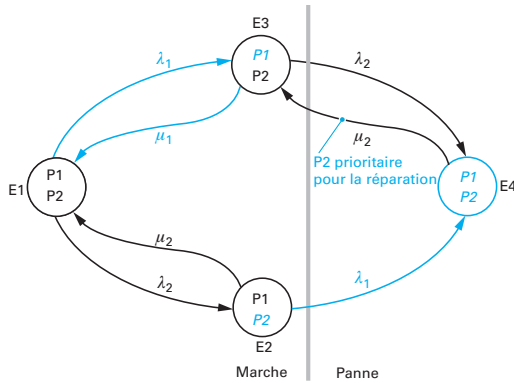
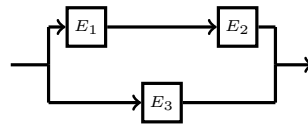


Tableau 1 – Disponibilité en fonction du temps (1)				
Paramètres	$\lambda_1$	1, 0E - 05	$\mu_1$	1, 0E - 01
	$\lambda_2$	5, 0E - 04	$\mu_2$	4, 2E - 02
Temps (h)	États			
	1	2	3	4
0	1, 000E + 00	0, 000E + 00	0, 000E + 00	0, 000E + 00
20	9, 932E - 01	6, 735E - 03	8, 595E - 05	1, 000E - 06
40	9, 903E - 01	9, 613E - 03	9, 758E - 05	2, 229E - 06
60	9, 891E - 01	1, 084E - 02	9, 934E - 05	3, 038E - 06
80	9, 885E - 01	1, 137E - 02	9, 972E - 05	3, 498E - 06
100	9, 883E - 01	1, 159E - 02	9, 986E - 05	3, 743E - 06
120	9, 882E - 01	1, 169E - 02	9, 992E - 05	3, 869E - 06
140	9, 882E - 01	1, 173E - 02	9, 996E - 05	3, 931E - 06
160	9, 881E - 01	1, 175E - 02	9, 997E - 05	3, 962E - 06
180	9, 881E - 01	1, 176E - 02	9, 998E - 05	3, 977E - 06
200	9, 881E - 01	1, 176E - 02	9, 999E - 05	3, 984E - 06
300	9, 881E - 01	1, 176E - 02	9, 999E - 05	3, 990E - 06
400	9, 881E - 01	1, 176E - 02	9, 999E - 05	3, 991E - 06

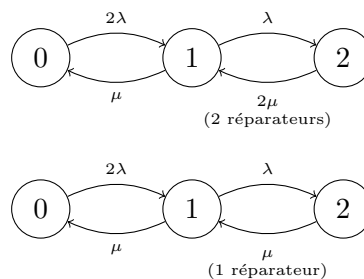
(1) On rappelle que la notation E-05, par exemple, signifie  $\times 10^{-5}$ .

**Exercice 24** Donner la chaîne de Markov associée au système suivant.



Supposons que les composants ne sont pas réparables et que  $\lambda_i = 10^{-5}$ , comparer les simulations Grif avec diagramme de fiabilité et chaîne de Markov.

**Simplification** Quand on modélise un système avec une chaîne de Markov, on rencontre le problème de l'explosion combinatoire des états puisque la chaîne a a priori  $2^n$  états pour un système de  $n$  éléments à 2 états. Il est néanmoins possible de réduire la taille de la chaîne en agglomérant des états. Cela suppose que les composants soient identiques avec les mêmes taux de défaillance et de réparation. Dans le cas d'un système à deux composants actifs, on peut simplifier les chaînes vues ci-dessus de la sorte. L'état  $i$  correspond à l'ensemble des états où il y a  $i$  pannes.



**Exercice 25** Donner la forme générale de la chaîne de Markov d'un système de  $n$  composants en redondance active avec  $k$  réparateurs.

## 4.2 Evaluation de la fiabilité, de la disponibilité et du MTTF

Pour calculer les attributs de sûreté de fonctionnement sur un système modélisé par une chaîne de Markov, il faut calculer la probabilité d'être dans un état à un instant  $t$ .

**Equations d'états du système** La transition entre un état  $i$  et  $j$  se fait avec un taux de transition instantané  $\rho_{i,j}$  (viennent des comportements physiques).  $P_i$  est la probabilité d'être dans l'état  $i$ . L'équation d'état d'un graphe de Markov :

$$P_i(t + dt) = P(\text{ dans l'état } i \text{ à } t \text{ et durant } [t, t + dt]) + \sum_{j \neq i} P(\text{ dans l'état } j \text{ à } t \text{ dans l'état } i \text{ à } t + dt)$$

Comme  $\rho_{i,j}(t)dt$  est la probabilité de passer de  $i$  à  $j$  entre  $t$  et  $t + dt$ , on a :

$$P_i(t + dt) = P_i(t)[1 - \sum_{j \neq i} \rho_{i,j}(t)dt] + \sum_{j \neq i} P_j(t)\rho_{j,i}(t)dt$$

On en déduit :

$$\frac{P_i(t + dt) - P_i(t)}{dt} = -P_i(t) \sum_{j \neq i} \rho_{i,j}(t) + \sum_{j \neq i} P_j(t)\rho_{j,i}(t)$$

On obtient un système d'équations de Chapman-Kolmogorov du premier ordre :

$$\frac{dP(t)}{dt} = M(t)P(t)$$

où  $M$  est la matrice des taux de transition  $M_{ij} = \rho_{j,i}$  avec  $\rho_{i,i} = -\sum_{j \neq i} \rho_{i,j}$ .

