Sûreté de fonctionnement d'architectures informatiques embarquées sur automobile

Christian Ziegler

Thèse

présentée au
Laboratoire d'Analyse et d'Architecture des Systèmes du C.N.R.S.
en vue de l'obtention du
Doctorat de l'Institut National Polytechnique de Toulouse
Spécialité : Informatique

Par

Christian ZIEGLER
Ingénieur INSA

SÛRETÉ DE FONCTIONNEMENT D'ARCHITECTURES INFORMATIQUES EMBARQUÉES SUR AUTOMOBILE

Soutenue le 12 Juillet 1996 devant le jury :

M.	D.	POWELL	Président (*)
M.	M.	DANG	Rapporteur
M.	J.-P.	THOMESSE	Rapporteur
M.	A.	COSTES	Examinateur
M.	Ph.	DESROCHES	Examinateur
M.	B.	GIRARD	Examinateur

(*) Directeur de thèse

Rapport LAAS N°96289

Cette thèse a été préparée au
Laboratoire d'Analyse et d'Architecture des Systèmes du C.N.R.S.
7, avenue du Colonel Roche
31077 Toulouse Cedex

Remerciements

Le travail présenté dans ce mémoire a été effectué au Laboratoire d'Analyse et d'Architecture des Systèmes du Centre National de la Recherche Scientifique, dirigé par Monsieur Alain Costes, Professeur à l'INP de Toulouse, que je remercie pour son accueil et pour l'honneur qu'il me fait en participant au jury de cette thèse.

J'exprime ma grande gratitude à Monsieur Jean-Claude Laprie, Directeur de Recherche au CNRS, pour m'avoir intégré dans son groupe "Tolérance aux Fautes et Sûreté de Fonctionnement Informatique (TSF)".

Surtout, je tiens à exprimer une forte reconnaissance et toute mon estime à Monsieur David Powell, Directeur de Recherche au CNRS, pour avoir accepté de diriger ces travaux. Malgré ses intenses activités il a toujours été à l'écoute, en faisant preuve de disponibilité. A travers ses conseils et ses idées il a su me transmettre les orientations scientifiques importantes qui m'ont permis de mener à bien mon travail de recherche.

Cette thèse n'aurait pas pu être effectuée sans le soutien de la société SIEMENS Automotive S.A. à travers le laboratoire commun MIRGAS, un soutien non seulement financier mais également technique et moral. Je remercie à ce titre Monsieur François Vernières, Directeur du MIRGAS, et Monsieur André Titli, Directeur Adjoint, de m'avoir accueilli dans leur laboratoire.

J'exprime toute ma gratitude à Monsieur Philippe Desroches de la société SIEMENS qui pendant mes années de thèse a trouvé le temps de s'investir dans des activités de recherche fondamentale en manifestant son intérêt. Je le remercie également d'avoir accepté d'être examinateur de ma thèse.

Je tiens à exprimer mes vifs remerciements à :

- Monsieur Michel Dang, Professeur à l'Institut National Polytechnique de Grenoble (INPG)
- Monsieur Jean-Pierre Thomesse, Professeur à l'Ecole Nationale Supérieure d'Electricité et de Mécanique de Nancy (ENSEM-INPL)
- Monsieur Bernard Girard, Directeur de la Stratégie des Technologies Avancées dans le groupe Peugeot-Citroën à Paris

pour l'honneur qu'ils me font en participant au jury de cette thèse et particulièrement à Messieurs Michel Dang et Jean-Pierre Thomesse pour avoir accepté la charge de Rapporteur.

Je tiens à remercier également tous les membres du groupe TSF qui, par leur aide, leurs conseils et leur amitié, m'ont permis de rendre mon travail agréable. Je pense en

particulier à Muriel Daran, Nicolae Fota et Christine Mazuet avec lesquels j'ai partagé le plaisir d'aller nager. Un grand merci également à Yves Crouzet qui a toujours su résoudre les divers problèmes de support d'informatique. Notre secrétaire devouée Joëlle Penavayre a le mérite d'être toujours à l'écoute des autres et je garde un souvenir agréable des discussions avec elle. Je suis également reconnaissant à Jean-Paul Blanquart et Mohamed Kaâniche, pour la minutie avec laquelle ils ont relu mon manuscrit.

Au niveau du LAAS, je voudrais tout d'abord remercier les ingénieurs de l'équipe "Informatique et Instrumentation", en particulier Martine Aguera, Serge Bachmann, Laurent Blain et Jean-Etienne Doucet, pour leur aide précieuse concernant l'utilisation de l'outil SURF-2. Je remercie aussi Guy Juanole, Professeur à l'Université Paul Sabatier, ainsi que Robert Valette, Directeur de Recherche au CNRS, avec lesquels j'ai eu de nombreux échanges d'idées fructueux et des discussions agréables. Mes remerciements s'adressent également à tous les membres du service "Réception-Standard" (représenté par Mesdames Marcelle Bisson et Michèle Tomasi-Bisson), du service "Magasin" (surtout Christian Marrot et André Pouil) ainsi que du service "Documentation-Edition" (en particulier Anne Bergez, Jean Catala, Daniel Daurat, Arlette Evrard, Michèle Powell et Roger Zittel) pour leur aide.

Un très grand merci également à tous mes amis personnels du LAAS avec lesquels j'ai partagé la plupart des repas de midi à la cantine et qui m'ont permis de garder mon sens de l'humour et de découvrir les spécialités gastromoniques de la France. En espérant de ne pas avoir oublié une de ces personnes, je cite : Gérard Bauzil, Christophe Bricout, Jean Catala, Bernard Franc, Guy Galinier, Guy Larregola, Christian Marrot, André Pouil, Eric Tasselli et Roger Zittel.

Sommaire

Introduction

L'automobile représente un moyen de transport qui s'est énormément développé pendant ce siècle et qui est utilisé dans de nombreuses circonstances. Sachant que la défaillance technique d'un véhicule automobile (par exemple, le non-fonctionnement du freinage) peut affecter la vie d'êtres humains, il est évident que la sûreté de fonctionnement est un facteur incontournable à prendre en compte dans la spécification, la conception, le développement et la validation de l'automobile. Dans le milieu industriel, notamment dans l'industrie automobile, le respect de la sûreté de fonctionnement se traduit généralement par des démarches d'*assurance qualité*. En ce qui concerne les parties mécaniques et hydrauliques de l'automobile, l'assurance qualité passe essentiellement par l'introduction d'un facteur de *sur-dimensionnement*.

Or, la tendance est de supprimer en partie, voire de remplacer, certains systèmes mécaniques tels que la traction, le freinage ou la direction par des commandes électroniques. Cela est fait dans le but entre autres de baisser les coûts, de diminuer la pollution, d'améliorer les performances et de réduire le poids. De plus, on implante de nouvelles fonctionnalités gérées par l'électronique qui sont souvent destinées à améliorer la sécurité des conducteurs, par exemple le coussin gonflable.

Les systèmes électroniques introduits dans l'automobile évoluent sans cesse. En effet, l'amélioration de la technologie des semi-conducteurs ainsi que les progrès en miniaturisation et intégration en électronique permettent de développer des capteurs, actionneurs et calculateurs de plus en plus complexes et compacts. De plus, on constate une introduction de l'*informatique embarquée* par le biais de microprocesseurs, les systèmes embarqués comportant aujourd'hui beaucoup de logiciel.

L'introduction des systèmes embarqués électroniques-informatiques constitue un saut technologique qu'il faut maîtriser, car la nouvelle technologie représente et manipule les grandeurs physiques sous une forme numérique. La nature abstraite de cette information ne permet plus alors d'appliquer les démarches de conception de systèmes mécaniques pour se protéger contre les fautes en vie opérationnelle. De plus, l'augmentation importante du nombre de fonctions informatiques dans l'automobile conduit à une complexité qui, si elle n'est pas maîtrisée, pourrait être néfaste sur le plan de la sûreté de fonctionnement, d'autant plus que les systèmes informatiques sont soumis à des contraintes d'environnement sévères.

Il est important de noter que, à l'heure actuelle, les systèmes embarqués électroniques sont conçus spécifiquement par rapport à une fonctionnalité précise, ce qui signifie que les techniques de sûreté de fonctionnement sont appliquées localement au niveau du système embarqué. La complexité matérielle et logicielle des systèmes embarqués automobiles étant sans cesse croissante, les concepteurs cherchent à diminuer le nombre de composants et à intégrer plusieurs fonctionnalités sur un même module, dans le but de réduire les coûts et de faciliter l'harmonisation du fonctionnement global. Donc, une approche globale à la conception de l'ensemble des systèmes embarqués et, par conséquent, des techniques de sûreté de fonctionnement associées sera nécessaire. Cette approche devrait permettre de mieux remplir les objectifs de fonctionnalité, de sûreté de fonctionnement et de coût.

Nos travaux visent à s'assurer que la sûreté de fonctionnement du véhicule équipé de systèmes électroniques ne soit pas dégradée par rapport au véhicule équipé de systèmes mécaniques. Dans un but d'illustration de la nouvelle approche globale de conception, nous allons proposer un spectre qui définit différentes solutions d'architecture pour un ensemble de systèmes embarqués donnés. Ce spectre nous permettra surtout de comparer les architectures par rapport à plusieurs critères. La sûreté de fonctionnement étant le critère primordial, nous allons développer une méthode d'évaluation permettant d'effectuer une prévision quantitative des fautes pour chacune des architectures du spectre. C'est ainsi que nous espérons pouvoir guider le choix d'architecture pour un ensemble de fonctions données.

Dans le premier chapitre de ce mémoire, nous détaillons l'historique et l'état de l'art des systèmes embarqués automobiles. Après une définition de la terminologie de la sûreté de fonctionnement, plus particulièrement de la tolérance aux fautes, nous exposons ensuite les particularités de la sûreté de fonctionnement des systèmes embarqués sur l'automobile.

Le deuxième chapitre décrit les différentes techniques de sûreté de fonctionnement qui sont employées afin de détecter et de traiter les erreurs des capteurs, actionneurs, calculateurs et liaisons. Quant aux liaisons, la tendance actuelle vers l'intégration des différents systèmes électroniques embarqués nous amène à comparer la sûreté de fonctionnement de différents réseaux multiplexés du domaine automobile. C'est dans cette optique également que nous proposons un spectre d'architectures qui permet d'illustrer et de classer différentes possibilités existantes entre une architecture entièrement fédérée à un bout du spectre et une architecture entièrement intégrée à l'autre. Une comparaison préliminaire des architectures du spectre est effectuée, avec un effet de loupe sur l'aspect coût.

Le troisième chapitre est dédié à l'évaluation quantitative de la sûreté de fonctionnement dans l'objectif de comparer certaines architectures présentées dans le deuxième chapitre. Parmi les différentes techniques d'évaluation possibles, nous

focalisons sur la technique d'évaluation probabiliste par modélisation markovienne, en passant par une méthode de modélisation modulaire en Réseaux de Petri Stochastiques Généralisés. Cette méthode est illustrée sur différentes architectures du coussin gonflable et de la direction électronique.

L'objectif du quatrième chapitre est de comparer quantitativement différentes architectures au niveau de leur sûreté de fonctionnement à l'aide des modèles développés dans le troisième chapitre. Les résultats obtenus sont présentés et analysés, dans l'objectif de tirer des conclusions concernant le choix d'une architecture pour une fonction donnée.

Chapitre I

Préliminaires

Dans ce chapitre introductif, nous détaillons la problématique de la sûreté de fonctionnement des systèmes embarqués sur l'automobile. Cette problématique étant fortement liée à l'évolution technologique de l'automobile, nous détaillons l'historique et l'état de l'art des systèmes embarqués automobiles dans le premier paragraphe de ce chapitre.

La conception de l'automobile, et plus particulièrement des systèmes informatiques embarqués, est soumise à diverses contraintes. Dans le deuxième paragraphe de ce chapitre, nous traitons des contraintes de conception d'ordre général, indépendantes de celles issues de la sûreté de fonctionnement.

Afin d'établir une terminologie cohérente pour la suite de ce mémoire, nous donnons dans le troisième paragraphe une synthèse des définitions de base de la sûreté de fonctionnement avec un effet de loupe sur les concepts de la tolérance aux fautes.

Après cet état de l'art en la matière, nous allons montrer les particularités de la sûreté de fonctionnement pour les systèmes embarqués sur l'automobile (voir également [Ziegler *et al.* 1994a]). Nous nous intéressons au niveau de sûreté de fonctionnement qui doit être envisagé pour les nouveaux systèmes informatiques embarqués sur l'automobile et aux contraintes qui s'opposent à cet objectif. C'est ainsi que nous exposons dans le quatrième paragraphe le problème de la spécification et de la validation des objectifs de sûreté de fonctionnement, pour ensuite détailler dans le cinquième paragraphe les contraintes affectant la sûreté de fonctionnement.

I.1 Equipement automobile : Historique et état de l'art

Dans ce paragraphe, nous décrivons l'évolution des systèmes embarqués sur l'automobile en focalisant sur l'introduction de l'électronique et de l'informatique.

Les systèmes embarqués automobile ont connu et connaissent toujours une évolution rapide qui est essentiellement due à l'introduction de l'électronique depuis les années 60. En effet, l'évolution de la technologie des semi-conducteurs a

fortement influencé la conception des véhicules ainsi que leur équipement, apportant des améliorations importantes concernant en particulier :

- l'augmentation du confort et des performances (conduite, manœuvrabilité, comportement routier et freinage) ;
- la diminution de la pollution ;
- la diminution de la consommation ;
- l'augmentation de la sécurité.

Certains systèmes mécaniques ont été et sont remplacés par des systèmes électroniques dont la complexité matérielle et logicielle est sans cesse croissante [Numazawa 1988]. Ces systèmes embarqués comportent aujourd'hui beaucoup de logiciel. En effet, la part de l'informatique embarquée sur l'automobile est de plus en plus grande ; actuellement, elle représente environ 15% du coût total d'un véhicule et on estime que le coût de l'informatique dans le véhicule de l'an 2000 représentera 20% du coût total [Kopetz 1995]. La *Figure 1-1* (adaptée de [Panik 1988, Rivard 1989]) illustre le développement de l'équipement électronique et informatique du véhicule (voir aussi [Aono 1988, Paulsen 1988]). On peut distinguer quatre phases qui se recouvrent.

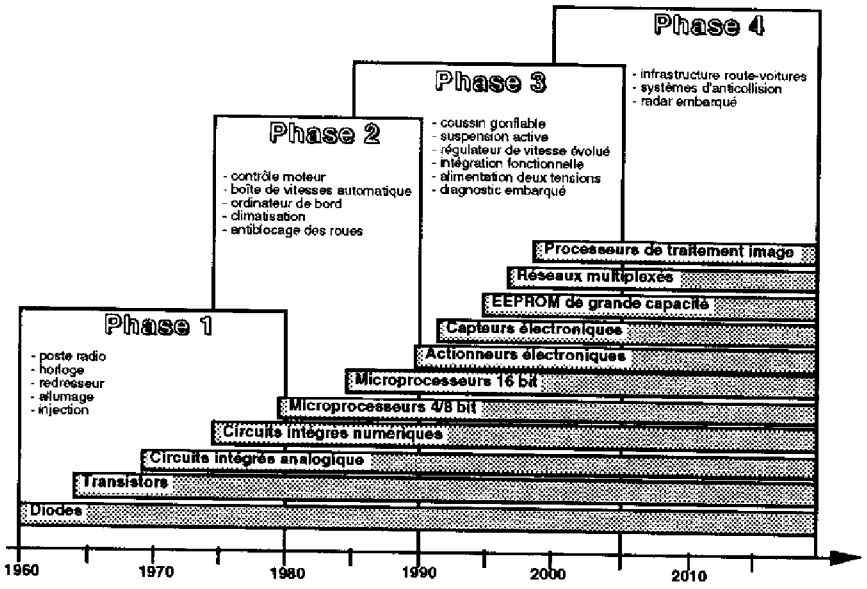


Figure 1-1 : Historique et évolutions prévisibles de l'informatique embarquée automobile (adapté de [Panik 1988, Rivard 1989]*)*

Dans une *première phase* (1960-1980), l'utilisation des éléments semi-conducteurs discrets a permis des applications essentiellement destinées à améliorer le confort (par exemple l'horloge) et les performances (par exemple l'allumage).

La *deuxième phase* (1975-1990) est marquée par l'introduction de l'informatique embarquée par le biais de microprocesseurs permettant à la fois d'améliorer les fonctions existantes et d'implanter de nouvelles fonctions. On voit ainsi apparaître le système d'antiblocage des roues qui augmente la sécurité de freinage. L'allumage évolue vers un système complet de contrôle moteur permettant de diminuer la consommation et la pollution. Certains systèmes mécaniques sont en partie supprimés ou remplacés, c'est le cas par exemple du distributeur d'allumage.

Il était prévu dans [Rodda 1990] que le volume du logiciel embarqué sur l'automobile doublerait tous les 5 ans (cf. *Tableau 1-1*). Il faut noter que la taille des logiciels de certains systèmes commercialisés actuellement dépasse ces prévisions ; par exemple, le système du contrôle moteur commercialisé par Siemens Automotive en 1995 comprend environ 100 Koctets de logiciel. A cette complexité logicielle s'ajoute un nombre croissant de capteurs et d'actionneurs.