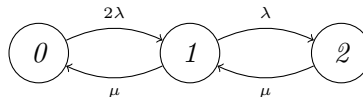
On en déduit directement la disponibilité du système

$$A(t) = \sum_{i \in \text{états de fonctionnement}} P_i(t)$$

**Exercice 26** *Ecrire l'équation d'état de la chaîne de Markov d'un système à un composant, dont les taux de défaillance et de réparation sont constants, et la résoudre. En déduire la disponibilité.*

**Résolution** La résolution de l'équation d'état, lorsqu'on connaît la distribution initiale  $P_i(0)$ , peut être effectuée par des méthodes de résolution explicite à l'aide de la transformée de Laplace, de discrétisation ou de calcul de valeurs propres de la matrice (en effet, si  $M$  est constante on trouve  $P(t) = e^{Mt}P(0)$ ). La méthode des valeurs propres posent des problèmes pour les systèmes très fiables. En effet, la plus grande valeur propre  $-\beta_1$  est beaucoup plus petite en valeur absolue que les autres et la précision risque d'être mauvaise lorsque le nombre d'état est assez grand. Ceci est d'autant plus gênant que c'est cette valeur propre qui détermine le comportement du système quand  $t$  est grand. Du fait de ces limites, cette méthode est peu utilisée en pratique. La résolution d'un système linéaire d'équations différentielles du premier ordre est classique en analyse numérique et de nombreux programmes sont disponibles.

**Exemple 16** *On illustre le calcul de la résolution du graphe d'état pour un système en redondance active à un réparateur. La chaîne a déjà été décrite, il s'agit de*



L'équation d'états associée est alors

$$\begin{cases} \frac{dP_1(t)}{dt} = -2\lambda P_1(t) + \mu P_2(t) \\ \frac{dP_2(t)}{dt} = 2\lambda P_1(t) - (\lambda + \mu)P_2(t) + \mu P_3(t) \\ \frac{dP_3(t)}{dt} = \lambda P_2(t) - \mu P_3(t) \end{cases}$$

On suppose  $P_1(0) = 1$  et  $P_2(0) = P_3(0) = 0$ . Pour résoudre ce système, on utilise transformée de Laplace. On note  $L(P_i) = L_i$  et on rappelle

$$\begin{aligned} L(P_i)(s) &= \int_0^\infty e^{-st} P_i(t) dt \\ L\left(\frac{dP_i(t)}{dt}\right)(s) &= sL_i(s) - P_i(0) \end{aligned}$$

On trouve alors

$$\begin{cases} s.L_1(s) - 1 = -2\lambda L_1(s) + \mu L_2(s) \\ s.L_2(s) = 2\lambda L_1(s) - (\lambda + \mu)L_2(s) + \mu L_3(s) \\ s.L_3(s) = \lambda L_2(s) - \mu L_3(s) \end{cases}$$

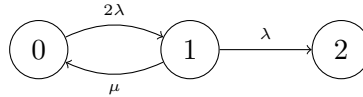
d'où

$$\begin{cases} L_1(s) = \frac{s^2 + (2\mu + \lambda)s + \mu^2}{s.f(s)} \\ s.L_2(s) = 2\frac{\lambda(s + \mu)}{s.f(s)} \\ s.L_3(s) = 2\frac{\lambda^2}{s.f(s)} \end{cases}$$

avec  $f(s) = s^2 + s(2\mu + 3\lambda) + (2\lambda^2 + 2\lambda\mu + \mu^2)$  On calcule ensuite les transformées de Laplace inverses en utilisant la formule  $L^{-1}\left(\frac{1}{s + \alpha}\right) = e^{-\alpha t}$  et on trouve

$$\begin{cases} P_1(t) = \frac{\alpha}{2\lambda^2} \left(s_1 + \lambda + 2\mu + \frac{\mu^2}{s_1}\right) e^{s_1 t} + \frac{\beta}{2\lambda^2} \left(s_2 + \lambda + 2\mu + \frac{\mu^2}{s_2}\right) e^{s_2 t} + \frac{\mu^2 \gamma}{2\lambda^2} \\ P_2(t) = \frac{\alpha}{\lambda} \left(1 + \frac{\mu}{s_1}\right) e^{s_1 t} + \frac{\beta}{\lambda} \left(1 + \frac{\mu}{s_2}\right) e^{s_2 t} + \frac{\mu \gamma}{\lambda} \\ P_3(t) = \frac{\alpha}{s_1} e^{s_1 t} + \frac{\beta}{s_2} e^{s_2 t} + \gamma \end{cases}$$

**Calcul de la fiabilité et du MTTF** Pour calculer la fiabilité d'un système représenté sous forme d'une chaîne de Markov, il faut modifier la chaîne de façon à éliminer toutes les transitions de réparation d'un état de panne vers un état de fonctionnement. Les états de panne deviennent alors absorbants. Ainsi la nouvelle chaîne de Markov associée à un système en redondance active à 2 composants devient



On trouve alors la nouvelle équation d'état de cette chaîne  $\frac{dP(t)}{dt} = M'(t)P(t)$ . Ce qui permet ensuite de calculer la fiabilité du système :

$$R(t) = \sum_{i \in \text{états de fonctionnement}} P_i(t)$$

Comme seuls les états de fonctionnement contribuent au calcul de la fiabilité et que les états de panne sont absorbants, il suffit de résoudre le système d'équation sur les états de fonctionnement.