Tableau 1-1 : Volume mémoire du logiciel dans l'automobile en Koctets [Rodda 1990]

	1990	1995	2000
Antiblocage des roues	10	12	18
Communication	16	32	100
Coussin gonflable	4	8	12
Direction	1	4	12
Fonctions habitacle	4	8	16
Groupe moto propulseur	16	32	96
Radio	8	8	16
Suspension	4	12	24
Total	**63**	**116**	**294**

Dans la *troisième phase* (1985-2005) on constate que les progrès en miniaturisation et intégration en électronique permettent de développer des capteurs, actionneurs et calculateurs de plus en plus complexes et compacts. On observe en particulier l'émergence d'une nouvelle technologie qui est celle des microsystèmes. L'objectif est ici d'intégrer de manière monolithique et hybride des systèmes complets comptants capteurs, actionneurs et le traitement du signal (VLSI, Very Large Scale Integration). La demande vise à élargir le champ d'application des systèmes électroniques intégrés en appliquant les principes de fabrication collective de la microélectronique à l'intégration des systèmes. En effet, dans le but de réduire les coûts, les concepteurs cherchent à diminuer le nombre de composants et à intégrer plusieurs fonctionnalités sur un même module. Par exemple, l'antiblocage et

l'antipatinage des roues sont souvent implantés sur le même calculateur. D'autres possibilités d'intégration sont en cours d'analyse et d'étude, par exemple le contrôle de la boîte de vitesses automatique avec le contrôle moteur. Le LAAS a activement participé à l'évolution de l'électronique automobile comme partenaire au sein des programmes PROMETHEUS et PROCHIP [Estève *et al.* 1995], du projet européen AMIS et d'actions plus régionales tel le projet CITA ou le diagnostic de pannes [Poulard 1996].

On approche actuellement de la *quatrième phase* (après 2000) dans laquelle les efforts porteront plus particulièrement sur l'aspect sécurité routière. On sait que, parmi les systèmes de transport, l'automobile est — de loin — le plus dangereux [Baranowski 1990]. Dans l'Union Européenne, plus de 1,2 millions d'accidents sont comptabilisés chaque année, avec environ 50 000 morts et 1,7 millions de blessés(1). La *Figure 1-2* [DEKRA 1992] indique que 85 % des accidents d'automobile sont dus à des erreurs du conducteur et que seulement 1% des accidents sont dus à des défaillances techniques causées par une mauvaise maintenance et l'usure mécanique.

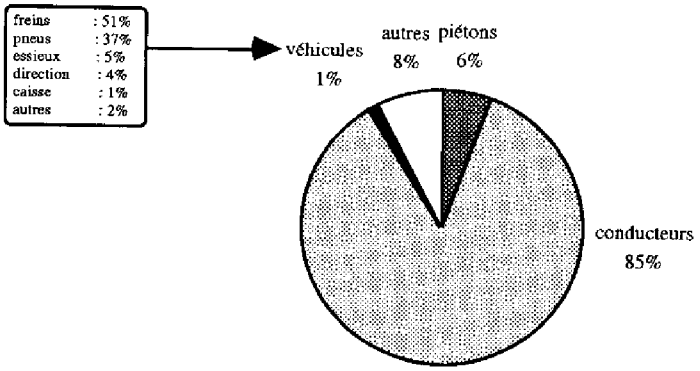


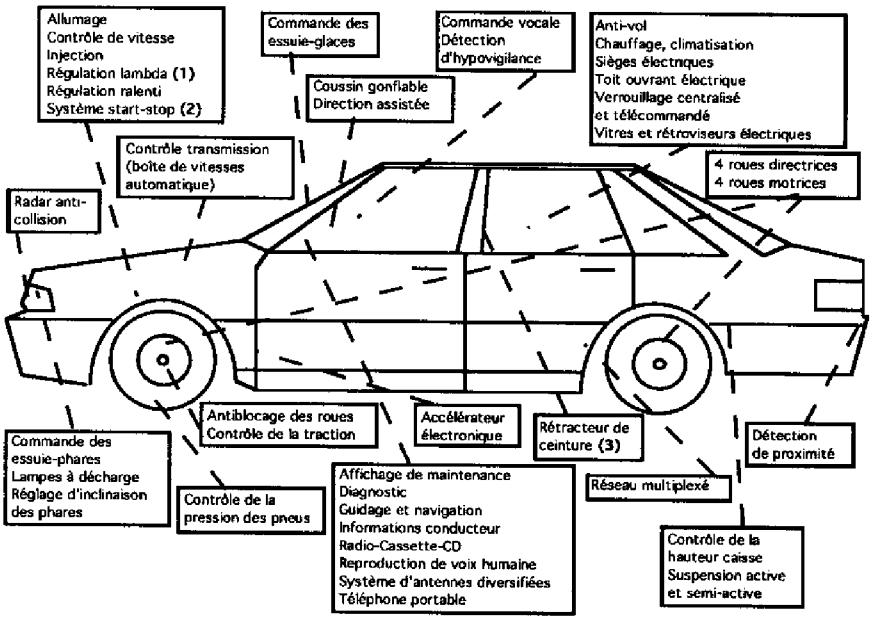
Figure 1-2 : Causes d'accidents d'automobile en RFA [DEKRA 1992]

Une réduction effective des accidents passe donc par la diminution du nombre ainsi que par l'atténuation des conséquences des erreurs humaines. Une contribution à l'augmentation de la sécurité routière pourrait être apportée par de nouveaux systèmes électroniques telle la régulation de vitesse qui garantirait une distance constante entre les véhicules (appelé AICC pour *Autonomous Intelligent Cruise Control* [Martin 1993, Palmquist 1993]), la détection d'hypovigilance [Muzet & Clot 1993, Giralt & Vernières 1995] ou l'anti-collision [Fontaine *et al.* 1989]. Ces systèmes, ainsi que d'autres systèmes tels que l'aide à la navigation, permettraient aussi d'augmenter la fluidité du trafic et, ainsi, de diminuer la congestion des routes qui devient un problème de plus en plus grave, surtout en milieu urbain. On

(1) D'après le Conseiller Technique auprès du Délégué Interministériel à la Sécurité Routière.

envisage, par exemple, la communication entre les véhicules et des centres de contrôle (guidage des véhicules par des balises au bord de la route ou par des fils enterrés sous la route). Les aspects évoqués ici font actuellement l'objet de nombreux programmes de recherche dans le monde entier, par exemple PROMETHEUS en Europe et IVHS (maintenant appelé ITS) aux Etats-Unis [Chen & Ervin 1992, Mauro 1993].

La *Figure 1-3* illustre le nombre très élevé de systèmes embarqués électroniques disponibles aujourd'hui (voir [Numazawa 1988, Arai 1990]), mais les progrès réalisés sur ces systèmes n'ont certainement pas atteint leur limite. Les améliorations futures seront poussées par les progrès en technologie des semi-conducteurs, par les demandes de la clientèle ainsi que par des normes d'antipollution et de sécurité routière plus sévères.



(1) Régulation lambda : Lors d'une utilisation du pot catalytique on met une sonde dans le pot d'échappement qui mesure le coefficient d'air (appelé "Lambda"). L'injection règle le débit en essence de telle manière que Lambda soit égal à 1, ce qui permet de garantir un bon fonctionnement du pot catalytique.
(2) Système start-stop : Le moteur est coupé quand la voiture arrête de rouler et il redémarre lors de la prochaine sollicitation (appui sur l'accélérateur).
(3) Rétracteur de ceinture : Lors d'un fort choc, la ceinture se rétracte automatiquement pour éviter un contact du conducteur avec le volant.

Figure 1-3 : Informatique embarquée de l'automobile d'aujourd'hui et dans le futur proche

I.2 Equipement automobile : Contraintes d'ordre général

Lors de la conception et du développement d'un nouveau véhicule, le constructeur doit tenir compte de plusieurs contraintes. Dans ce paragraphe, nous focalisons sur les contraintes affectant les systèmes embarqués électroniques de l'automobile, où nous pouvons distinguer les contraintes suivantes :

- Le **coût** :
  L'automobile est un produit grand public (actuellement, il y a environ 500 millions de véhicules dans le monde [Numazawa 1988, Hoefflinger 1993]) et les constructeurs sont soumis à une concurrence toujours plus sévère. La conception, le développement, la production et la maintenance ont une incidence sur le coût. Les systèmes embarqués électroniques doivent être conçus de la façon la moins chère possible et dans des délais de plus en plus courts, tout en garantissant un fonctionnement sûr.
- La **législation** [Rathkey 1992] :
  Les normes d'antipollution en Europe, au Japon et aux Etats-Unis imposent surtout une forte réduction des émissions polluantes, d'où la nécessité de pots catalytiques et de systèmes de contrôle moteur de plus en plus sophistiqués. D'autres lois, aux Etats-Unis par exemple, obligent le constructeur à mettre en œuvre un système de sécurité passive[1] (le coussin gonflable ou le rétracteur de ceinture).
- La **modularité fonctionnelle** :
  Dans le monde, il existe une grande variété de véhicules qui sont proposés en plusieurs variantes avec des options nombreuses. Par conséquent, les véhicules sont presque tous équipés différemment. De plus, l'évolution entre gammes de modèles successifs est très rapide. Cela signifie que le constructeur, et plus particulièrement l'équipementier, doit concevoir une architecture qui facilite la gestion de la modularité fonctionnelle.
- Le **poids** et l'**encombrement** :
  Le poids des systèmes embarqués électroniques doit être limité afin de minimiser la consommation du véhicule. La minimisation de l'encombrement à son tour facilite la conception de l'automobile.
- La **consommation électrique** :
  Pour ne pas trop augmenter la charge du moteur, la puissance consommée par les systèmes embarqués électroniques est limitée.

---

[1] Dans l'industrie automobile, on entend par sécurité passive tous les systèmes qui sont destinés à réduire les conséquences d'accidents. La sécurité active regroupe les systèmes qui ont pour objectif d'éviter les accidents.

- La **structure industrielle** :
  Il y a en général différents équipementiers qui fournissent leurs produits au constructeur automobile, ce qui complique la gestion et l'optimisation globale des fonctions électroniques. [Morin 1994] indique que quatre scénarios futurs peuvent être envisagés :
  a) Il y aura un fournisseur unique pour l'équipement électronique complet du véhicule. Cette solution garantirait une certaine cohérence de l'architecture réalisée et permettrait une mise en responsabilité évidente. L'inconvénient serait surtout le rôle dominant de l'équipementier retenu par rapport au constructeur.
  b) Le constructeur automobile spécifiera l'architecture matérielle et logicielle aux équipementiers de son choix. L'avantage principal de cette solution serait la maîtrise de la compétence par le constructeur. Mais la charge de travail pour le constructeur serait relativement élevée. Un autre inconvénient serait la mise en place d'un partage de responsabilité entre constructeur et équipementiers.
  c) Il y aura un réalisateur unique qui intègre toutes les spécifications des équipementiers. Plusieurs équipementiers lui fourniront les capteurs, actionneurs et calculateurs et lui spécifieront également le logiciel nécessaire. L'avantage de cette solution serait la préservation de la compétence de l'équipementier. L'inconvénient principal serait le partage de la responsabilité entre les équipementiers et le réalisateur. De plus, le transfert de compétence des équipementiers au réalisateur pourrait signifier un allongement de la durée de développement.
  d) Le constructeur définira une structure d'accueil logiciel qui pourra intégrer les logiciels fournis par plusieurs équipementiers. Il spécifie également les fonctions et l'allocation des ressources du calculateur aux équipementiers. Il est responsable de l'intégration et de la validation du système global. Chaque équipementier conçoit, réalise et valide ses fonctions. L'avantage de cette solution serait le maintien du rôle et de la compétence des équipementiers. Mais cela nécessiterait la mise en place d'un processus de développement et d'un cadre de référence logiciel rigoureux. Actuellement, de telles structures n'existent pas (sauf au sein d'une même société).

Les contraintes que nous venons d'exposer conditionnent les grands choix de conception et l'introduction de nouvelles technologies. La difficulté consiste à trouver le juste compromis entre ces contraintes souvent contradictoires et difficilement conciliables. D'une part, le client accorde une attention particulière au niveau des performances, à la qualité du comportement et à la sécurité ; il souhaite que son prochain véhicule soit plus confortable que le précédent, et aussi plus sûr. D'autre part, on observe une prise de conscience économique et écologique qui impose aux véhicules d'être sans cesse plus sobres, plus propres et plus silencieux. Mais plus de

confort, plus de sécurité et moins de bruit sont généralement responsables d'une augmentation du poids et du coût. Le souci de l'environnement et de la consommation, la recherche des performances et d'une bonne tenue de route exigent, pour leur part, un poids aussi réduit que possible.

I.3 La sûreté de fonctionnement

Ce paragraphe présente une synthèse des définitions données dans [Laprie *et al.* 1996], référence à laquelle le lecteur peut se reporter pour des aspects plus détaillés ou complémentaires.

La *sûreté de fonctionnement* est la propriété d'un système permettant à ses utilisateurs de placer une confiance justifiée dans le service délivré. Le *service délivré* par un système est le comportement tel que perçu par ses utilisateurs. Un *utilisateur* est un autre système (humain ou physique) qui interagit avec le système considéré. La taxonomie définie dans ce paragraphe est représentée et résumée par le schéma de la *Figure 1-4*.

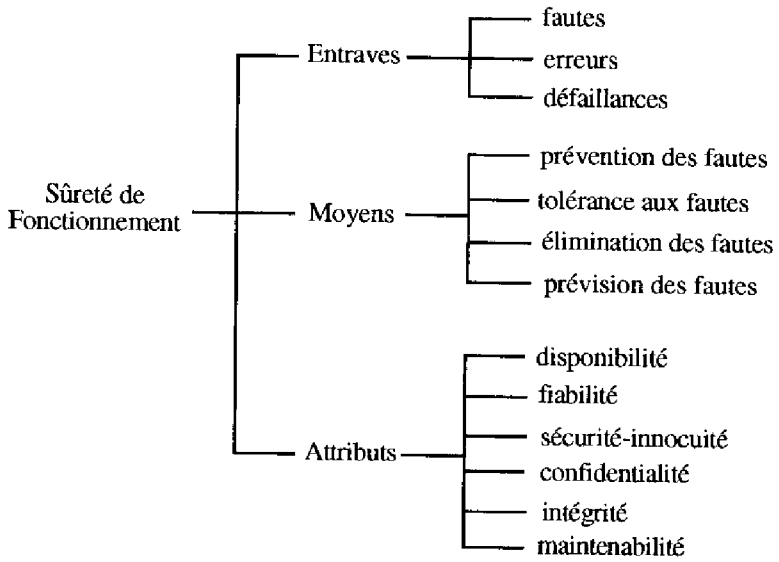


Figure 1-4: Taxonomie de la sûreté de fonctionnement [Laprie *et al.* 1996]

Dans les trois sections suivantes, nous allons définir les *entraves*, *moyens* et *attributs* de la sûreté de fonctionnement. La quatrième section est destinée à la *tolérance aux fautes*.

I.3.1 Entraves

Les *entraves* sont les circonstances indésirables de la sûreté de fonctionnement :

- Une défaillance d'un système est le fait que le service délivré dévie de l'accomplissement de la fonction du système.
- Une erreur est la partie de l'état du système qui est susceptible d'entraîner une défaillance.
- Une faute est la cause adjugée ou supposée d'une erreur.

Les mécanismes de création et de manifestation des fautes, erreurs et défaillances peuvent être résumés comme suit :

Une faute est *dormante* (c'est-à-dire qu'elle ne produit pas d'erreur) ou *active*. Une faute peut passer, de manière cyclique, de l'état dormant à l'état actif et vice versa.

Dès que la faute est active elle crée une erreur qui peut rester *latente* (une erreur latente peut disparaître sans être détectée) ou être *détectée*. Une erreur est donc la manifestation d'une faute dans le système. En général, elle se *propage* et crée de nouvelles erreurs.

La défaillance est la conséquence d'une erreur non-confinée, elle est la manifestation d'une erreur sur le service du système.

Un système peut être décomposé d'une façon hiérarchique en plusieurs sous-systèmes qui interagissent entre eux. De ce point de vue, l'enchaînement *faute --> erreur --> défaillance* est récurrent, ce qui veut dire qu'une défaillance d'un sous-système devient une faute pour les autres sous-systèmes interagissant avec lui ainsi que pour le système qui le contient.

Dans ce contexte, on définit le *domaine de confinement* d'une faute : ce sont les systèmes ou sous-systèmes auxquels on peut attribuer l'origine de la défaillance.

Un système ne défaille pas toujours de la même manière, ce qui conduit à la notion de *mode de défaillance*, qui peut être caractérisé selon trois points de vue :

- Le domaine de défaillance, qui conduit à distinguer entre *défaillances en valeur* et *défaillances temporelles*. Une classe de défaillances relative à la fois aux valeurs et aux conditions temporelles est constituée par les *défaillances par arrêt*, c'est-à-dire l'activité du système n'est plus perceptible aux utilisateurs. Cette absence d'activité peut se traduire par un *figement* de l'état des sorties (on parle aussi de *figement sur défaillance*), ou par un *silence* des sorties (on parle aussi de *silence sur défaillance*).

- La perception des défaillances par plusieurs utilisateurs du système, qui conduit à distinguer entre *défaillances cohérentes* et *défaillances incohérentes* (ou byzantines [Lamport *et al.* 1982]).
- Les conséquences des défaillances sur l'environnement du système, qui conduit à ordonner les modes de défaillance en plusieurs niveaux de sévérité (ou gravité), auxquels sont généralement associés des probabilités d'occurrence maximales admissibles. D'une manière générale, on peut définir deux niveaux extrêmes :
  a) Les *défaillances bénignes*, dont les conséquences sont du même ordre de grandeur que le bénéfice procuré par le service délivré en l'absence de défaillance.
  b) Les *défaillances catastrophiques*, dont les conséquences sont incommensurablement différentes du bénéfice procuré par le service délivré en l'absence de défaillance.

  La notion de sévérité des défaillances permet de définir la notion de *criticité* : la criticité d'un système est la plus forte sévérité de ses modes de défaillance.
  Un système dont toutes les défaillances sont — dans une mesure acceptable — des défaillances bénignes est un système *sûr en présence de défaillance*.

I.3.2 Moyens

La conception et la réalisation d'un système sûr de fonctionnement passent par l'utilisation combinée de différentes techniques et méthodes, appelées *moyens* de la sûreté de fonctionnement :

- La prévention des fautes : comment empêcher l'occurrence ou l'introduction de fautes. La prévention des fautes est appliquée au cours de toute la phase de conception et construction du système, par exemple en utilisant des méthodes et outils spécifiques au cours de la définition du système ou en améliorant le procédé de fabrication.
- La tolérance aux fautes : comment fournir un service approprié en dépit des fautes. La tolérance aux fautes intervient lors de la définition de l'architecture du système, elle est mise en œuvre par le traitement des fautes et par le traitement des erreurs. Ce point sera détaillé dans le paragraphe I.3.4 .
- L'élimination des fautes : comment réduire la présence des fautes. On essaye d'éliminer les fautes résiduelles pendant la phase de conception et fabrication lors des étapes de vérification, d'inspection et de test ainsi que durant la vie opérationnelle par le biais de la maintenance.

- La prévision des fautes : comment estimer la présence, la création et les conséquences des fautes. La prévision des fautes intervient afin de valider les choix d'architecture ou comme critère d'arrêt des tests.

La prévention des fautes et la tolérance aux fautes permettent de fournir au système l'aptitude à délivrer un service conforme à l'accomplissement de sa fonction. L'élimination et la prévision des fautes permettent de donner confiance dans cette aptitude.

I.3.3 Attributs

Selon l'application à laquelle le système est destiné, la sûreté de fonctionnement peut être vue selon des propriétés différentes mais complémentaires qui permettent de définir ses *attributs* :

- La fiabilité : l'aptitude d'un système à délivrer le service attendu pendant une durée donnée (*continuité du service*).
- La disponibilité : l'aptitude d'un système à être en état de délivrer le service attendu à un instant donné (*être prêt à l'utilisation*).
- La sécurité-innocuité: l'aptitude d'un système à éviter de faire apparaître des défaillances catastrophiques (*non-occurrence de conséquences catastrophiques pour l'environnement*).
- La confidentialité : l'aptitude à préserver la confidentialité des informations dans le système (*non-occurrence de divulgations non-autorisées de l'information*).
- L'intégrité : l'aptitude à préserver l'intégrité des informations dans le système (*non-occurrence d'altérations inappropriées de l'information*).
- La maintenabilité : la facilité de préserver ou améliorer l'aptitude du système à délivrer le service attendu (*aptitude aux réparations et aux évolutions*).

L'association, à la confidentialité, de l'intégrité et de la disponibilité vis-à-vis des actions autorisées, conduit à la sécurité-confidentialité.

Les attributs permettent d'exprimer les propriétés qui sont attendues du système et d'apprécier la qualité du service délivré.

La vie d'un système en service est perçue par ses utilisateurs comme une alternance entre deux états du service par rapport à l'accomplissement de la fonction du système :

- Le service correct : le service délivré accomplit la fonction du système.
- Le service incorrect : le service délivré n'accomplit pas la fonction du système.

La transition du service correct au service incorrect correspond à une défaillance et la transition du service incorrect au service correct correspond à la restauration du service. La quantification de l'alternance entre service correct et service incorrect permet de définir fiabilité, disponibilité et maintenabilité comme des *mesures* de la sûreté de fonctionnement :

- La fiabilité : la mesure de la délivrance continue d'un service correct, ou, de façon équivalente, du temps jusqu'à défaillance.
- La disponibilité : la mesure de la délivrance d'un service correct par rapport à l'alternance "service correct — service incorrect".
- La maintenabilité : la mesure du temps jusqu'à restauration depuis la dernière défaillance survenue, ou ce qui est équivalent, de la délivrance continue d'un service incorrect.

I.3.4 La tolérance aux fautes

Le but de la tolérance aux fautes est d'éviter que les erreurs et les fautes entraînent une défaillance du système. Les techniques de tolérance aux fautes sont basées sur les principes de la redondance, ce qui signifie que le système complet contient des éléments qui ne seraient pas nécessaires dans un système ignorant les fautes. La tolérance aux fautes est mise en œuvre par les techniques de *traitement d'erreur* et de *traitement de faute* [Laprie *et al.* 1996].

Le **traitement d'erreur** est destiné à éliminer les erreurs, si possible avant qu'une défaillance ne survienne. Le traitement d'erreur fait appel à trois primitives :

- La détection d'erreur : on identifie un état erroné comme tel.
- Le diagnostic d'erreur : on estime les dommages créés par l'erreur qui a été détectée et par les erreurs éventuellement propagées avant la détection.
- Le recouvrement d'erreur : on substitue à l'état erroné un état exempt d'erreur où l'état exempt d'erreur peut être atteint par trois techniques différentes :
  - La reprise (recouvrement arrière) : on ramène le système dans un état préalablement occupé et supposé exempt d'erreur.
  - La poursuite (recouvrement avant) : on trouve un nouvel état exempt d'erreur.
  - La compensation d'erreur : on délivre un service approprié malgré la présence d'une erreur interne, donc l'état erroné doit contenir suffisamment de redondance pour permettre la transformation de l'état erroné en un état exempt d'erreur. On distingue deux types de compensation. Dans le cas de la compensation dynamique, on détecte l'erreur avant de la compenser par une commutation d'un composant défaillant vers un composant non-défaillant. Dans le cas de la

compensation statique, l'erreur est masquée (par exemple par vote majoritaire) et éventuellement détectée par la suite. La compensation statique est appliquée systématiquement, même en l'absence d'erreur. On parle aussi de *masquage d'erreur.*

Le surcoût temporel (en temps d'exécution) nécessaire pour le traitement d'erreur peut varier considérablement selon la technique adoptée. Dans le cas de la reprise et de la poursuite, le surcoût temporel est plus important que dans le cas de la compensation. D'une manière générale, on peut dire que plus on dispose de redondance structurelle, plus faible sera la redondance temporelle. Cela conditionne souvent le choix de la stratégie de tolérance aux fautes à adopter par rapport à la granularité temporelle de l'utilisateur du système.

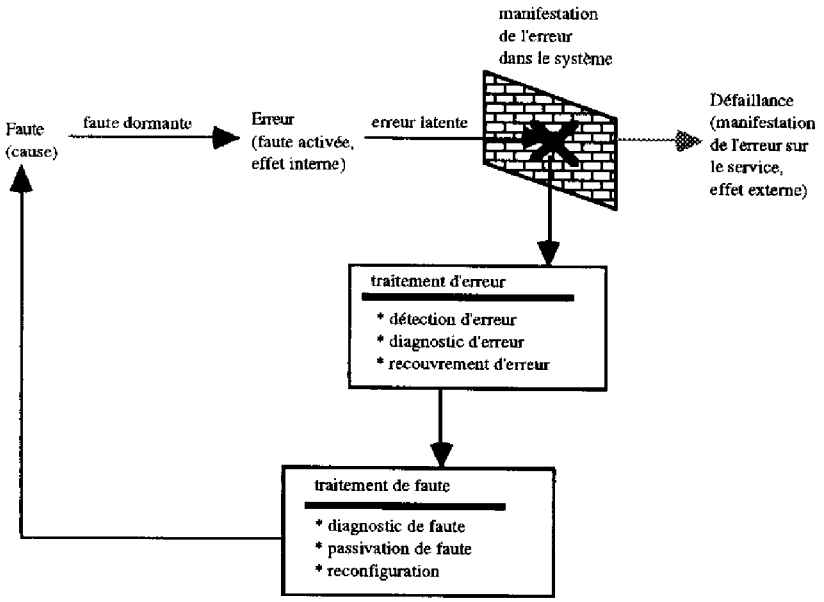


Figure 1-5 : Traitement de faute et traitement d'erreur

Le **traitement de faute** est destiné à éviter que la ou les fautes ne soient activées à nouveau. Il comporte trois étapes :

- Le diagnostic de faute : on localise et caractérise les fautes.
- La passivation de faute : on empêche une nouvelle activation des fautes.
- La reconfiguration : au cas où le système ne peut plus délivrer le même service qu'auparavant, on modifie la structure du système afin de délivrer un service acceptable, bien qu'éventuellement dégradé (il est même envisageable de pouvoir tolérer de nouvelles fautes après la reconfiguration du système).

Le concept de tolérance aux fautes est illustré par la *Figure 1-5*. Il s'agit d'un concept récursif, cela veut dire que les mécanismes destinés à mettre en œuvre la tolérance aux fautes doivent eux-mêmes être protégés contre les fautes susceptibles de les affecter.

I.4 Equipement automobile : Objectifs de sûreté de fonctionnement

Lors de la conception d'un système, le constructeur spécifie non seulement sa fonctionnalité, mais aussi les objectifs à atteindre en termes de sûreté de fonctionnement. La spécification des objectifs de sûreté de fonctionnement est accompagnée d'une procédure de validation pour vérifier que les objectifs ont été atteints. Des méthodes et des outils de développement spécifiques sont souvent appliqués pour supporter cette démarche.

Concernant la spécification et la validation des objectifs de sûreté de fonctionnement dans l'automobile, il s'agit à l'heure actuelle de savoir maîtriser le saut technologique depuis la mécanique et l'hydraulique vers l'électronique et l'informatique. En effet, l'introduction de toute technologie amène à faire une comparaison avec l'ancienne. En cas de remplacement d'un système mécanique ou hydraulique par l'électronique, il est évident que la sûreté de fonctionnement doit être au moins aussi élevée qu'auparavant. En particulier, il faut exiger un niveau de sécurité au moins équivalent au niveau atteint par les systèmes mécaniques classiques. Mais une augmentation de la sécurité du véhicule (par le principe de la redondance) entraîne une complexité accrue qui peut avoir un effet négatif sur la fiabilité du véhicule. Les effets positifs sur la sécurité de conduite apportés par les systèmes embarqués informatiques peuvent donc conduire à une augmentation du nombre de défaillances techniques. Il faut bien évidemment s'assurer que les effets positifs soient plus importants que les effets négatifs. De plus, une détection d'erreur très sensible et une politique de "sûreté en présence de défaillance" augmentent la sécurité tout en diminuant la fiabilité du système. Cela sont des problèmes bien connus en sûreté de fonctionnement et ils signifient qu'un compromis entre sécurité et fiabilité doit être trouvé pour que le niveau de sécurité ne se fasse pas au détriment de la fiabilité du véhicule — il faut s'assurer que le risque technique reste négligeable devant le risque global pour que les nouveaux systèmes soient acceptés par le client.

Dans les sections suivantes, nous allons décrire l'état de l'art ainsi que l'évolution future concernant la spécification et la validation des objectifs de sûreté de fonctionnement.

I.4.1 Etat de l'art

A l'heure actuelle, les constructeurs automobiles ne sont pas astreints à des contraintes formelles concernant la spécification et la validation des objectifs de sûreté de fonctionnement. En particulier, l'architecture des systèmes informatiques dans l'automobile n'est pas certifiée vis-à-vis de la sécurité par une autorité indépendante (comme cela se fait, par exemple, dans l'aéronautique). Dans le but de satisfaire la clientèle et d'être à la hauteur de la concurrence, les constructeurs spécifient leurs propres objectifs en termes de coût et de sûreté de fonctionnement (à l'heure actuelle, essentiellement des objectifs de fiabilité). Ils établissent également des procédures de validation qui s'intègrent au développement et à la fabrication et qui doivent garantir le niveau de sûreté de fonctionnement qui a été spécifié au préalable. Les procédures de validation sont associées à l'évaluation, car l'évaluation, ordinale ou probabiliste, de la sûreté de fonctionnement permet de valider progressivement les choix de conception et de réalisation.

Les **évaluations ordinales** sont basées essentiellement sur l'emploi des AMDEC (Analyses des Modes de Défaillance, de leurs Effets, et de leur Criticité) ou sur des Arbres de Fautes (voir aussi [Villemeur 1988]). Dans le cas de l'AMDEC, on étudie pour chaque système les effets des défaillances. Par exemple, la défaillance de fonctions tels que le freinage ou la direction a des effets plus graves que la défaillance des lève-vitres. Parmi les défaillances possibles on pourra considérer par exemple : le fonctionnement intempestif, le non-fonctionnement lors de la sollicitation, la perte de la fonction après sollicitation, etc. Cela permet de faire une classification des systèmes en fonction de leur criticité. Cette classification fournit un guide pour le choix de l'architecture matérielle et logicielle supportant ces fonctions. De manière analogue à la proposition de [Torin 1992], on peut définir cinq niveaux de criticité. Pour cette classification il faut également tenir compte des différentes phases d'utilisation : le démarrage, la conduite en file, le dépassement, l'arrêt (avec moteur tournant), la manœuvre de stationnement, etc. Par exemple, une coupure du contrôle moteur peut avoir des conséquences catastrophiques si elle se produit pendant une manœuvre de dépassement. Les conditions d'utilisation peuvent avoir également une influence sur la criticité. Par exemple, la défaillance de la climatisation, jugée habituellement de criticité négligeable, prendrait un aspect plus que gênant dans un climat désertique. Le *Tableau 1-2* est un exemple de classification, établi pour des phases et des conditions d'utilisation non-extrêmes. Après l'analyse de criticité, on peut déterminer le risque correspondant en fonction de la fréquence d'apparition et de la criticité de la défaillance [Bell & Reinert 1993].

Les **évaluations probabilistes** correspondent à la prévision quantifiée des défaillances survenant en exploitation. Elles reposent sur une évaluation par simulation ou par l'analyse d'un modèle du système telle qu'une chaîne de Markov et

peuvent être menées dès la phase de conception, afin de valider les choix d'architecture. Dans l'industrie automobile, les évaluations probabilistes ne sont que très rarement employées actuellement.

Tableau 1-2 : Niveaux de criticité

Criticité	Pire effet de défaillance	Exemples
Catastrophique	La mort d'une ou plusieurs personnes	Freinage électronique, direction électronique
Critique	La blessure d'une ou plusieurs personnes	Contrôle de la traction, régulateur de vitesse, coussin gonflable
Significative	Le remorquage du véhicule	Contrôle moteur, contrôle d'assiette, suspension active
Mineure	Une gêne pour le conducteur	Limitation de vitesse, réglage de siège, rétroviseurs électriques, vitres électriques, système de diagnostic
Négligeable	La diminution du confort	Climatisation, essuie-phares, radio, téléphone

La manière la plus courante de valider l'ensemble du véhicule (et en particulier les systèmes embarqués informatiques) en vie opérationnelle et d'éliminer presque entièrement les fautes résiduelles, est de mettre en circulation une flotte de véhicules de test qui cumule, par exemple, un million de kilomètres avant le lancement de la fabrication du véhicule en série.

I.4.2 Evolution future

Lors de la spécification et validation des objectifs de sûreté de fonctionnement, il faudra prendre en compte surtout une forte intégration au niveau matériel, une complexité de plus en plus élevée au niveau logiciel et la cohabitation de logiciels de criticités différentes. Par exemple, la validation des barrières de confinement d'erreurs entre fonctions de niveaux de criticité différentes devra recevoir une attention toute particulière.

De plus, les constructeurs devront s'attendre à l'intervention d'une législation normative, voire de vérification, comme dans l'avionique [DO178 1992] ou le nucléaire [ISO_6527 1995, ISO_7385 1995] par exemple. Des organismes comme l'IEC (International Electrotechnical Commission) ou l'ISO (International Organization for Standardization) préparent actuellement des recommandations internationales qui amèneront les constructeurs et les équipementiers à revoir la façon de spécifier, de mettre en œuvre et de valider leurs systèmes [IEC_122 1995, IEC_123 1995, ISO_10605 1995, ISO_11451 1995, ISO_11452 1995, ISO_7637 1995]. Ces recommandations influeront également sur l'architecture matérielle et logicielle des systèmes pour que celle-ci soit facilement validable. La spécification et

validation des objectifs de sûreté de fonctionnement se fera probablement selon des critères qualitatifs et quantitatifs.

D'une part, les normes seront certainement d'ordre **qualitatif** parce que de telles normes sont relativement faciles à vérifier. Pour des systèmes critiques comme la direction électronique ou le freinage électronique, le constructeur devra montrer les mesures qu'il a prises pour que le système soit *tolérant à k fautes*. Pour des systèmes moins critiques tels que la climatisation, la conception d'un système *arrêt sur défaillance* serait plus appropriée. Ces critères qualitatifs ne peuvent fixer qu'une borne inférieure pour la redondance nécessaire car ils supposent que les mécanismes de détection d'erreur et de tolérance aux fautes sont parfaits.

D'autre part, dans le domaine de la sûreté, il pourrait y avoir aussi des lois ou des recommandations pour spécifier et valider des objectifs **quantitatifs**. En fonction de l'application, ces objectifs se déclinent de différentes façons. Par exemple, l'objectif de fiabilité du contrôle moteur peut être exprimé comme *taux de défaillance maximal*. Pour les systèmes qui ne sont sollicités que rarement (par exemple, le coussin gonflable), la spécification d'une *probabilité de défaillance à la demande* et d'une *probabilité de défaillance intempestive* seraient plus appropriées. Pour certains systèmes telle que la direction électronique, un *temps d'interruption maximal* devrait être spécifié.