**Exercice 27** On considère les systèmes à deux composants non réparables série ou redondance active. Calculer la fiabilité et le MTFF des systèmes. Retrouve-t-on les résultats de la partie 2.4.3 page 23 ?

**Exercice 28** Calculer la fiabilité du système en redondance active à 2 composants.

**Maintenabilité** On peut calculer l'immaintenabilité  $\bar{M}(t) = 1 - M(t)$  directement à partir de la chaîne de Markov précédente. En effet, c'est la probabilité que le système qui est initialement en panne ne puisse pas être réparé. Il faut donc considérer la chaîne de Markov où on supprime toutes les transitions des états de marche vers les états de panne.

$$\bar{M}(t) = \sum_{i \in \text{états de panne}} P_i(t)$$

**Exercice 29** Calculer la maintenabilité du système en redondance active à 2 composants.

### 4.3 Réseaux de Petri stochastiques

Carl Adam Petri a proposé dans ses travaux de thèse (1962) un nouvel outil dédiés à la modélisation des automates. Dans les années 70, S. Natkin et G. Florin ont proposé des réseaux de Petri markoviens. Dans les années 80, J.P. Signoret et A. Leroy ont utilisé les réseaux de Petri comme modèles comportementales pour réaliser des simulations de Monte-Carlo (pour les grands systèmes). De nombreuses recherches sont en cours sur le sujet, notamment LAAS, MOCA-RP, Marsan et al., Trivedi et al.(USA),

Les réseaux de Petri sont un bon outil pour modéliser le comportement dysfonctionnel d'un système. On appréhende plus aisément les différentes pannes et l'impact sur le système. Pour rappel, un réseau de Petri est un graphe orienté avec deux types de nœuds : les places (états ou conditions) représentées par des cercles et les transitions (ou événements) symbolisées par des barres. Ces nœuds sont connectés entre eux par des arcs orientés à des places aux transitions (arcs amont) et des transitions aux places (arcs aval) exclusivement. La circulation de jetons (marqueurs indivisibles), symbolisant la présence à un instant donné d'une information ou d'une initialisation particulière aux places où ils résident, permet la modélisation dynamique du comportement système (aussi bien désiré que redouté) au sein du réseau. Un réseau de Petri stochastique (ou dans notre cas fiabiliste) est un réseau de Petri étendu tel qu'on associe à chaque transition une durée de franchissement aléatoire ou déterministe (nulle ou non). Si la durée déterministe est 0, on parle de transitions immédiates. Une présentation complète se trouve dans le livre [3].

#### 4.3.1 Modélisation des systèmes avec des réseaux de Petri

réseaux de Petri sont un bon outil pour modéliser les comportements dysfonctionnels.

**Système avec un composant unique** Supposons que le composant à un taux de défaillance constant  $\lambda$  et un taux de réparation constant  $\mu$ . En plus des deux états standard (ok et réparation), on peut également décrire le moment où le système tombe en panne et attend d'être réparé. Le réseau de Petri est montré figure 18, partie gauche. Sur ce premier modèle, l'attente et le réparateur sont représentés par un délai constant. Dans le modèle de droite, on représente le réparateur par un jeton dans une place. Si le réparateur est disponible, il intervient de suite pour réparer le système.

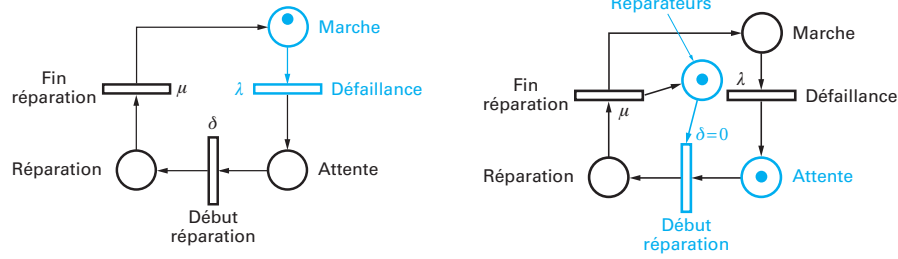


FIGURE 18 – Modèle pour un composant unique réparable

**Exercice 30** Donner les réseaux de Petri associés à

1. un système à deux composants réparables et deux réparateurs ;
2. un système à deux composants non réparables.

Les états de fonctionnement et de pannes sont alors des ensembles de marquages du réseau de Petri. Dans le cas de la figure 18 partie droite, si le système contient les deux composants en série, un seul marquage représente les états de fonctionnement, s'il s'agit d'une redondance active 5 marquages constituent la fonctionnement.