I.5 Equipement automobile : Contraintes affectant la sûreté de fonctionnement

Les contraintes affectant la sûreté de fonctionnement d'un système sont directement liées aux fautes, car les fautes sont la cause potentielle de la non-sûreté de fonctionnement des systèmes. [Laprie *et al.* 1996] donne une classification des fautes qui est indépendante de l'application et qui est basée sur cinq points de vue différents : la cause phénoménologique (physique ou humaine), la nature (accidentelle ou intentionnelle sans volonté de nuire ou intentionnelle avec volonté de nuire), la phase de création (pendant le développement ou pendant l'opération), les frontières du système (interne ou externe) et la persistance temporelle (permanente ou temporaire). En associant ces différents points de vue et en éliminant les combinaisons qui ont peu d'intérêt on arrive à distinguer 17 classes de fautes. Les différentes classes de faute se distinguent non seulement par leur caractère, mais aussi par leur traitement de faute associé. Nous regroupons les 17 classes de faute en 10 ensembles que nous détaillons pour le domaine de l'automobile en se rapportant aux lettres indiquées sur la *Figure 1-6*.

(a) Les fautes physiques, accidentelles, de développement, internes, permanentes ou temporaires (souvent dénommées fautes de production) sont essentiellement dues au processus de fabrication et à la qualité des matériaux employés. Pour les systèmes

automobiles, ces fautes ont une grande importance suite à la production en grande série. Les erreurs dues aux fautes de production peuvent être détectées par des tests qui s'intercalent dans le processus de fabrication et qui permettent d'écarter les unités défaillantes.

(b) Les fautes physiques, accidentelles, opérationnelles, internes ou externes, permanentes (souvent dénommées fautes permanentes) sont les fautes "traditionnelles" que l'on considère en premier lors de la conception d'un système tolérant les fautes. Elles constituent un problème majeur pour les connecteurs, capteurs et actionneurs qui doivent — d'un point de vue climatique — résister à l'humidité (jusqu'à 99%) ainsi qu'à des températures variant de -40°C à +125°C. Les fautes permanentes peuvent seulement être tolérées sans dégradation fonctionnelle en utilisant des composants redondants.

(c) Les fautes physiques, accidentelles, opérationnelles, internes, temporaires (souvent dénommées fautes intermittentes) sont courantes dans les systèmes automobiles. Elles correspondent par exemple à des changements dans les paramètres d'un composant matériel dû essentiellement à des contraintes mécaniques (les vibrations et les chocs affectent surtout les capteurs, actionneurs et connexions [Zanoni & Pavan 1993]). Leur tolérance peut demander, selon leur taux de récurrence, soit de la redondance temporelle soit de la redondance structurelle.

(d) Les fautes physiques, accidentelles, opérationnelles, externes, temporaires (souvent dénommées fautes transitoires) sont en particulier dues à l'environnement électromagnétique et sont de loin le phénomène physique prépondérant, surtout pour les canaux de transmission. Par exemple, les concepteurs du protocole TTP [Kopetz & Grünsteidl 1993] supposent pour des applications automobile que les taux de faute transitoire des supports de transmission sont 10 fois plus élevés que leurs taux de faute permanente. De plus, des interférences électromagnétiques puissantes peuvent amener à des fautes transitoires corrélées affectant plusieurs sous-systèmes qui peuvent demander des procédures spéciales de recouvrement global. [Noble 1992] donne une liste des sources d'interférence et la façon dont ce problème est traité actuellement dans l'industrie automobile. La redondance temporelle peut suffire pour tolérer des fautes transitoires.

(e) Les fautes dues à l'homme, accidentelles ou intentionnelles sans volonté de nuire, de développement, internes, permanentes ou temporaires (souvent dénommées fautes de conception) sont actuellement traitées dans l'industrie automobile essentiellement par des activités de prévention et d'élimination des fautes (c'est-à-dire, la vérification et le test), approche popularisée par le passé sous l'appellation *zéro défaut* [Laprie 1993]. Malheureusement, dû à des contraintes de coût, la preuve formelle ou le test exhaustif sont seulement faisables pour les fonctions très critiques

et il est possible que des fautes résiduelles de conception soient présentes dans les fonctions moins critiques.

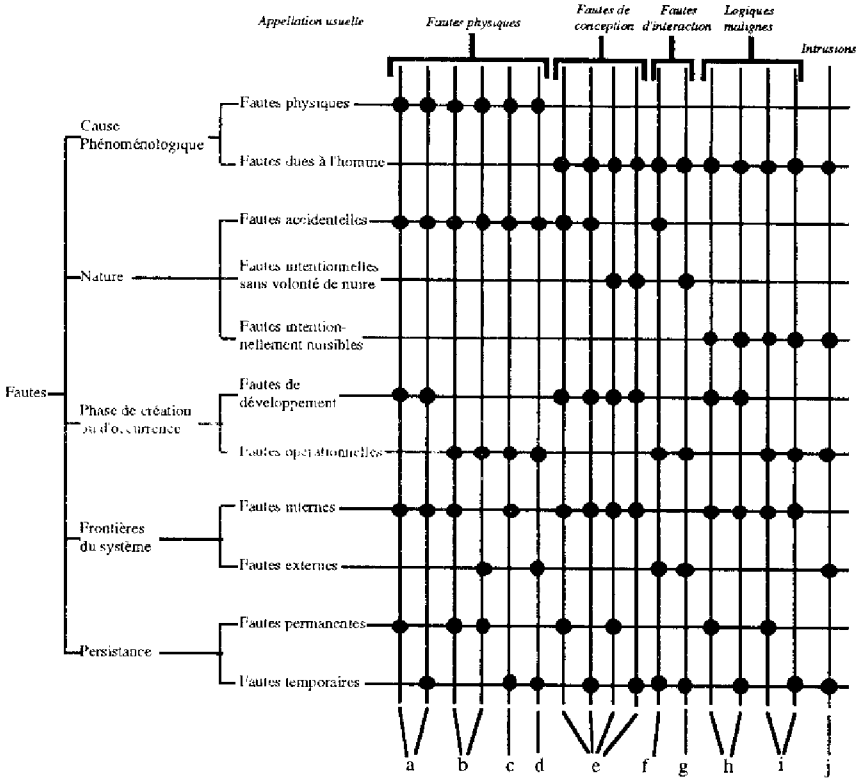


Figure 1-6 : Classes de faute (adapté de [Laprie *et al.* 1996])

Dans l'avenir, suite à la complexité et à la criticité croissantes des systèmes embarqués (et surtout de leur logiciel), les fautes de conception nécessiteront plus d'attention. Il est à noter que l'on trouve déjà des systèmes tolérant les fautes de conception par la technique de diversification, le processeur principal étant surveillé par un autre type de processeur avec un logiciel simplifié. Cependant, la tolérance aux fautes de conception par diversification est employée avec prudence, car la nécessité d'un deuxième processeur et d'une mémoire plus grande augmente sensiblement les coûts récurrents du système. De plus, la conception, l'implantation et le test d'un logiciel supplémentaire aurait une conséquence sur le coût de développement du système global.

Dans un système bien testé, la plupart des fautes de conception sont seulement activées sous des conditions très rares qui sont difficiles à reproduire. Beaucoup de fautes de conception peuvent donc être traitées et tolérées en tant que fautes

intermittentes, par exemple en appliquant les techniques de redondance temporelle telles que le recouvrement en arrière, car les conditions ont changé lors de la réexécution.

Un autre aspect à considérer par rapport aux fautes de conception résulte de l'intégration croissante des systèmes automobiles. Il sera en effet très important d'implanter des barrières de confinement d'erreurs qui empêchent la propagation des erreurs d'une fonction peu critique vers une fonction plus critique.

(f) Les fautes dues à l'homme, accidentelles, opérationnelles, externes, temporaires (souvent dénommées fautes d'interaction) sont extrêmement importantes car, contrairement à la plupart des autres technologies avancées, le véhicule automobile est mis dans les mains de personnes qui parfois ne savent pas l'utiliser correctement. Les conducteurs d'automobile — contrairement aux pilotes d'avion ou aux conducteurs de train — ne reçoivent pas de formation très spécialisée, approfondie et renouvelée régulièrement, ce qui est une des raisons du nombre élevé d'erreurs humaines conduisant à des accidents. L'introduction de nouvelles fonctions d'aide à la conduite peut, dans certains cas, avoir des effets néfastes sur le comportement des conducteurs [Savary 1993]. La confiance du conducteur dans des applications tels l'antiblocage des roues, l'anticollision ou l'amélioration de la visibilité peut favoriser la prise de risque et l'inattention par rapport aux événements survenant sur la route [Saad 1988, Hansen 1995]. L'automobile étant un système conçu pour l'utilisation par des êtres humains, la conception de l'interface homme-machine doit surtout être axée sur l'évitement des fautes d'interaction.

(g) Les fautes dues à l'homme, intentionnelles mais sans volonté de nuire, opérationnelles, externes, temporaires (souvent dénommées aussi fautes d'interaction) peuvent résulter de l'action de maintenance du concessionnaire ou de l'action non-autorisée d'un "bricoleur" qui, par exemple, viole délibérément des procédures sans réaliser les conséquences malheureuses de son action. Ce problème est accentué par l'existence d'une prise de diagnostic qui permet l'accès au système informatique. Même si actuellement le risque de telles fautes est relativement petit (par exemple, contourner les systèmes de contrôle de pollution afin d'améliorer les performances), les changements dans des futurs systèmes embarqués pourraient créer des situations plus dangereuses. Les fonctions critiques doivent donc être protégées contre des modifications illégitimes sans dégrader la fiabilité ou empêcher la maintenance prévue. Le logiciel des systèmes électroniques ne doit pas être facilement accessible ou modifiable. De bonnes solutions à ce problème, pouvant être conjuguées avec des schémas de protection cryptographique, sont un domaine intéressant pour la recherche future.

(h) Les fautes dues à l'homme, intentionnellement nuisibles, de développement, internes, permanentes ou temporaires (souvent dénommées logiques malignes)

recouvrent par exemple les chevaux de Troie, les portes dérobées et les bombes logiques ou temporelles [Landwehr *et al.* 1994]. Seulement un développeur "espion" d'une entreprise concurrente pourrait avoir l'intérêt d'implanter de telles fautes. Ces logiques malignes ne sont pas considérées dans l'automobile.

(i) Les fautes dues à l'homme, intentionnellement nuisibles, opérationnelles, internes, permanentes ou temporaires (souvent dénommées aussi logiques malignes) recouvrent par exemple les virus et les vers [Landwehr *et al.* 1994]. Ces fautes ne sont pas pertinentes dans les systèmes automobile actuels dont la mémoire n'est que rarement reprogrammable.

(j) Les fautes dues à l'homme, intentionnellement nuisibles, opérationnelles, externes, temporaires (souvent dénommées intrusions) jouent un rôle important pour les systèmes électroniques d'antivol et d'antidémarrage. La présence de tels systèmes a pour conséquence que les véhicules peuvent être volés par exemple en reproduisant le code d'accès ou en contournant tout simplement ces systèmes électroniquement. L'intégration des systèmes d'antivol dans le contrôle moteur pourrait rendre l'accès malintentionné plus difficile.

Parmi les classes de faute présentées dans ce paragraphe, seulement les fautes physiques opérationnelles, c'est-à-dire les classes (b), (c) et (d), sont considérées dans le cadre de nos travaux de recherche.

Conclusion

L'utilisation de l'informatique à bord de véhicules a apporté des améliorations importantes concernant l'augmentation du confort et des performances (par exemple : la suspension semi-active), la diminution de la pollution et de la consommation (par exemple : le contrôle moteur) ainsi que l'amélioration de la sécurité (par exemple : le coussin de sécurité, l'antiblocage des roues). La complexité matérielle et logicielle des systèmes embarqués automobiles continue de croître, ce qui pourrait conduire à une augmentation relative du nombre de défaillances techniques qui pourrait annuler les effets positifs sur la sécurité de conduite apportés par l'informatique embarquée, d'autant plus que les systèmes informatiques sont soumis à des contraintes d'environnement sévères. Nos travaux visent à s'assurer que l'informatisation des fonctions automobiles ne conduise pas globalement à une dégradation de la sûreté de fonctionnement du véhicule par rapport au véhicule équipé de commandes classiques.

L'automobile étant une industrie de production en très grande série, les coûts récurrents sont dominants lors de tout choix technique. Il est évident que les objectifs de sûreté de fonctionnement ont une influence sur le coût de l'architecture et sur le coût de la procédure de validation associée — il faut faire attention à ne pas sur-spécifier les objectifs de sûreté pour chaque fonction. La tendance actuelle en systèmes embarqués automobile est vers l'intégration, ce qui conduira certainement à

la prise en compte de fonctions de criticités différentes. Par exemple, une fonction peu critique telle que la climatisation ne nécessite ni une architecture redondante ni une validation sophistiquée, alors qu'une fonction critique telle que la direction électronique nécessite une architecture redondante et une validation rigoureuse. Par conséquent, la spécification et validation des objectifs de sûreté de fonctionnement est essentiellement déterminée par le compromis entre sûreté de fonctionnement et coût.

Dans les chapitres suivants, nous allons détailler les différents aspects intervenant dans la définition d'une architecture et dans l'évaluation de sa sûreté de fonctionnement.

Chapitre II

Architectures sûres de fonctionnement

La conception des systèmes informatiques embarqués doit permettre de respecter non seulement les objectifs fonctionnels, mais aussi les objectifs de sûreté de fonctionnement. En automobile, ces objectifs se définissent essentiellement par rapport aux fautes physiques et de nombreuses techniques ont été développées pour détecter ou plus généralement traiter les erreurs qui en résultent. Dans le cadre de notre travail, nous focalisons sur les techniques de tolérance adaptées aux fautes physiques et nous nous référons à la terminologie de la tolérance aux fautes qui est établie au paragraphe I.3.4.

A l'heure actuelle, les techniques de sûreté de fonctionnement sont appliquées **localement** et sont alors spécifiques par rapport à la fonctionnalité du module. La plupart des systèmes informatiques embarqués sur l'automobile découplent partiellement les commandes directes et interviennent donc en parallèle du système mécanique existant (par exemple : l'antiblocage des roues qui contrôle la pression dans le circuit hydraulique de freinage, ou bien la modification de la position du papillon autour de la position imposée par le câble d'accélérateur). Ces systèmes ont un comportement de sûreté en présence de défaillance, c'est-à-dire que la partie électronique devient passive après la détection d'une défaillance et conduit le système dans un mode sûr, mais dégradé, dans lequel la partie mécanique assure toujours la fonction de base. En ce qui concerne la partie mécanique et hydraulique du système, on a recours au sur-dimensionnement.

Comme déjà indiqué au paragraphe I.1, les systèmes futurs seront beaucoup plus intégrés que les systèmes actuels. En particulier, il est probable que des fonctions de criticités différentes devront interagir entre elles. Donc, une approche **globale** de la sûreté de fonctionnement sera nécessaire. Une telle approche globale vise à réduire les coûts de la tolérance aux fautes implantée tout en améliorant les performances, et elle devra également permettre la modularité fonctionnelle.

Afin de pouvoir maîtriser la complexité des systèmes électroniques et informatiques, on découpe ces systèmes généralement en sous-systèmes. La défaillance d'un sous-système peut être vue comme une faute au niveau système et

des moyens de tolérance appropriés peuvent être mis en place. Le but essentiel de ces moyens est de placer des barrières tentant d'empêcher que l'activation d'une faute d'un sous-système sous la forme d'une erreur ne se propage au système global conduisant ainsi à sa défaillance. Dans ce chapitre, nous considérons successivement quatre sous-systèmes : les capteurs, les actionneurs, les calculateurs et les liaisons de communication. Dans les paragraphes II.1 à II.4 nous exposons les défaillances et les techniques de sûreté de fonctionnement relatives à chacun de ces sous-systèmes. Dans le cinquième paragraphe, nous proposons un spectre d'architectures [Ziegler *et al.* 1994b] qui illustre différentes possibilités existantes entre une architecture entièrement fédérée à un bout du spectre et une architecture entièrement intégrée à l'autre. Ce spectre nous permet de comparer les différentes architectures par rapport à des critères qualitatifs et au coût.

II.1 Capteurs

Les capteurs constituent la source principale de défaillance des systèmes embarqués en raison des sévères contraintes d'environnement (surtout les vibrations et les perturbations électromagnétiques). L'expérience montre que les calculateurs défaillent beaucoup moins souvent que les capteurs. Les modes de défaillance des capteurs peuvent être caractérisés selon les trois points de vue présentés dans le paragraphe I.3.1. Cela nous amène à distinguer les modes de défaillance suivants :

- Les défaillances en valeur : les capteurs envoient des données erronées. Par exemple, le capteur de température moteur peut — suite à un défaut de fabrication — présenter un biais constant par rapport à la valeur réelle. Il est également imaginable que la mesure de la vitesse de roue soit momentanément modifiée par une perturbation électromagnétique.
- Les défaillances temporelles : les capteurs envoient les données trop tard. Par exemple, le capteur de la sonde lambda ne répond pas à la sollicitation du système de contrôle-moteur dans le délai de réponse spécifié. Cette défaillance peut se reproduire à chaque sollicitation, mais souvent elle peut être considérée comme une perturbation transitoire. Le problème des défaillances temporelles des capteurs est par exemple traité dans [Poledna 1995b].
- Les défaillances par arrêt : les capteurs n'envoient plus de données (silence) ou une valeur constante (figement). Par exemple, le mauvais branchement du capteur de niveau d'essence dans le réservoir peut avoir de telles conséquences.
- Les défaillances incohérentes (souvent dénommées défaillances byzantines) : au cas où un capteur est lié à plusieurs calculateurs, les défaillances des capteurs peuvent être perçues différemment par les calculateurs. Il est par exemple possible que le capteur de régime moteur envoie des valeurs

différentes aux calculateurs de contrôle moteur et de contrôle de boîte de vitesses automatique. Le problème des défaillances byzantines des capteurs est par exemple exposé dans [Poledna 1995a].