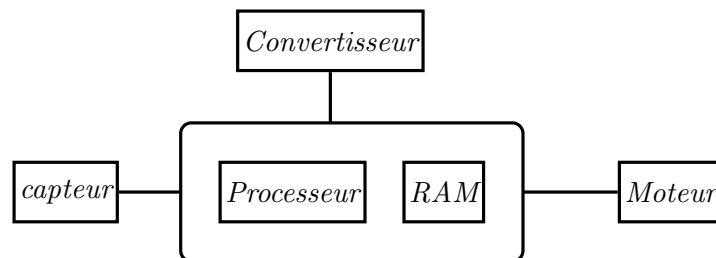
	série	redondance
Marche <sub>1</sub>	1	1 0 0 1 1
Attente <sub>1</sub>	0	0 1 0 0 0
Réparation <sub>1</sub>	0	0 0 1 0 0
Réparateurs	1	1 0 0 0 0
Marche <sub>2</sub>	1	1 1 1 0 0
Attente <sub>2</sub>	0	0 0 0 1 0
Réparation <sub>2</sub>	0	0 0 0 0 1

**Génération de chaînes de Markov** Si on représente le graphe des états associés, système de transition entre marquage on retombe sur la chaîne de Markov représentant le système.

**Exercice 31** Donner les chaînes de Markov obtenues à partir des réseaux de Petri associés à

1. un système à deux composants réparables et deux réparateurs ;
2. un système à deux composants non réparables.

**Exemple 17** On donne un exemple complet d'un système mécatronique étudié dans [10]. Le système étudié est l'ouverture automatique d'une portière : le conducteur du véhicule en possession de la clé est détecté par le capteur du système ce qui déclenche l'activation d'un convertisseur qui alimente le reste du système. L'information est reçue puis traitée par un processeur muni d'un RAM qui déclenche un moteur qui ouvrira la portière.



Les auteurs ont modélisé le système en un réseau de Petri : chaque composant a deux états correct ou défaillance, tout équipement arrêté ne peut tomber en panne. Le système sous forme de réseau de Petri est modélisé dans la figure 19. 12 marquages sont accessibles et le graphe de marquage du réseau de Petri est donné dans la figure 20.

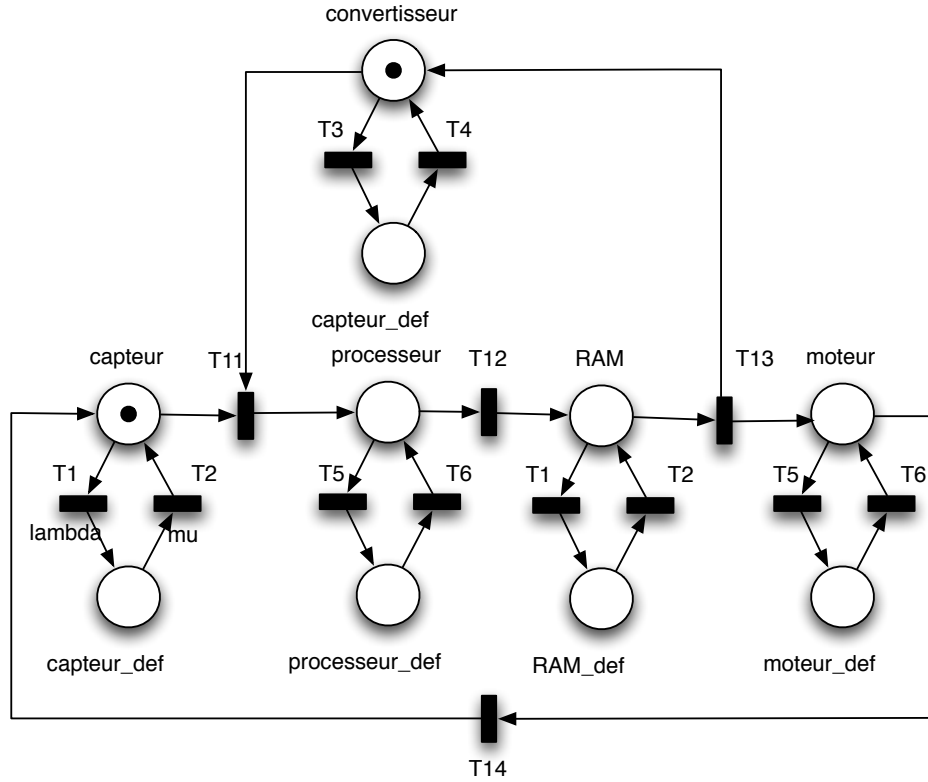


FIGURE 19 – Réseau de Petri du système

**Simulation de Monte-Carlo** L'exploitation quantitative d'un réseau de Petri se fait généralement, soit en traitant le modèle markovien issu du graphe des marquages accessibles du réseau de Petri, chaque marquage non évanescent correspondant à un état de la chaîne de Markov, soit en animant par simulation de Monte-Carlo directement le modèle réseau de Petri et non son graphe d'accessibilité.

La simulation se montre relativement insensible à la taille du modèle à animer, elle est donc utilisée pour traiter des systèmes de grandes tailles. L'idée est de réaliser une marche aléatoire à travers l'espace d'états du système modélisé. Le principe des simulations de Monte-Carlo est de jouer un certain nombre de scénarii d'évolution du réseau de Petri en tirant pseudo-aléatoirement les délais associés aux transitions et en faisant des statistiques sur des valeurs ayant un intérêt telles que le nombre de tirs d'une transition, le temps moyen de séjour dans une place ... Un scénario redouté est une liste d'événements qui conduisent d'un état de fonctionnement normal à un état redouté avec une relation d'ordre partiel entre ces événements. Si  $f(t)$  est la loi de probabilité associée à une transition, pour obtenir un délai pseudo-aléatoire  $\delta$  on choisit un nombre  $z$  entre 0 et 1 (suivant une loi uniforme) et on prend  $\delta = f^{-1}(z)$ . Lorsque  $f$  n'est pas

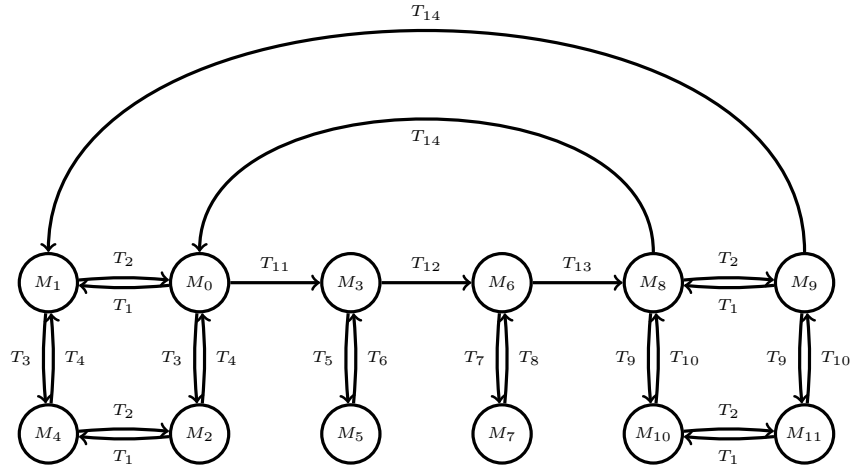


FIGURE 20 – Graphe de marquage du système

inversible, on calcule la valeur numériquement.

Les inconvénients de cette approche sont la précision variable qui dépend du nombre de simulations effectuées et le temps de traitement qui est potentiellement long et peut difficilement s'appliquer aux événements rares.

Les deux structures principales sont :

**Horloge de simulation :** elle enregistre le temps de mission d'une simulation.

**Liste d'événements :** les événements sont maintenus dans un ordre chronologique. Leurs dates d'occurrence sont générées aléatoirement en fonction de leurs distributions probabilistes.

La simulation se déroule alors de la manière suivante :

- Choisir l'événement le plus récent pour l'exécution.
- Exécuter l'événement et avancer l'horloge.
- Mettre à jour les données (liste d'événements, variables d'observation,...)
- Arrêter si une des deux conditions suivantes se vérifient :
  - le temps de mission est dépassé
  - ou il n'y a plus d'événement à exécuter.

## 4.4 AltaRica

Le langage Altarica a été conçu pour spécifier formellement le comportement de systèmes en présence de fautes et exploiter ces spécifications à l'aide d'outils complémentaires : simulateur symbolique, model-checker ou générateur d'arbre de défaillances. AltaRica est un langage de description de système basé sur des automates à contraintes. La première définition du langage est apparue en 1996 au Labri à Bordeaux conjointement avec des industriels (comme Dassault) et le premier atelier a été mis en place en 2001.

### 4.4.1 Modélisation des système avec AltaRica

Le langage Altarica a été conçu au LaBri pour spécifier formellement le comportement de systèmes en présence de fautes et exploiter une unique spécification à l'aide d'outils complémentaires :

simulateur symbolique, model-checker, générateur d'arbre de défaillance, simulation stochastique. Le simulateur permet d'animer des spécifications comportementales non déterministes. Après avoir brièvement présenté le langage AltaRica, nous donnons une méthodologie pour modéliser des systèmes d'un point de vue sûreté de fonctionnement.

```

Node block
flow I,A,R : bool : in ;
O : bool : out ;
state S : bool ;
event fail ;
trans S=true |- fail -> S := false ;
assert O = I and S and R and A ;
init S := true ;
law extern <event fail>=«constant 1e-4»
edon