Il est possible aussi que des combinaisons entre les différents mode de défaillance se présentent. Par exemple, les données capteurs peuvent être à la fois erronées en valeur et décalées dans le temps.

De nombreuses techniques de tolérance aux fautes sont employées afin d'assurer le fonctionnement correct (ou dégradé) du système global même en présence de défaillances des capteurs :

- La réplication du capteur permet de tolérer des fautes permanentes ainsi que des fautes intermittentes à taux de récurrence élevé qui sont dues à des défaillances en valeur ou à des défaillances par arrêt du capteur.
- La relecture du même capteur permet surtout de tolérer des fautes transitoires ainsi que des fautes intermittentes à taux de récurrence bas qui sont dues à des défaillances en valeur ou à des défaillances temporelles du capteur.
- Un réseau d'entrée (filtre passe-bas) peut être prévu pour tolérer les fautes transitoires qui sont dues à des défaillances en valeur du capteur.
- Une résistance *pull-up* ou *pull-down* permet de mettre l'interface d'entrée du calculateur à un potentiel électrique prédéfini lorsque le capteur est débranché. Cela permet au calculateur de détecter une mauvaise connexion des entrées.
- Une diode de protection assure une limite supérieure pour le potentiel électrique de l'interface d'entrée du calculateur ce qui permet de tolérer la présence d'une surtension du capteur.
- Les contrôles de vraisemblance consistent à vérifier la conformité des interfaces d'entrée du calculateur par rapport à plusieurs critères : le circuit ouvert, le court-circuit à la masse, le court-circuit à l'alimentation, les fourchettes de valeurs possibles, l'écart maximal par rapport au résultat précédent, la cohérence des informations. Le rôle des contrôles de vraisemblance est de détecter ou tolérer des fautes qui sont essentiellement dues à des défaillances en valeur ou à des défaillances par arrêt du capteur.

II.2 Actionneurs

Les actionneurs constituent une autre source importante de défaillance en raison des sévères contraintes d'environnement. Les modes de défaillance des actionneurs peuvent également être caractérisés selon les trois points de vue présentés dans le paragraphe I.3.1. De manière analogue au classement des défaillances capteurs, nous pouvons distinguer les modes de défaillance actionneurs suivants :

- ❑ Les défaillances en valeur : l'actionneur agit de façon trop forte ou trop faible. Par exemple, l'injecteur débite constamment ou par moment trop d'essence dans le cylindre.
- ❑ Les défaillances temporelles : l'actionneur agit avec un décalage temporel. Par exemple, la bobine d'allumage crée la haute tension trop tôt ou trop tard.
- ❑ Les défaillances par arrêt : l'actionneur ne fait plus d'action (silence) ou une action constante (figement). Par exemple, l'actionneur de vitesse (de la boîte de vitesse automatique) ne réagit plus à la commande de changement de vitesse et reste bloqué en position d'arrêt.

De façon analogue au classement des défaillances capteurs, il est possible que des combinaisons entre les différents modes de défaillance se présentent. Par exemple, les actions des actionneurs peuvent être à la fois erronées en valeur et décalées dans le temps.

Au niveau du système global, de nombreuses techniques de tolérance aux fautes sont employées afin d'assurer le fonctionnement correct (ou dégradé) du système même en présence de défaillances des actionneurs. La technique la plus courante est la réplication de l'actionneur permettant surtout de tolérer des fautes permanentes qui sont dues à des défaillances en valeur ou à des défaillances par arrêt de l'actionneur. La réplication de l'actionneur se décline en trois variantes principales :

- ❑ Deux actionneurs sont mis en série (ET logique), c'est-à-dire leur fonctionnement correct est nécessaire pour que l'action souhaitée soit effectuée. Par exemple, afin de réduire la probabilité de gonflement intempestif du coussin gonflable, deux interrupteurs autorisant le déclenchement de l'actionneur pyrotechnique peuvent être mis en série.
- ❑ Deux actionneurs sont mis en parallèle (OU logique), c'est-à-dire le fonctionnement correct d'un seul actionneur est suffisant afin d'effectuer l'action souhaitée. Par exemple, afin d'augmenter la probabilité de gonflement du coussin gonflable lors d'une sollicitation, deux interrupteurs autorisant le déclenchement de l'actionneur pyrotechnique peuvent être mis en parallèle.
- ❑ Trois actionneurs forment un voteur, c'est-à-dire que l'action effectuée correspond à la majorité du vote (pour des signaux binaires) ou à la valeur moyenne des actionneurs (dans le cas de signaux analogiques). Par exemple, une future direction électronique pourrait consister à moyenner mécaniquement les couples exercés par trois moteurs.

On note qu'il est courant de combiner les trois formes de réplication, par exemple en mettant en série deux structures parallèles, ou bien en formant une structure parallèle à partir de deux structures série.

L'actionneur est commandé par le circuit de sortie correspondant du calculateur. Afin d'assurer le fonctionnement des circuits de commande, il est nécessaire d'effectuer différents tests :

- Les tests des sorties consistent à vérifier la conformité des interfaces de sortie du calculateur par rapport aux critères suivants : circuit ouvert et court-circuit.
- Les tests des étages de sortie : Dans le cas du coussin gonflable, on envoie un courant suffisamment petit pour ne pas déclencher l'action, mais qui permet de tester le fonctionnement des étages de sortie.

II.3 Calculateurs

Les calculateurs actuellement embarqués sur l'automobile sont généralement des microcontrôleurs qui sont essentiellement composés d'un processeur, d'une mémoire et d'interfaces. De tels composants peuvent présenter une multitude de modes de défaillance qui résultent souvent d'un enchaînement *faute --> erreur --> défaillance* relativement complexe qui est difficile à reproduire ou à analyser. Afin de pouvoir maîtriser les défaillances des calculateurs, de nombreuses techniques de sûreté de fonctionnement sont appliquées. Nous présentons dans les deux sous-paragraphes II.3.1 et II.3.2 les différents algorithmes de traitement d'erreur et de traitement de faute qui sont actuellement implantés dans les calculateurs embarqués (cf. paragraphe I.3.4 pour les définitions).

II.3.1 Traitement d'erreur

Dans ce sous-paragraphe, nous détaillons les trois parties du traitement d'erreur, à savoir la détection d'erreur, le diagnostic d'erreur et le recouvrement d'erreur.

(1) La détection d'erreur peut être effectuée de différentes façons : les contrôles temporels et d'exécution, la duplication et comparaison, ainsi que le diagnostic en ligne.

- Les contrôles temporels et d'exécution par "chien de garde" sont très couramment utilisés pour surveiller l'activité du processeur central. Le "chien de garde" doit être rafraîchi avec une fréquence fixée ; si le rafraîchissement n'a pas lieu avec la fréquence fixée suite à un dysfonctionnement de l'unité centrale, il y a génération d'un signal d'erreur. Souvent, le "chien de garde" est intégré dans un processeur de surveillance qui peut aussi exécuter un logiciel simplifié qui vérifie le calcul des sorties du processeur principal. Un autre contrôle temporel consiste à détecter la dégradation de l'horloge, pour assurer la vitesse de traitement spécifiée. Par exemple, le système de contrôle moteur est souvent composé d'un processeur avec un contrôle temporel et d'exécution.

- La duplication du calculateur permet de détecter, voire de tolérer les erreurs dues à des fautes physiques internes et externes. Dans le domaine de l'automobile, on se limite à l'heure actuelle à la technique de duplication et comparaison (architecture Duplex). Par rapport au calculateur simple avec autodétection, le calculateur dupliqué permet de disposer d'une plus grande couverture de détection d'erreur. Lors de la duplication du calculateur, le problème des erreurs de cause commune se pose. Pour empêcher la présence des mêmes fautes physiques internes dans les deux calculateurs, il peut être avantageux de diversifier les calculateurs et leur logiciel. Concernant les fautes physiques externes, il faut éviter qu'elles créent les mêmes erreurs dans les deux calculateurs. Cela peut être obtenu par la répartition géographique des calculateurs, le décalage temporel de leurs traitements, ou encore la diversification matérielle et logicielle. Par exemple, le système d'antiblocage des roues fait souvent appel à une structure avec deux processeurs qui effectuent une comparaison mutuelle, car il est important d'assurer une bonne couverture de détection d'erreurs en raison de l'objectif principal de sécurité.
- Le diagnostic en ligne est utilisé sous les deux formes suivantes :
  La *première* est la présence de programmes d'autotest qui sont activés lors de la mise en marche, dans le but de révéler des fautes dormantes au niveau des capteurs et actionneurs ainsi que de la mémoire, fautes qui pourraient produire des erreurs en fonctionnement opérationnel. Les programmes d'autotest interviennent alors en alternance avec le mode opérationnel du système. Il est important de noter que les programmes d'autotest ne sont pas efficaces vis-à-vis des fautes temporaires.
  La *deuxième* est la présence d'un processeur de surveillance qui peut être utilisé non seulement pour détecter des erreurs, mais aussi pour effectuer le diagnostic de faute.

Pendant l'exécution d'un logiciel sur un calculateur, d'autres contrôles sont effectués. Par exemple, lors d'un accès à la mémoire, on vérifie si l'adresse mémoire se trouve dans la plage prédéfinie. Un autre exemple est le calcul d'une somme de contrôle sur la mémoire morte, pour détecter des erreurs de mémorisation affectant le logiciel.

En général, la détection d'une erreur doit être confirmée pendant plusieurs cycles de calcul successifs, avant qu'un diagnostic et un recouvrement soient entrepris.

(2) Le diagnostic d'erreur consiste à estimer les dommages créés par l'erreur qui a été détectée. Les systèmes actuellement embarqués sont relativement monolithiques et indépendants les uns des autres. Par conséquent, la fonction de diagnostic d'erreur n'est pas identifiable dans ces architectures. Cependant, l'intégration fonctionnelle peut faire évoluer cet état de fait dans l'avenir.

(3) Le recouvrement d'erreur consiste à substituer à l'état erroné un état exempt d'erreur où l'état exempt d'erreur peut être atteint par trois techniques différentes (généralement spécifiques au système), à savoir la reprise, la poursuite ou la compensation.

Actuellement, les architectures embarquées sur l'automobile sont basées sur des structures de type Simplex ou Duplex où l'on peut implanter soit la technique de reprise, soit la technique de poursuite. La plupart des systèmes embarqués automobile sont des systèmes dont les sorties sont fonction uniquement de l'état actuel des entrées (la vitesse des roues, la température du moteur, etc.). C'est pour cela que la technique de recouvrement d'erreur par poursuite est généralement utilisée. Par exemple, dans le cas du contrôle moteur (voir aussi [Behnke *et al.* 1993, Sellner & Merker 1993, Breitwieser *et al.* 1994]), une réinitialisation du système est tentée. D'un point de vue informatique, cela correspond à un traitement d'exception. Lors de l'évaluation de la sûreté de fonctionnement des architectures dans le chapitre III de ce mémoire, nous allons proposer une façon de modéliser la tentative de recouvrement d'erreur par poursuite.

Dans le futur, il est possible que des architectures du type "Bi-Duplex" ou "Triple Modular Redundancy" (triplication et vote) soient introduites pour des systèmes nécessitant la tolérance aux fautes, tels qu'un futur système de direction électronique. Ces techniques permettraient de compenser l'erreur (compensation dynamique pour le Bi-Duplex et compensation statique pour le TMR). Ces architectures seront également modélisées dans le chapitre III.

II.3.2 Traitement de faute

Après le recouvrement d'erreur on applique le traitement de faute, dont les trois parties sont détaillées dans ce sous-paragraphe, à savoir le diagnostic de faute, la passivation de faute et la reconfiguration du système.

(1) Le diagnostic de faute consiste à analyser la faute. Par exemple, dans le cas du contrôle moteur, la tentative de réinitialisation est suivie d'un diagnostic afin de déterminer le caractère temporel de la faute : si la même faute persiste, alors on suppose qu'il s'agit d'une faute permanente et on essaye de la passiver. Dans ce cas, le conducteur est généralement averti de la défaillance associée et l'erreur correspondante est stockée dans la mémoire qui est accessible par l'interface de diagnostic (lors de la prochaine maintenance).

(2) La passivation de faute consiste à empêcher une nouvelle activation des fautes. Dans la plupart des cas, on peut déclencher un mode dégradé. Par exemple, dans le cas du système de contrôle moteur, certains capteurs défaillants peuvent être "remplacés" par des valeurs par défaut. Dans le pire des cas, on arrête le système entièrement, c'est-à-dire que le véhicule doit être remorqué. Par contre, dans le cas

du système d'antiblocage des roues (voir aussi [Rittmannsberger 1988, Meyna & Gerstenmeier 1992]), la passivation de faute consiste toujours à supprimer la fonction d'antiblocage.

(3) La reconfiguration du système : à l'heure actuelle, la structure matérielle et logicielle des systèmes embarqués automobiles est figée pour toute la vie opérationnelle et on ne peut souvent que modifier leur fonctionnalité. Cela est par exemple le cas pour le système d'antiblocage des roues, où une commutation sur le freinage classique peut être entreprise après avoir passivé la faute permanente du système électronique.

Dans l'avenir, l'intégration fonctionnelle conduira probablement à des systèmes plus souples, permettant par exemple la reprogrammation de la mémoire ou la réallocation des processeurs aux différentes tâches en cas de défaillance, permettant ainsi de modifier la répartition fonctionnelle du système global.

II.4 Communication multiplexée

Afin de faire face à la future intégration fonctionnelle des systèmes, on a besoin d'un support de communication entre les systèmes. Ce support est de plus en plus souvent réalisé par un réseau multiplexé. Ces dernières années, les constructeurs automobiles ont presque tous développé des réseaux multiplexés spécifiques pour répondre à des objectifs de sûreté de fonctionnement, de fonctionnalité et de coût. Il existe alors à l'heure actuelle une multitude de protocoles différents. Ces protocoles sont incompatibles entre eux et il est évident que des efforts de standardisation seront nécessaires pour parvenir à interconnecter des systèmes embarqués provenant de fournisseurs différents et utilisables par des constructeurs différents. Parmi ces protocoles, certains semblent s'imposer sur le marché de l'automobile, d'autres n'ont été appliqués que chez un seul constructeur et certains n'ont jamais été produits en série.

La SAE (Society of Automotive Engineers) propose de classer les réseaux multiplexés en trois classes [SAE_J1213/1 1994, SAE_J2057/1 1994]. Premièrement, les réseaux de la *classe A* doivent remplacer les connexions point-à-point entre les systèmes classiques, ceci dans le but surtout de réduire le câblage. Les réseaux de *classe A* ont généralement un débit faible (< 10 Kbit/s). Deuxièmement, les réseaux de *classe B* ont pour objectif d'éliminer des capteurs et actionneurs en facilitant leur partage par plusieurs systèmes. Le débit des réseaux de *classe B* varie entre 10 Kbit/s et 125 Kbit/s. Enfin, les réseaux de *classe C* sont destinés à véhiculer des informations temps réel entre des systèmes de contrôle-commande avec une vitesse très élevée (débit entre 125 Kbit/s et 1 Mbit/s), ceci dans le but de faciliter la répartition fonctionnelle et de réduire le câblage.

Le sous-paragraphe II.4.1 donne une liste des réseaux multiplexés qui ont été (ou sont) étudiés dans le cadre d'une application automobile. La structure de ces réseaux étant généralement basée sur le modèle OSI (Open System Interconnection) de l'ISO (International Organization for Standardization) [ISO_7498 1995], nous présentons ce modèle dans le paragraphe II.4.2. Dans les sous-paragraphes II.4.3 et II.4.4, nous nous intéressons surtout aux concepts CAN et J1850 qui sont déjà normalisés et qui risquent de s'imposer sur le marché. Le protocole TTP nous intéresse également, car il se distingue des autres par son approche de sûreté de fonctionnement (cf. paragraphe II.4.5).

II.4.1 Liste des réseaux multiplexés

La liste suivante donne les protocoles les plus connus actuellement dans l'automobile, les trois premiers protocoles de la liste étant détaillés dans des sous-paragraphes ultérieurs.