```

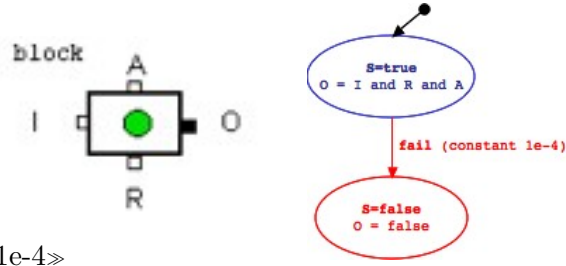


FIGURE 21 – Structure d'un nœud AltaRica

La figure 4.4.1 décrit la modélisation d'un composant en AltaRica. Un nœud est défini par son nom (ici *block*), des variables internes seulement visibles dans ce nœud introduits par la clause *state* (dans l'exemple il n'y a qu'une variable *s*), des variables externes appelées *flow* visibles par d'autres nœuds (dans l'exemple il y a *O* qui est émise et *I*, *A*, *R* qui sont lue par le nœud), des transitions (gardes, événements, affectations) décrivant après le mot clé *trans* les différents comportements du composant (dans l'exemple, une seule transition est décrite spécifiant que si *s* a pour valeur *true* et que l'événement *fail* survient, alors *s* prend pour valeur *false*), des contraintes définissant les configurations autorisées entre les variables. Ces assertions mettent principalement en relation les variables de flux avec les variables d'état. Dans l'exemple, la variable de flux *O* prend pour valeur *correct* si *s* et *I* sont corrects et *lost* sinon, ce qui traduit bien la notion d'erreur. Enfin, une valeur d'initialisation peut être déclarée. On suppose dans l'exemple que le fonctionnement est *correct* initialement. La sémantique d'un nœud AltaRica est donnée par un *système de transition interfacé* qui est un système de transition étendu avec des variables externes. La sémantique du nœud *fonction* est représentée dans la partie droite de la figure 4.4.1.

Formellement un nœud AltaRica, sans hiérarchie, est défini par un automate de mode  $\mathcal{M} = \langle D, S, F^{in}, F^{out}, dom, \Sigma, \delta, \sigma, I \rangle$  avec :

- $D$  est un domaine fini de valeurs des variables,
- $S$ ,  $F^{in}$  et  $F^{out}$  sont des ensembles finis de variables appelées respectivement variable d'état, variable de flux d'entrée et de sortie. Ces ensembles sont tous 2 à 2 disjoints. Dans la suite, on note  $V = S \cup F^{in} \cup F^{out}$ ,
- $dom : V \rightarrow 2^D$  est une fonction qui associe à une variable son domaine telle que  $\forall v \in V, dom(v) \neq \emptyset$ ,
- $\Sigma$  est un ensemble fini d'événements,
- $\delta$  est une fonction partielle appelée *transition* :  $dom(S) \times dom(F^{in}) \times \Sigma \rightarrow dom(S)$ . On appelle  $dom(S) \times dom(F^{in})$  la garde de la transition,
- $\sigma$  est une fonction totale appelée *assertion* :  $dom(S) \times dom(F^{in}) \rightarrow dom(F^{out})$ ,
- $I \subset dom(S)$  décrit les conditions initiales.

Pour définir des modes de défaillance plus complexes que simplement la perte, on peut utiliser des domaines abstraits en AltaRica. Par exemple, une valeur calculée par un calculateur peut

être *correct*, *erroneous* si elle est calculée mais le résultat est faux (par exemple si les entrées sont fausses ou si le logiciel a un bug) ou *lost* lorsqu'aucune sortie n'est émise (si le calculateur est déconnecté par exemple). On note alors  $\text{domain data} = \{\text{correct}, \text{lost}, \text{erroneous}\}$ .

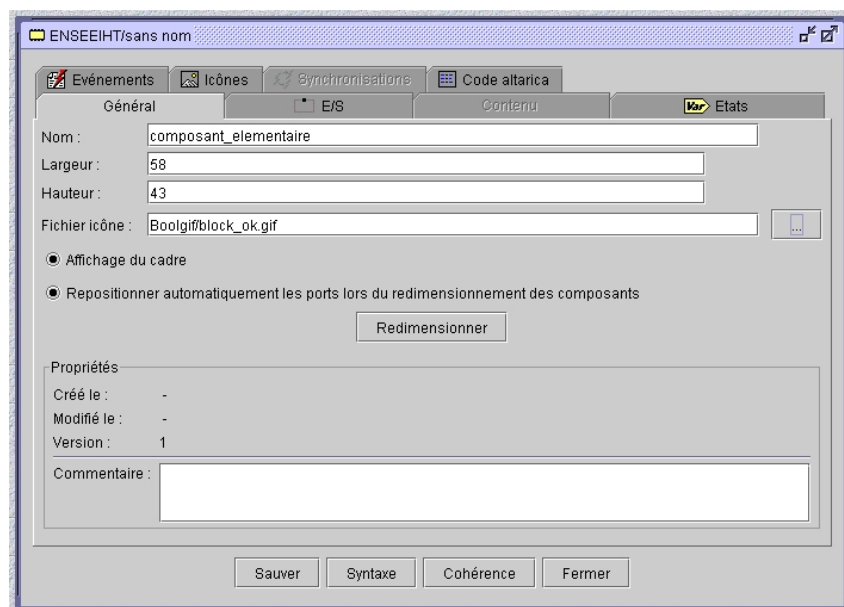
Un système est la combinaison de composants et de sous-systèmes. On peut alors définir un nœud AltaRica hiérarchique faisant appel à un ensemble de composants et/ou système. Si on souhaite modéliser le système de deux composants en série :

```
node serie
  state s:bool;
  sub C1, C2: block
  assert C1.out = C2.in;
    s = C2.out;
```

#### 4.4.2 Codage avec l'outil OCAS

Pour lancer l'outil, aller dans le menu N7 et lancer Cecilia.bat, choisir OCAS, mettre le login : admin et mot de passe : admin.