- **Le "CAN" :** le protocole CAN ("Controller Area Network") a été essentiellement spécifié et développé par les sociétés Bosch et Intel (voir [Bosch 1990] ou [SAE_J1583 1994] pour la spécification). A l'heure actuelle, il semble que le protocole CAN s'impose non seulement sur le marché de l'automobile, mais aussi dans d'autres domaines industriels [Zeltwanger 1994]. Dans la littérature, le CAN est souvent décrit comme un réseau des *classes B ou C*. Il est désormais normalisé par l'ISO ([ISO_11898 1995]) et équipe surtout les véhicules haut de gamme de Mercedes et de BMW. Le paragraphe II.4.1 donne plus de détails sur le protocole CAN.
- **Le "Class B Data Communication Network Interface" ou "J1850" :** il s'agit ici d'un protocole de *classe B* qui a été essentiellement développé pour le marché automobile américain. Sa spécification est élaborée par la SAE [SAE_J1850 1994] et il sera probablement adopté par les constructeurs Ford, General Motors et Chrysler [Hansen 1994b]. Dans le paragraphe II.4.2, nous préciserons le fonctionnement de ce protocole.
- **Le "TTP" :** l'université technique de Vienne (Autriche) propose un protocole synchrone [Kopetz & Grünsteidl 1994], le TTP ("Time Triggered Protocol"). Ce protocole permet surtout de garantir des temps de latence maximaux des messages. Il semble que les constructeurs d'automobile sont de plus en plus intéressés par une telle approche qui remplirait les objectifs de la *classe C*. Le paragraphe II.4.3 donne plus de précisions sur le fonctionnement du protocole TTP.
- **Le "VAN" :** le protocole VAN ("Vehicle Area Network") a été développé par les sociétés Renault et Peugeot. Il est normalisé en France [AFNOR 1990] ainsi qu'au niveau international [ISO_11519 1995]. Son fonctionnement est

également décrit dans [Herbault 1991, Ortis *et al.* 1992]. Une originalité du VAN consiste à pouvoir commuter entre une communication sur deux fils (paire torsadée, blindée ou non-blindée) et une communication dégradée sur un fil simple, l'autre fil étant par exemple court-circuité à la masse ou coupé. Nous n'allons pas présenter de détails sur ce protocole de *classe B*, car les sociétés Renault et Peugeot ont décidé récemment d'abandonner le VAN en faveur du CAN. Par ailleurs, le protocole VAN ressemble étroitement aux protocoles J1850 et CAN.

- **Le "ABUS" :** la société Volkswagen a développé le protocole ABUS ("Automotive Bit-serial Universal-interface System") pour des applications de *classe B*, mais à notre connaissance il n'a jamais été implanté en série. Un bref aperçu du protocole ABUS peut être trouvé dans [SAE_J2056/2 1994].
- **Le "Token Slot Device" :** ce protocole est proposé par la société General Motors. Il a été développé pour des applications de la *classe C* et il permet notamment de garantir des temps de latence maximaux pour tous les messages circulant sur le réseau. Il semble que ce réseau serait abandonné en faveur du J1850. Une description très détaillée du Token Slot Device peut être trouvée dans [SAE_J2106 1994].
- **Le "CCD" :** le protocole CCD ("Chrysler's Collision Detection") — appartenant à la *classe B* — est implanté actuellement dans plusieurs véhicules de série de la société Chrysler (voir [Fassnacht & Madden 1988, Hansen 1994a, SAE_J1567 1994]). Mais le constructeur américain a apparemment décidé de remplacer le CCD par le protocole J1850.
- **Le "I-Bus" :** le protocole I-Bus ("Instrumentation communication bus") est un protocole de *classe A*. Il a été développé par BMW pour le multiplexage du tableau de bord (compteur de vitesse, poste radio, système de navigation, etc.) et des fonctions châssis (vitres et rétroviseurs électriques, fermeture centralisée, etc.). Le I-bus a été introduit dans le modèle 850i, mais il semble que le constructeur bavarois abandonne ce concept en faveur du CAN. Une brève description du I-Bus est donnée dans [Mahalek 1992].
- **Le "RTA" :** le protocole RTA ("Real-Time Architecture") de Saab et Volvo est présenté dans [Lawson *et al.* 1992]. C'est un protocole de *classe B*. A l'heure actuelle, le concept du RTA n'a pas encore été produit en série.

Parmi les protocoles présentés, certains sont normalisés. De plus, il existe une recommandation de la SAE pour les réseaux embarqués dans les poids lourds qui définit pour chaque message échangé : sa longueur, son type, sa précision, ses valeurs limites, sa période d'envoi et sa priorité [SAE_J1587 1994].

II.4.2 Le modèle OSI

Le modèle OSI vise la normalisation des concepts permettant d'interconnecter des calculateurs hétérogènes géographiquement répartis. Il consiste en une hiérarchie de communication à sept couches où chaque couche fournit des services à la couche immédiatement supérieure. Parmi les sept couches du modèle OSI, les réseaux automobiles reprennent les deux couches les plus basses de l'architecture OSI, soit — dans l'ordre de l'hiérarchie, de bas en haut — la couche physique et la couche de liaison. Ces deux couches constituent le minimum nécessaire pour pouvoir échanger des informations entre les stations. La couche physique fournit les moyens technologiques et les procédures pour établir, maintenir et libérer des connexions physiques entre des entités de la couche liaison. La couche de liaison fournit les moyens et les procédures nécessaires pour établir, maintenir et libérer une ou plusieurs liaisons entre des entités de la couche immédiatement supérieure. Elle peut être subdivisée — dans l'ordre de l'hiérarchie, de bas en haut — en deux sous-couches qu'on appelle MAC ("Medium Access Control") et LLC ("Logical Link Control").

II.4.3 Le CAN (Controller Area Network)

Le fonctionnement du CAN ainsi que les circuits électroniques correspondants sont détaillés dans [Dais & Unruh 1992a, Dais & Unruh 1992b, Paret 1992c, Paret 1992b, Paret 1992a, Paret 1993]. Les trois couches (physique, MAC, et LLC) sont décrites dans les trois sous-paragraphes suivants, le quatrième sous-paragraphe étant dédié aux techniques de tolérance aux fautes.

II.4.3.1 La couche physique

La couche physique décrit les caractéristiques physiques du réseau, elle concerne principalement le raccordement (médium et connectique), le couplage électrique, l'encodage des informations et les mécanismes de détection de conflits.

Dans le cas du CAN, la transmission des messages peut être effectuée sur deux fils blindés ou non-blindés et la vitesse maximale de transmission est de 1 Mbit/s. Le codage des bits est du type NRZ ("No Return to Zero"), utilisant la méthode d'insertion de bits dans le flot des données : lorsqu'il y a plus de cinq bits consécutifs de même valeur, le transmetteur intercale un bit de valeur opposée qui est reconnu et enlevé par le récepteur. Cela assure qu'un état stable de plus de cinq bits consécutifs ne se présente jamais sur le bus.

II.4.3.2 La couche MAC

Cette couche contrôle l'accès à la voie de communication, elle sert à coordonner l'utilisation du réseau entre toutes les stations. En effet, le CAN est un bus de

transmission série à accès asynchrone. Cela veut dire que n'importe quel nœud peut transmettre lorsque le bus est disponible. Le protocole est basé sur le principe de la *diffusion générale*, c'est-à-dire que lors de la transmission, aucune station n'est adressée en particulier, mais le contenu de chaque message est explicité par un identificateur reçu de façon univoque par toutes les stations. Grâce à cet identificateur, les stations, qui sont en permanence à l'écoute du réseau, reconnaissent et traitent les messages qui les concernent et ignorent les autres. Dans la littérature, on parle également d'un *adressage logique*. L'adressage logique assure une grande flexibilité de configuration. Il est possible d'ajouter des stations réceptrices à un réseau CAN sans modifier les stations existantes et c'est pour cela que l'on parle aussi de *système ouvert*.

L'identificateur de chaque message détermine également sa priorité. Ces priorités sont attribuées lors de l'analyse conceptuelle du réseau, au moyen de valeurs binaires, et ne peuvent donner lieu à aucune modification dynamique. Dans le cas de transmissions simultanées, les stations concernées entrent en compétition et un arbitrage est effectué par l'intermédiaire des priorités (on parle également de *priorité par entête à forçage*). En fait, les stations reçoivent un "ET" logique des signaux transmis sur la voie de communication. Seules les stations émettant un bit de l'identificateur égal à "0" (appelé bit dominant) continuent à émettre tandis que les stations émettant un bit égal à "1" (appelé bit récessif) détectent un conflit. La détection de conflit est basée sur des mécanismes de la couche physique : en fait, lors de l'accès simultané d'un bit "1" et d'un bit "0" au bus, la valeur "0" écrase la valeur "1" sur le bus. Cela veut dire que les stations qui ont tenté d'envoyer le bit "1", relisent un bit "0" et, ayant détecté par conséquent la perte de l'arbitrage, elles cessent immédiatement leurs émissions (voir *Figure 2-1*). Le message avec l'identificateur le plus bas est alors le plus prioritaire et gagne la compétition ; les autres stations effectueront une nouvelle tentative plus tard. Il s'agit d'une technique d'arbitrage similaire à celle présentée dans [Caumont *et al.* 1982].

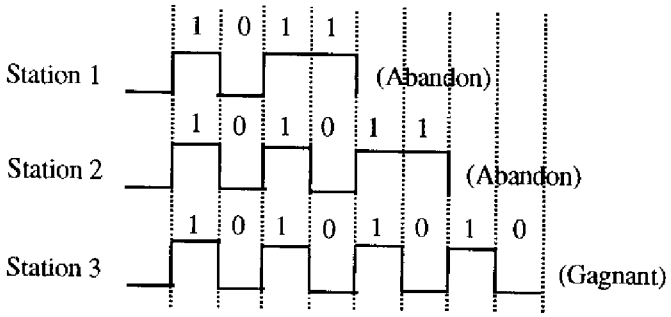


Figure 2-1 : Principe de l'arbitrage sur le bus

Dans le cadre de la sûreté de fonctionnement, on s'intéresse surtout au temps de latence, c'est-à-dire le temps qui s'écoule entre l'envoi d'un message et sa réception. Parmi tous les messages susceptibles d'être envoyés, on peut borner le temps de latence seulement pour le message le plus prioritaire : sous l'hypothèse que le bus est libre, c'est le nombre de bits maximal du message multiplié par le temps par bit. Par exemple, le temps de latence d'un message de 32 bits se calcule à 64 µs pour une vitesse de 500 Kbit/s.

Le fait qu'on ne puisse pas spécifier un temps de latence maximal pour chaque message envoyé est un grand inconvénient du point de vue de la sûreté de fonctionnement. C'est pour cela que des études sont menées afin d'implanter un protocole qui serait basé sur le CAN et qui permettrait de borner les temps de latence. Par exemple, [Tindell & Burns 1994] montrent que le temps de latence peut être borné sous les hypothèses que chaque station n'envoie que des messages périodiques et que le nombre de messages d'erreur est limité. Au cas où les stations envoient des messages apériodiques, il faut poser l'hypothèse que le temps entre l'envoi de deux messages apériodiques successifs présente une borne inférieure.

La couche MAC gère non seulement la mise en file d'attente, l'envoi et la réception des trames de données, mais aussi l'envoi de trames spéciales après l'occurrence d'une erreur ainsi que les retransmissions de trames.

II.4.3.3 La couche LLC

Cette couche permet d'établir des liaisons logiques entre stations. Elle enveloppe les données transmises sur le réseau de descripteurs, formant ainsi des trames. La fonction de ces trames est d'établir un protocole de communication et de détecter (et recouvrir) des erreurs. Quatre types de trame peuvent être transmis sur le bus : la trame de données, la trame de demande de données, la trame d'erreur et la trame de surcharge. Ces quatre trames sont détaillées par la suite :

1) **La trame de données** ("Data Frame") est la trame principale, et elle sert à envoyer des informations d'une station vers une autre station. La trame de données est constituée de plusieurs champs :
   - 1 bit de début de trame : *SOF* ("Start Of Frame"). Il s'agit d'un bit dominant signalant le début d'une nouvelle trame sur le bus et permettant aux stations de se synchroniser sur le premier front du bit.
   - 12 bits d'identificateur : *ID* ("Identifier"). Celui-ci contient 11 bits d'identification du message et 1 bit appelé *RTR* ("Remote Transmission Request"). Les 11 premiers bits déterminent la priorité du message (voir paragraphe II.4.3.2). La valeur du bit *RTR* détermine s'il s'agit d'une trame de données ou d'une trame de demande de données.

La longueur de l'identificateur est de 12 bits en format standard du CAN. Il existe aussi un format étendu du CAN, pour lequel l'identificateur a la longueur totale de 32 bits.

- 6 bits de contrôle : *CTR* ("Control"). Le *CTR* comprend 1 bit pour distinguer entre format standard et format étendu, 1 bit non utilisé, ainsi que 4 bits appelés *DLC* ("Data Length Code") indiquant le nombre d'octets de données.
- 0-64 bits de données : *DATA*. C'est le champ contenant l'information utile qui doit être envoyée d'une station vers une autre station. Dans le cas des systèmes embarqués sur l'automobile, cela peut être par exemple la valeur du régime moteur qui est diffusée sur le bus par le système de contrôle moteur. Cette valeur peut être utilisée par exemple par le système de contrôle de la boîte de vitesses automatique.
- 16 bits de contrôle de redondance cyclique : *CRC* ("Cyclic Redundancy Check"). Le *CRC* est composé de 15 bits qui contrôlent les champs *SOF, ID, CTR, DATA* et le *CRC* même. A la fin de ces 15 bits on rajoute 1 bit que l'on appelle *DEL* ("Delimiter") et qui sert à indiquer la fin du champ *CRC* et qui est toujours un état récessif. Un concept fondamental caractérisant le *CRC* est la *distance de Hamming* entre deux mots binaires qui correspond au nombre de bits par lesquels les deux mots diffèrent. On définit la *distance d'un code* comme la distance de Hamming minimale entre deux mots quelconques valides du code. Pour pouvoir détecter e erreurs, la distance du code doit être supérieure ou égale à $e+1$. La distance de Hamming du CAN est égale à 6, c'est-à-dire qu'il peut détecter jusqu'à 5 bits erronés. Le code *CRC* comprend également un test de parité qui permet de détecter n'importe quel nombre impair de bits erronés.
- 2 bits d'acquittement : *ACK* ("Acknowledge"). Le premier bit est destiné à l'acquittement, le deuxième signale la fin de l'acquittement par un état récessif et s'appelle *DEL* ("Delimiter"). Les stations ayant reçu une trame non corrompu, envoient un bit dominant dans le champ *ACK*. Les autres stations envoient un bit récessif dans ce champ. Il suffit alors qu'une seule station ait reçu la trame correctement pour acquitter la trame (on parle aussi d'un *accusé de réception positif*). L'absence de l'acquittement peut être due à différentes situations : tous les récepteurs ont détecté une erreur de transmission, ou le champ d'acquittement est erroné, ou il n'y a pas de récepteurs sur le bus.
- 7 bits de fin de trame : *EOF* ("End of Frame"). Ce champ signale la fin d'une trame.

- ❑ 3 bits d'espace intertrame : *IFS* ("Inter Frame Space"). Ce champ est nécessaire pour garantir un espace minimal entre deux trames successives.

Une trame de données contient donc, au total, de 47 à 111 bits en format standard et de 47 à 131 bits en format étendu.

2) **La trame de demande de données** ("Remote Frame") est envoyée par une station qui désire recevoir des informations d'une autre station sous forme d'une trame de données. La trame de demande de données a le même identificateur que la trame de données correspondante. Elle est constituée des champs suivants : 1 bit *SOF*, 12 bits de *ID*, 6 bits de *CTR*, 16 bits de *CRC*, 2 bits de *ACK*, 7 bits de *EOF* et 3 bits de *IFS*, soit 47 bits au total.

3) **La trame d'erreur** ("Error Frame") : les trames circulant sur le bus sont constamment observées par toutes les stations. La station qui détecte en premier une erreur dans la trame présente sur le réseau, envoie une trame d'erreur. La transmission de la trame d'erreur est immédiate, elle interrompt la transmission en cours. Ce mécanisme évite la réception de la trame erronée par les autres stations. Donc, sous les hypothèses que la trame d'erreur n'est pas affectée elle-même par une erreur et que toutes les stations connectées au réseau sont en mesure de recevoir des messages, toutes les stations savent que la précédente trame de données était erronée. La station qui avait envoyé la trame erronée fait une nouvelle tentative d'envoi plus tard. La trame d'erreur est constituée des champs suivants :

- ❑ 6-12 bits de drapeau d'erreur ("Error Flag"). La station qui a détecté l'erreur en premier envoie 6 bits dominants qui sont interprétés par les autres stations comme une violation de la règle d'insertion des bits (voir couche physique pour plus de détails). Il est possible qu'une autre station détecte également une erreur un peu plus tard et qu'elle envoie par conséquent six bits dominants. On peut alors avoir une superposition de plusieurs drapeaux d'erreur qui ont une longueur maximale de 12 bits.
- ❑ 7 bits récessifs de délimiteur ("Error Delimiter") qui indiquent la fin de la trame d'erreur.

L'envoi de trames d'erreur est très efficace et il assure une cohérence pour le réseau global : toutes les stations ont la même vue de l'état du réseau. Par contre, il est possible qu'une station défaillante envoie constamment des trames d'erreur et bloque ainsi la communication en cours. Le réseau serait alors bloqué même pour l'envoi de trames incorrompues. C'est pour cela que les stations disposent de mécanismes d'autosurveillance permettant de détecter les fautes permanentes et de déconnecter la station défaillante du bus.