**Définition des composants** L'idée est de définir une librairie de composants puis de spécifier des systèmes complets utilisant les éléments de la librairie. Téléchargez sur l'url <http://www.cert.fr/anglais/> le fichier *icônes.zip*. Nous allons dans une premier définir un composant. Allez dans l'onglet *Composant* et créez une nouvelle famille *TP\_N7*. Créez ensuite un nouveau modèle dans cette famille *composant\_élémentaire*. Une fenêtre s'ouvre alors comme illustré ci-dessous. Dans l'onglet général, mettez le nom du composant *composant\_elementaire*. Ouvrez le fichier icône et importer les figures en gif contenus dans l'archive *icônes.zip*. Choisissez alors *block\_ok.gif*.



Dans l'onglet E/S, mettez une entrée *I* et une sortie *O* de type enum  $\{ok, lost\}$ . Positionner sur le schéma les entrées/sorties. Dans l'onglet *event*, mettez un événement *fail*. Dans l'onglet *state* mettez un état *S* de type enum  $\{ok, lost\}$ , dont la valeur initiale est *ok*. Dans l'onglet *icônes*, ajoutez les deux icônes *block\_ok.gif* et *block\_nok.gif*. Dans l'onglet *Altairica Code* mettez

```

trans
S=ok |- fail -> S:=lost;

assert
0=case {S=ok : I,
else lost};

icone = case {S=ok : 1,
else 2};;

```

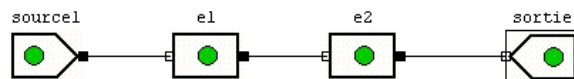
Tous les onglets de l'éditeur de modèle contiennent quatre boutons communs :

- Sauver : permet de sauvegarder les informations saisies,
- Syntaxe : permet d'effectuer un contrôle de syntaxe sur le code AltaRica
- Cohérence : permet d'effectuer un contrôle de cohérence sur le code AltaRica,
- Fermer : permet de fermer l'éditeur de modèle.

Cliquez sur syntaxe pour vérifier la syntaxe et sur cohérence pour vérifier la cohérence.

Créez ensuite un composant *source* avec une sortie qui est toujours *ok*. Choisissez l'icône *source.gif*. Créez un composant *sortie* avec une entrée et une sortie tels que *sortie=entree*. Choisissez les icônes *sortie\_ok.gif* et *sortie\_nok.gif*.

**Définition d'un système série** A partir de cette première librairie, nous pouvons concevoir des systèmes séries et parallèles. Dans l'onglet système, créez un projet *modeles\_AR* puis un système *modele\_serie*. Une fenêtre s'ouvre alors. Retourner dans l'onglet composant drag and dropper 2 composants élémentaires, une source et une sortie. Tracer les liens de sorte à obtenir un système en série comme dans la figure.



Pour lancer la simulation cliquez sur le feu vert et utilisez l'insecte pour tirer les événements de panne. Générez ensuite les coupes avec les séquenceurs et jouez-les ensuite. Nous devons retrouver les mêmes coupes minimales qu'avec les arbres de défaillance.

**Exercice 32** *Ecrivez un système en redondance active non réparable et réparable. N'oubliez pas d'étendre la librairie si besoin.*

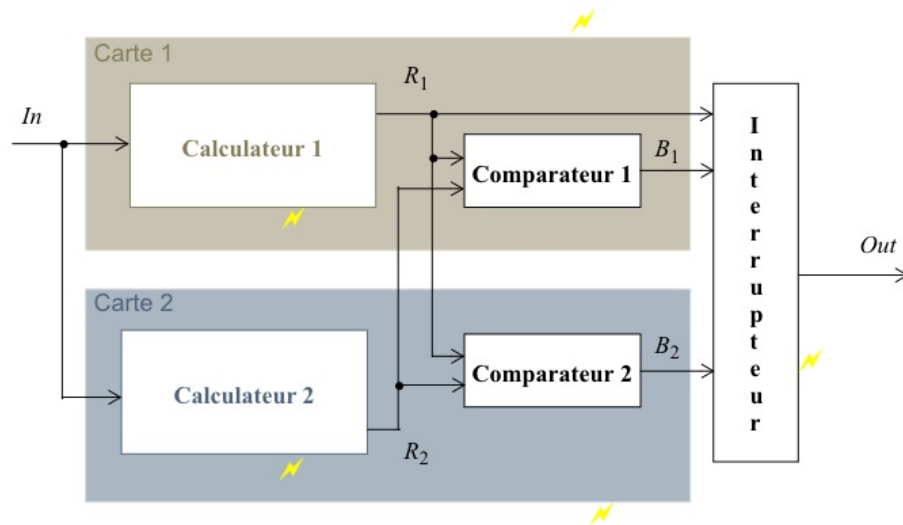
#### 4.4.3 TP d'AltaRica

**Exercice 33** *Ecrire le système de la vanne hydraulique et calculer les coupes minimales. Générez l'arbre de défaillance, dans Systèmes, Génération d'arbre (Exécutable inférence). Ouvrez le fichier .dag généré et comparez-le avec l'arbre que vous avez écrit à la deuxième séance.*

**Exercice 34** *Ecrire le système de la vanne hydraulique et calculer les coupes minimales. Générez l'arbre de défaillance, dans Systèmes, Génération d'arbre (Exécutable inférence). Ouvrez le fichier .dag généré et comparez-le avec l'arbre que vous avez écrit à la deuxième séance.*

**Exercice 35** Modélisez le système défini dans l'exercice 20 p.31. Calculez les coupes minimales et l'arbre de défaillance.