4) **La trame de surcharge** ("Overload Frame") a deux fonctions : **a)** elle indique qu'une station n'est pas prête à recevoir des données ou **b)** elle signale qu'un bit dominant a été détecté pendant l'espace intertrame. La trame de surcharge peut être envoyée au plus tôt lors du premier bit du champ *IFS*. La trame de surcharge a la même structure que la trame d'erreur, c'est-à-dire que les six premiers bits dominants indiquent aux stations une surcharge. Toute autre station recevant l'indication de surcharge transmet également une trame de surcharge. On peut donc avoir entre 6 et 12 bits de drapeau de surcharge ("Overload Flag") qui sont suivis de 7 bits récessifs indiquant la fin de la trame de surcharge ("Overload Delimiter"). De manière analogue à l'envoi de la trame d'erreur, il est possible qu'une station défaillante envoie constamment une trame de surcharge, bloquant ainsi toute communication sur le réseau.

On peut calculer le surcoût ("overhead") de chaque trame : c'est le rapport entre le nombre de bits encadrant (*SOF, ID, EOF, ...*) et la longueur totale de la trame. Par exemple, dans le cas de la trame de données, le surcoût est fonction de la longueur des données et il varie entre 47/(64+47) = 42 % et 47/(0+47) = 100 % (format standard).

II.4.3.4 Les techniques de tolérance aux fautes

Dans ce paragraphe, nous résumons les différentes techniques de tolérance aux fautes que nous avons en partie déjà indiquées dans les paragraphes précédents (voir également [Mathony *et al.* 1990]) :

- Le code *CRC* : la station réceptrice compare le *CRC* reçu avec celui qu'elle calcule elle-même. Lorsque les deux codes *CRC* sont différents, la station réceptrice indique une erreur de transmission. Le code *CRC* peut détecter jusqu'à 5 bits erronés (cf. paragraphe II.4.3.3, trame de données).
- Le contrôle de la structure de trame est effectué au transmetteur et au récepteur. Ils examinent la structure de la trame en vérifiant les champs de format fixe (*EOF, IFS, ACK_DEL, CRC_DEL*) ainsi que la longueur de la trame.
- Le contrôle d'acquittement est exécuté par le transmetteur en vérifiant la présence d'un bit dominant dans le champ *ACK*. La présence d'un bit récessif dans *ACK* peut résulter d'une erreur de transmission reconnue par tous les récepteurs, d'une altération du champ *ACK*, ou de l'absence de récepteurs (cf. paragraphe II.4.3.3, trame de données).
- Le contrôle de codage des bits vérifie la règle d'insertion des bits (cf. paragraphe II.4.3.1). Lorsqu'une station (émettrice ou réceptrice) observe plus de cinq bits consécutifs de la même valeur, une erreur est signalée.
- Le contrôle du niveau électrique : en fait, la station émettrice contrôle en permanence le niveau électrique sur le réseau. Si le niveau électrique observé

ne correspond pas au niveau électrique envoyé, une erreur est signalée sauf pendant les phases d'arbitrage (cf. paragraphe II.4.3.2) et d'acquittement (cf. paragraphe II.4.3.3, trame de données).

Dans chaque station, il y a un indicateur pour les erreurs de transmission et un indicateur pour les erreurs de réception. La valeur de chaque indicateur est augmentée ou diminuée en fonction du type d'erreur détectée. La valeur de ces indicateurs détermine le mode de fonctionnement du nœud. Trois modes sont possibles : 1) Le mode normal ; 2) Le mode erreur, c'est-à-dire l'envoi d'une trame d'erreur ; 3) Le mode absence, c'est-à-dire le nœud n'envoie et ne reçoit aucun message (voir [SAE_J1583 1994] pour plus de détails). Au niveau de chaque station, il faut surtout éviter la modification erronée du champ d'acquittement ainsi que l'envoi trop répétitif de la trame d'erreur. Cela signifie que les stations doivent avoir un comportement de *silence sur défaillance*.

II.4.4 Le J1850

Le protocole J1850 peut être structuré — de façon semblable au CAN — en trois couches, à savoir la couche physique, la couche MAC et la couche LLC. Ces couches sont décrites dans les trois sous-paragraphes suivants, le quatrième sous-paragraphe étant dédié aux techniques de tolérance aux fautes.

II.4.4.1 La couche physique

A l'heure actuelle, il est prévu de produire deux versions du J1850, l'une travaillant à une vitesse de transmission de 10,4 Kbit/s, l'autre à 41,6 Kbit/s ; dans un avenir plus lointain, les vitesses de transmission seront probablement à 41,6 Kbit/s et 125 Kbit/s. Les différentes versions du J1850 se distinguent aussi légèrement par leur implantation physique, permettant ainsi d'adapter le protocole aux besoins spécifiques de l'application. La transmission est effectuée sur un fil simple, mais on peut transmettre sur une paire de fils pour disposer de la redondance (comme dans le VAN). Le codage des bits est du type PWM ("Pulse Width Modulation") ou VPW ("Variable Pulse Width Modulation").

II.4.4.2 La couche MAC

L'accès au bus est basé sur le même principe que dans le CAN, c'est-à-dire le message est diffusé sur le bus avec un adressage logique et la résolution de conflit entre deux messages simultanés se fait par un arbitrage sur le bus. A la différence du CAN, l'arbitrage s'étend sur toute la trame dans la mesure où la station émettrice abandonne simplement l'émission de la trame dès qu'elle détecte une différence entre le bit envoyé et le bit relu. C'est pour cela que l'envoi d'une trame d'erreur n'est pas prévu dans le protocole J1850.

II.4.4.3 La couche LLC

La trame enveloppant le message envoyé est constituée des champs suivants : 2 bits de début de trame, 0-64 bits de données (contenant également l'identificateur de message qui — de manière analogue au CAN — détermine la priorité), 8 bits de *CRC* qui portent sur les bits de données (la distance de Hamming est égale à 3 et il n'y a pas de test de parité) et 1 bit de fin de données.

Les stations ayant détecté le bit de fin de données, peuvent immédiatement transmettre une trame de réponse dont la fin est indiquée par un bit récessif. Dans la littérature, on parle aussi de "réponse dans la trame", mais en fait il s'agit d'une concaténation de deux trames. Cette *trame de réponse* peut avoir trois formes différentes :

1) Elle peut être un acquittement provenant d'une ou de plusieurs stations. En fait, les stations qui reçoivent la trame de données correctement, envoient une trame de réponse afin d'acquitter le message reçu. Dans ce cas, il s'agit soit d'un octet d'identification soit d'une concaténation de plusieurs octets d'identification (l'octet d'identification est typiquement l'adresse physique de la station réceptrice), suivi d'un bit récessif indiquant la fin de la trame de réponse. Une station qui reçoit une trame de données erronée, doit ignorer cette trame et ne doit pas envoyer de trame de réponse ; par ailleurs, l'envoi d'une trame d'erreur n'est pas prévu (cf. paragraphe II.4.4.2). En cas de collision entre les stations qui désirent envoyer l'octet d'identification, un processus d'arbitrage sur le bus — de façon analogue au processus d'arbitrage des trames de données — est déclenché et les stations ayant perdu la compétition font une nouvelle tentative d'envoi d'acquittement immédiatement après la transmission de l'identificateur présent.
   Contrairement au CAN, la concaténation des acquittements pourrait permettre de localiser des stations défaillantes. Par exemple, la station A diffuse une trame sur le bus et elle sait que cette trame doit être reçue par les stations B, C et D. Cette connaissance pourrait être codée dans l'identificateur (ce qui correspondrait à un *adressage physique*). La station A détecterait par exemple qu'elle a reçu les acquittements des stations B et C, mais pas de la station D. Elle en conclurait que B et C ont correctement reçu le message, et que D ne l'a peut-être pas reçu. Néanmoins, dans le protocole J1850 on ne profite pas de cette possibilité et on fait l'hypothèse simplificatrice que l'acquittement d'une seule station est suffisant pour conclure sur le fonctionnement correct du réseau. Dans notre exemple, cela signifie que le message serait perdu pour la station D.

2) La trame de réponse peut être également constituée de 0-64 bits de données suivi d'un octet de *CRC* (portant sur les données) et d'un bit récessif indiquant

la fin de la trame de réponse. Dans ce cas, et sauf violation des hypothèses, une compétition entre plusieurs stations n'est pas possible.

3) Lorsqu'il n'y pas de station qui transmet une réponse, le bus reste dans l'état récessif pendant au minimum un temps de bit, signalant ainsi la fin définitive de la trame.

Dans tous les trois cas de figure, le bit de fin de trame est suivi par 2 bits d'espace intertrame.

II.4.4.4 Les techniques de tolérance aux fautes

Mis à part le *CRC* (cf. paragraphe II.4.4.3), les autres techniques de tolérance aux fautes employées sont les suivantes :

- Le contrôle de la plage dynamique : il est possible que des interférences électromagnétiques mettent la station réceptrice au-dehors de sa plage dynamique d'opération, le récepteur ne décode plus correctement la trame. Un dispositif approprié doit garder la sortie du récepteur dans l'état précédant le dépassement pendant la durée de l'interférence. Lorsque la durée de l'interférence est relativement courte, on doit pouvoir retrouver la valeur correcte de la sortie, sinon une erreur est signalée.
- Le contrôle de codage des bits correspond à vérifier que les mots reçus appartiennent au code spécifié. Un code qui n'appartient ni au "0" ni au "1" est signalé comme erreur.
- Le contrôle du niveau électrique : en fait, la station émettrice contrôle en permanence le niveau électrique sur le réseau. Si le niveau électrique observé ne correspond pas au niveau électrique envoyé, la station interprète cela comme une perte de l'arbitrage et cesse immédiatement la transmission (cf. paragraphe II.4.4.2).
- Le contrôle de la structure de trame est effectué au transmetteur qui examine la structure de la trame en vérifiant le positionnement des champs "fin de données" et "fin de la trame de réponse" ainsi que la longueur maximale des trames concaténées.

II.4.5 Le TTP (Time Triggered Protocol)

Le TTP est un protocole qui, à l'origine, a été développé à l'université de Vienne en Autriche dans le cadre du réseau informatique MARS [Kopetz *et al.* 1989]. Ce réseau est un système réparti tolérant aux fautes dont le fonctionnement est gouverné par le temps. En respectant la terminologie employée dans le cadre du TTP, les stations connectées au réseau sont par la suite appelées *FTU* ("Failure Tolerant Unit"), chaque FTU pouvant être composé de plusieurs *nœud*s redondants.

II.4.5.1 Principe de fonctionnement

Le protocole TTP est conçu selon l'approche TDMA ("Time Division Multiple Access"). Ici, toutes les communications sont prévues lors de la phase de conception, les FTU connectés au réseau disposent d'une fenêtre temporelle fixe pour envoyer leurs trames. L'envoi des trames est déclenché par une horloge globale tolérante aux fautes qui est réalisée en synchronisant les horloges locales de chaque FTU du réseau [Kopetz & Ochsenreiter 1987, Kopetz & Schwabl 1989]. En l'absence de fautes, le comportement d'un tel réseau est entièrement déterministe. De plus, le protocole TTP est un support pour tolérer des fautes au niveau de chaque FTU et du canal de transmission. Chaque FTU a des mécanismes de détection d'erreurs internes : en cas d'autodétection d'erreur, le FTU se rend passif. Les messages qui sont détectés comme erronés au FTU récepteur sont ignorés. On suppose alors que tous les FTU ont un comportement de silence sur défaillance et que les défaillances de communication sont des défaillances par omission.

Quatre configurations du réseau TTP sont possibles [Kopetz & Grünsteidl 1993] :

- Dans la configuration 1, il y a deux canaux de communication en redondance, chaque nœud (dans ce cas, c'est équivalent à un FTU) envoie ses trames une fois sur chaque bus. On peut ainsi tolérer une faute de communication permanente ou transitoire.
- Dans la configuration 2, chaque nœud est dupliqué formant ainsi une véritable unité tolérante aux fautes, appelée FTU. Le protocole TTP garantit que les nœuds répliqués d'un FTU font les changements d'état en même temps. Le FTU est considéré en fonctionnement correct lorsqu'il y a au moins un de ses nœuds qui est opérationnel. Chacun des deux nœuds du FTU est connecté aux deux canaux redondants, sur un desquels il envoie ses trames une fois. Cela signifie que chaque FTU envoie ses trames deux fois. On tolère ainsi une faute permanente ou transitoire d'un nœud ainsi qu'une faute transitoire et une faute permanente de communication.
- La configuration 3 est d'un point de vue structurel équivalente à la configuration 2. La différence est que chacun des deux nœuds du FTU envoie ses trames une fois sur chaque bus. Cela signifie que chaque FTU envoie ses trames quatre fois. On tolère ainsi une faute permanente ou transitoire d'un nœud ainsi que trois fautes transitoires et une faute permanente de communication.
- La configuration 4 est une extension de la configuration 3, dans la mesure où l'on ajoute un troisième nœud au sein de chaque FTU. En fonctionnement normal, ce nœud supplémentaire ne participe pas activement à la communication, il observe simplement les messages échangés sur le réseau.

En cas de faute permanente de l'un des deux nœuds actifs du FTU, le troisième nœud est activé afin de remplacer le nœud défaillant. Il s'agit alors d'un nœud de secours qui, en respectant la terminologie du TTP, est appelé "shadow node".

Dans l'industrie automobile, les configurations 1 à 3 sont facilement imaginables, tandis qu'il est très peu probable de rencontrer la configuration 4. En effet, les fortes contraintes de coût dans le domaine de l'automobile empêcheraient certainement l'implantation de ressources matérielles non exploitées entièrement. C'est pour cela que sur la *Figure 2-2* seulement les trois premières configurations sont illustrées.

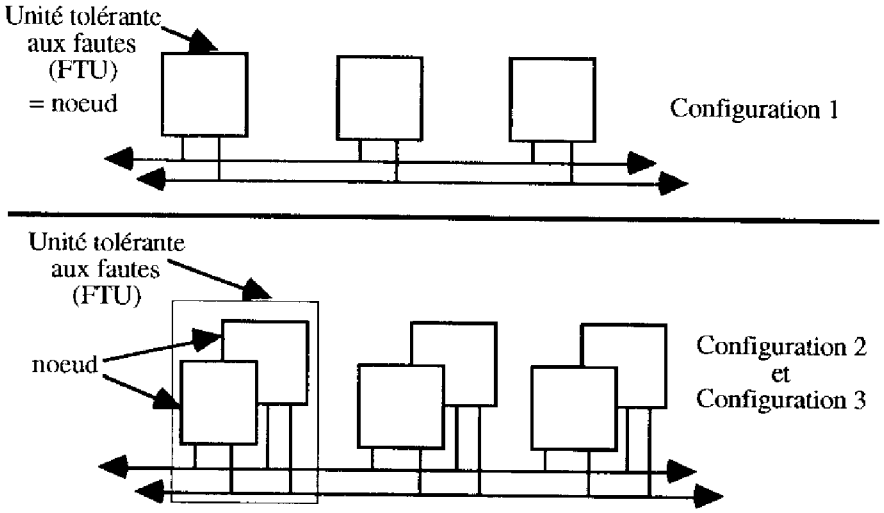


Figure 2-2: Trois configurations possibles du réseau TTP

La connexion d'un FTU optionnel au réseau peut être traité de deux façons différentes :

a) Lorsqu'on ajoute le FTU au réseau, on revoit toutes les communications et revérifie les temps de latence exigés.

b) Lors de la conception initiale, on peut laisser quelques fenêtres temporelles libres dans lesquelles des éventuelles trames supplémentaires peuvent être envoyées sans modifier les temps de latence des autres trames déjà existantes.

II.4.5.2 Le cycle TDMA

Pour rendre le protocole TTP encore plus flexible, on peut définir lors de la conception jusqu'à huit modes différents (c'est-à-dire, des cycles TDMA), entre lesquels on peut commuter dynamiquement pendant la phase opérationnelle. Ces modes correspondent à différentes phases d'utilisation ; par exemple, dans le cas de

l'automobile, on pourrait définir un mode pour la phase de démarrage, un mode pour la phase normale et un mode de secours. La commutation entre les différents modes est étudiée dans [Fohler 1993]. Un mode (ou cycle TDMA) est composé de plusieurs trames qui peuvent être séparées en deux groupes, à savoir la trame d'initialisation et la trame de données.

1) **La trame d'initialisation** est composée des champs suivants :

- L'en-tête (8 bits) est composée de plusieurs champs. Le premier bit permet de distinguer entre trame d'initialisation et trame de données. Les trois bits suivants indiquent un éventuel changement de mode, permettant ainsi de réagir aux événements asynchrones. Les quatre derniers bits sont destinés à l'acquittement de la trame précédente ; chacun des 4 bits permet d'acquitter une trame, c'est-à-dire que le FTU récepteur peut acquitter jusqu'à quatre trames. Chaque FTU qui est connecté au réseau et qui fonctionne correctement, "écoute" la trame présente sur le réseau. Dès qu'un FTU récepteur a acquitté la trame, les autres FTU recevant l'acquittement concluent que la trame a été envoyée correctement et que tous les FTU destinataires l'ont reçue.
- L'état du contrôleur (5 octets) décrit l'état actuel du système global : il indique les FTU actifs et inactifs, l'horloge globale et le cycle TDMA. Par conséquent, l'état du contrôleur est composé de plusieurs champs. Les premiers 8 bits forment le *vecteur d'appartenance*. Ce vecteur indique pour chaque FTU (ici, au maximum 8 FTU) s'il est actif (bit = 1) ou s'il est inactif (bit = 0). Un FTU qui ne reçoit aucune trame suppose que c'est le FTU émetteur qui défaille et met le bit correspondant du vecteur d'appartenance à 0 lors du prochain cycle TDMA. S'il y a d'autres FTU qui reçoivent cette trame, ils vont mettre ce bit à 1. Lors d'un tel conflit, la majorité gagne et le FTU réellement défaillant est éliminé (on parle aussi de *consensus sur l'appartenance*). Le vecteur d'appartenance est suivi de 16 bits contenant la valeur de l'horloge globale. Enfin, les derniers 16 bits de mode indiquent dans quel cycle TDMA se trouve le système global.
- Le CRC (16 bits) porte sur la concaténation de l'en-tête et de l'état du contrôleur. Lors d'un désaccord entre le CRC reçu et le CRC recalculé par le récepteur, le FTU récepteur ignore la trame et attend le prochain cycle TDMA pour rafraîchir les données correspondantes. Cela assure que la trame normale est acceptée uniquement lorsque les FTU émetteur et récepteur ont la même vue de l'état du contrôleur.
- L'espace inter-trame (4 bits).