**Exercice 36** On veut modéliser un calculateur tolérant aux pannes appelés COM-MON. Ce système est composé de deux cartes : dans chaque carte se trouvent un calculateur et un comparateur. Les calculateurs sont identiques et font les mêmes calculs à partir d'une entrée commune. Les comparateurs comparent les valeurs de sortie des deux calculateurs : si ces valeurs sont identiques, le comparateur envoie vrai sinon faux. A la sortie des deux cartes, un interrupteur laisse passer la valeur calculée par la première carte s'il reçoit deux discrets à vrai.



1. Donner les modes de défaillance des composants : calculateur et comparateur,
2. Réalisez un premier modèle avec un unique mode de défaillance lost. Calculez les coupes minimales,
3. Réalisez un deuxième modèle avec deux modes de défaillance lost et erroneous. Calculez les coupes minimales pour chaque mode.

On rajoute ensuite l'alimentation électrique du système COM-MON. Dès que l'interrupteur se déconnecte, il coupe définitivement l'alimentation électrique. Pour cela, on ajoute une loi `dirac(0)` sur l'événement stop de l'alimentation électrique. Calculez les nouvelles coupes minimales. Quelles conclusions en tirez-vous ?



## Références

**Livres** Plusieurs livres sont disponibles.

- [1] J.C. Laprie et al., *Guide de la sûreté de fonctionnement*, Cépaduès, 1996, 380 p.
- [2] A. Villemeur, *Sûreté de fonctionnement des systèmes industriels*, Eyrolles, Direction des études et recherches d'Electricité de France (EDF), 1997, 822 p.
- [3] M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, G. Franceschinis, *Modelling with Generalized Stochastic Petri Nets*. Wiley Series in Parallel Computing, John Wiley and Sons, 1994, ISBN : 0-471-93059-8.

**Cours** Plusieurs cours sont disponibles sur Internet.

- [4] Cours d'Andreas Polzer  
[http://www-i11.informatik.rwth-aachen.de/index.php?id=sre\\_ss06&L=0](http://www-i11.informatik.rwth-aachen.de/index.php?id=sre_ss06&L=0)
- [5] Cours de Philip Koopman  
<http://www.ece.cmu.edu/~ece849/>
- [6] Cours d'Eric Chatelet : *Approche Markovienne en sûreté de fonctionnement* - UTT - 2000.

**Thèses - Articles** Quelques articles disponibles sur Internet.

- [7] Aymeric Vincent, *Conception et réalisation d'un vérificateur de modèles AltaRica*. PhD thesis, LaBRI, University of Bordeaux 1, December 2003  
<http://www.labri.fr/perso/vincent/Research/V-CRVMAR-2003.pdf>
- [8] O. Coudert and J.C. Madre, *Implicit and incremental computation of primes and essential primes of Boolean functions*. Proceedings of the 29th ACM/IEEE Design Automation Conference, DAC'92, June (1992). <http://citeseer.ist.psu.edu/4897.html>
- [9] A. Rauzy, *New algorithms for fault trees analysis*. Reliab Engng Syst Saf 40 (1993), pp. 203–211 <http://iml.univ-mrs.fr/~arauzy/publis/Rau93a.pdf.zip>
- [10] Amel Demri, Abdérafi Charki, Fabrice Guérin, Patrice Kahn et Hervé Christofol, *Analyse Qualitative et Quantitative d'un Sys-tème Mécatronique*. CPI 2007.  
<http://www.supmeca.fr/cpi2007/articles2007/CPI2007-048-Demri.pdf>
- [11] F. Innal et Y. Dutuit, *Evaluation de la performance d'un système de production et des contributions individuelles de ses unités constitutives*, 6e Conférence Francophone de MOdélisation et SIMulation - MOSIM'06 - du 3 au 5 avril 2006 – Rabat –Maroc  
<http://www.isima.fr/mosim06/actes/articles/4-AMOEP/25.pdf>
- [12] Charles Castel et Christel Seguin, *Modèles formels pour l'évaluation de la sûreté de fonctionnement des architectures logicielles d'avionique modulaire intégrée*. In AFADL : Approches Formelles dans l'Assistance au Développement de Logiciels, 2002.  
<ftp://altarica.labri.fr/pub/publications/afadl2001.pdf>

**Manuels d'utilisation** Les manuels d'utilisation d'Arbor et OCAS se trouvent sur les machines.

- [13] *Module Arbor*.
- [14] *Module Ocas : Analyse système par arbre de défaillance*. Manuel Utilisateur OcasV4.3 (Septembre2007)