2) **La trame de données** est composée des champs suivants :

- L'en-tête (8 bits) a la même fonction que pour la trame d'initialisation.
- Les données (1-16 octets).
- Le CRC (16 bits) a la même fonction que pour la trame d'initialisation. Le calcul du CRC porte sur la concaténation de l'état du contrôleur (connu par la trame d'initialisation), de l'en-tête et des données.
- L'espace inter-trame (4 bits).

En général, un cycle TDMA est composé de plusieurs trames de données et d'une trame d'initialisation. Par exemple, dans l'application du protocole TTP à une configuration étalon définie par la SAE [Kopetz 1994, SAE_J2056/1 1994], il y a six FTU qui envoient 15 trames de données différentes ; deux FTU envoient en plus une trame d'initialisation.

II.4.6 Discussion

Lorsque l'on analyse les différents réseaux multiplexés présentés dans les trois paragraphes précédents, on doit considérer notamment les différences concernant leur sûreté de fonctionnement.

Un premier point à souligner est le temps de latence des messages. De par leur accès asynchrone au réseau avec résolution de conflits par arbitrage sur le bus, les protocoles CAN et J1850 ne peuvent garantir un temps de latence maximal que pour le message le plus prioritaire, cela sous l'hypothèse que le bus ne soit pas occupé. Ici, le protocole TTP se distingue par rapport à ces deux protocoles par la communication déterministe qui permet de garantir des temps de latence constants.

Un deuxième point important concerne les mécanismes de tolérance aux fautes. Dans le cas du CAN et du J1850, les fautes permanentes de communication peuvent être tolérées par la redondance physique du canal de transmission. En ce qui concerne les fautes transitoires sur le réseau ou dans les stations, leur tolérance peut être assurée par la réémission de messages. Pour le CAN, cette réémission est déclenchée dans la plupart des cas par une trame d'erreur qui indique aux stations du réseau que la précédente trame était erronée. L'absence d'acquittement peut également causer une nouvelle émission de la trame. Pour le J1850, la réémission d'une trame erronée se fait suite à l'abandon de la trame de données précédente. De manière analogue au CAN, l'absence d'un acquittement positif peut également induire la réémission de la trame précédente. La tolérance aux fautes permanentes dans les stations n'est pas considérée par les protocoles CAN et J1850, on suppose simplement que les stations ont un comportement de silence sur défaillance en cas de faute permanente. Le protocole TTP se distingue à ce point par rapport aux protocoles CAN et J1850 par ses différentes possibilités de configuration qui permettent de gérer une tolérance aux fautes locale à chaque station du réseau, même en cas de fautes permanentes des stations.

Malgré l'actuelle dominance du protocole CAN dans le domaine de l'automobile, le TTP semble intéresser certains constructeurs. Son implantation sur véhicule devrait être réalisée afin de pouvoir juger non seulement sur ses performances techniques, mais aussi sur l'aspect coût de production. Son surcoût éventuel par rapport à d'autres protocoles devra être pesé contre ses avantages au niveau de la sûreté de fonctionnement.

II.5 Spectre d'architectures

Plusieurs architectures sont imaginables pour répondre aux objectifs de fonctionnalité et de sûreté de fonctionnement. Pour faire face à la diversité des architectures, nous proposons de classer ces différentes architectures en tenant compte notamment de l'importance des réseaux multiplexés. Cela nous permet d'établir une base de comparaison bien structurée et de distinguer les architectures clairement. Nous allons développer un spectre d'architectures basé sur cinq points de vue ou *attributs*, chaque attribut étant caractérisé par deux valeurs représentant les positions opposées du point de vue.

II.5.1 Attributs de définition des architectures

Nous expliquons tout d'abord pour chacun des cinq attributs les deux valeurs extrêmes qu'il peut prendre.

1) L'allocation des processeurs :
   Le premier attribut distingue entre les processeurs dédiés et les processeurs banalisés.
   Dédier signifie dans notre contexte que chaque processeur (ou sous-ensemble de processeurs) est alloué à une fonction spécifique. Cela veut dire en particulier que *la gestion de la redondance ne peut être que locale*, c'est-à-dire particulière au processeur (ou sous-ensemble de processeurs).
   Banaliser est le fait de supprimer toutes les signes distinctifs. Concernant les processeurs, cela veut dire que chaque processeur peut être alloué à n'importe quelle fonction. La banalisation des processeurs rend possible *la gestion globale de la redondance*, c'est-à-dire une gestion de redondance qui s'applique à l'ensemble des processeurs.

2) L'accès aux entrées-sorties :
   Cet attribut distingue entre l'accès spécifique et l'accès partagé aux entrées-sorties.
   Un accès *spécifique* suppose que chaque processeur a sa propre liaison avec les capteurs et actionneurs, tandis qu'un accès *partagé* signifie que la communication entre le processeur et ses capteurs-actionneurs est réalisée par

l'intermédiaire d'un médium commun. Ce médium commun est en général un bus multiplexé (voir paragraphe II.4).

3) Le type d'allocation de redondance :
Le troisième attribut différencie entre la redondance hétérogène et la redondance homogène.
Les fonctions ont une redondance *hétérogène* lorsque chaque fonction a un degré de redondance qui lui est propre, tandis que l'épithète *homogène* exprime le fait que le degré de redondance est le même pour toutes les fonctions.

4) L'organisation fonctionnelle :
Nous distinguons ici entre la fédération fonctionnelle et l'intégration fonctionnelle.
A l'origine, le mot *fédération* est synonyme d'une association de plusieurs sociétés, états, syndicats, groupés sous une autorité commune (état fédéral, fédération sportive, ...). Dans le contexte de l'informatique automobile, nous proposons d'employer ce mot pour décrire un ensemble de fonctions qui n'ont pas de lien explicite entre elles et qui sont donc indépendantes. Le mot fédération est justifié par le fait que toutes ces fonctions sont conçues pour le même système — l'automobile — et qu'elles sont alors couplées mécaniquement ainsi que par l'interaction homme-machine.
Le mot *intégration* peut être défini d'une manière générale par la coordination des activités de plusieurs organes, nécessaires à un fonctionnement harmonieux. Dans notre cas, ce mot est utilisé pour parler des fonctions qui interagissent entre elles de façon à atteindre des objectifs partagés.

5) Le placement des processeurs :
Le dernier attribut distingue entre la distribution et la centralisation *géographique* des processeurs.
Dans le cas de la *distribution*, les processeurs sont répartis dans plusieurs endroits du véhicule. Dans le cas de la *centralisation*, les processeurs sont concentrés en un endroit unique dans le véhicule.

Sachant que chacun des attributs peut prendre deux valeurs opposées, on aboutit en théorie à 32 architectures possibles (*Tableau 2-1*). Nous identifions chacune de ces architectures par la nomenclature indiquant les valeurs de chaque attribut. Une combinaison d'architectures peut être représentée avec des astérisques pour indiquer les attributs indifférents.

Parmi les 32 architectures, on peut en éliminer quelques-unes qui comprennent des combinaisons soit impossibles soit difficilement imaginables (sur le *Tableau 2-1*, les architectures non considérées sont grisées) :

Tableau 2-1 : Classification des architectures

	Allocation des processeurs De : Dédiés Ba : Banalisés	Accès aux entrées-sorties Sp : Spécifique Pa : Partagé	Allocation des redondances He : Hétérogène Ho : Homogène	Organisation fonctionnelle Fe : Fédération In : Intégration	Placement des processeurs Di : Distribué Ce : Centralisé	Figure
1	De	Sp	He	Fe	Di	2-3
2	De	Sp	He	Fe	Ce	2-3
3	De	Sp	He	In	Di	2-4
4	De	Sp	He	In	Ce	2-4
5	De	Sp	Ho	Fe	Di	
6	De	Sp	Ho	Fe	Ce	
7	De	Sp	Ho	In	Di	
8	De	Sp	Ho	In	Ce	
9	De	Pa	He	Fe	Di	2-5
10	De	Pa	He	Fe	Ce	2-5
11	De	Pa	He	In	Di	2-5
12	De	Pa	He	In	Ce	2-5
13	De	Pa	Ho	Fe	Di	
14	De	Pa	Ho	Fe	Ce	
15	De	Pa	Ho	In	Di	
16	De	Pa	Ho	In	Ce	
17	Ba	Sp	He	Fe	Di	
18	Ba	Sp	He	Fe	Ce	
19	Ba	Sp	He	In	Di	
20	Ba	Sp	He	In	Ce	
21	Ba	Sp	Ho	Fe	Di	
22	Ba	Sp	Ho	Fe	Ce	
23	Ba	Sp	Ho	In	Di	
24	Ba	Sp	Ho	In	Ce	
25	Ba	Pa	He	Fe	Di	2-6
26	Ba	Pa	He	Fe	Ce	2-6
27	Ba	Pa	He	In	Di	2-6
28	Ba	Pa	He	In	Ce	2-6
29	Ba	Pa	Ho	Fe	Di	2-7
30	Ba	Pa	Ho	Fe	Ce	2-7
31	Ba	Pa	Ho	In	Di	2-7
32	Ba	Pa	Ho	In	Ce	2-7

a) La présence de processeurs dédiés signifie non seulement une allocation spécifique à la fonction correspondante, mais également une gestion locale de la

redondance. Une allocation homogène de la redondance ne revêt aucun intérêt particulier dans un tel cas et nous éliminons ainsi les huit architectures De*Ho**.

b) Lorsque les processeurs sont banalisés, on dispose d'une certaine flexibilité concernant leur allocation aux différentes tâches. Cette flexibilité se traduit également au niveau de l'accès aux capteurs et actionneurs. Il est difficilement imaginable alors d'avoir un accès spécifique aux entrées-sorties. Nous éliminons donc les huit architectures BaSp***.

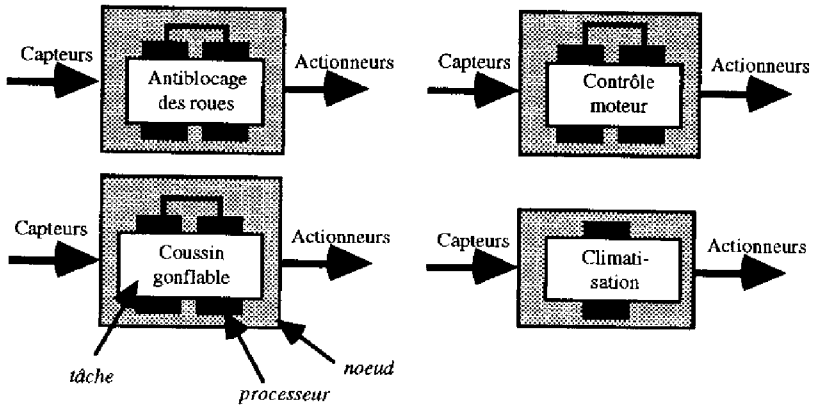
Après l'élimination des architectures non considérées, on aboutit à 16 architectures qui représentent un spectre allant de la conception entièrement fédérée à la conception entièrement intégrée. Pendant l'élaboration du spectre d'architectures, nous avons fait abstraction du matériel utilisé. En effet, les processeurs peuvent par exemple être de simples microcontrôleurs provenant de différents fabricants.

Le paragraphe suivant est dédié à la comparaison qualitative des architectures du spectre, montrant les différentes étapes vers l'intégration des systèmes. Les aspects d'intégration sont également discutés dans [Nakano 1988, Schilke *et al.* 1988, Watson 1988, Heinrich 1989, Panizza 1989].

II.5.2 Comparaison qualitative des architectures

Dans ce paragraphe, nous présentons et comparons de façon qualitative chacune des 16 architectures retenues en suivant l'ordre du *Tableau 2-1*.

A une extrémité du spectre se trouvent les architectures DeSpHeFe* (*Figure 2-3*). Chaque fonction est réalisée par un processeur dédié, les techniques de sûreté de fonctionnement sont appliquées localement et les degrés de redondance sont par définition hétérogènes. Lors d'une défaillance d'un système embarqué, la détection d'erreur et le traitement de faute sont pris en compte au niveau de la fonction. En général, la voie électronique est secourue par une voie mécanique (exemple : l'accélérateur électronique). Souvent, le calculateur est physiquement intégré au système mécanique et hydraulique afin d'éviter des interférences électromagnétiques sur les connexions et de réduire le coût. L'accès aux capteurs et actionneurs correspondants est spécifique. Le placement des processeurs est en général distribué (architecture DeSpHeFeDi) dû à la distribution géographique des actionneurs et capteurs, à la place disponible et à d'autres contraintes d'environnement. Mais une centralisation (architecture DeSpHeFeCe) pourrait être envisagée pour faciliter la maintenance. La caractéristique principale de ces architectures est alors la conception indépendante de chaque module. Au niveau global, on a souvent une complexité supérieure à celle qui serait nécessaire en termes de ressources matérielles. De plus, les schémas de gestion des redondances sont relativement rigides. Cela conduit en général à un coût global qui n'est pas optimisé.

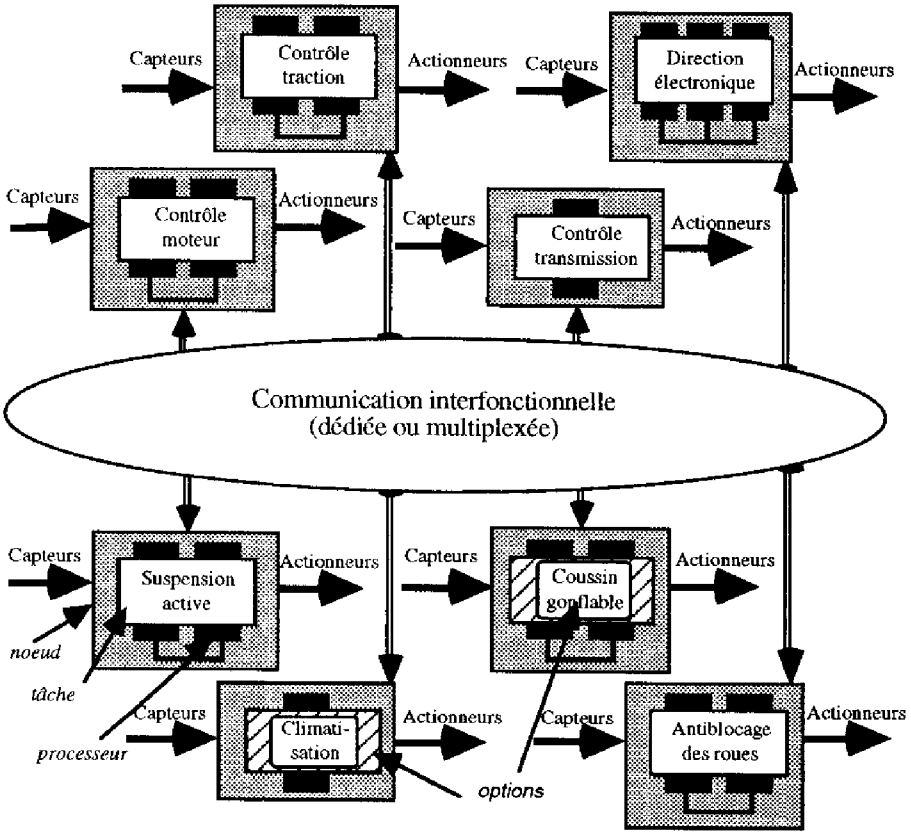


De	Processeurs dédiés	
Sp	Accès spécifique aux entrées-sorties	
He	Redondance hétérogène	
Fe	Fédération fonctionnelle	
Di/Ce	Distribution des processeurs	Centralisation des processeurs

*Figure 2-3 : Architectures DeSpHeFe**

A l'heure actuelle, les différentes fonctions automobiles ne sont plus totalement indépendantes les unes des autres. On voit apparaître des liens entre les fonctions pour permettre une amélioration du fonctionnement dans son ensemble (architectures DeSpHeIn*, *Figure 2-4*). Cette intégration fonctionnelle est réalisée en général par des fils point-à-point qui permettent la communication entre les modules [Lorenz *et al.* 1988]. A la place de ces connexions point-à-point, on essaye d'introduire des réseaux multiplexés qui pourraient augmenter les capacités de communication et ainsi la performance globale du véhicule [Fujiki *et al.* 1991, SAE_J2056/2 1994]. Des options peuvent être ajoutées au réseau, mais la conception des échanges d'information sur le bus doit éventuellement être revue pour intégrer la nouvelle fonction rajoutée. Les interfaces multiplexées nécessitent une gestion de protocole relativement complexe ce qui signifie un coût plus élevé. Malheureusement, les prix élevés des interfaces multiplexées empêchent à l'heure actuelle l'introduction des réseaux multiplexés en grande série. Uniquement quelques véhicules haut de gamme sont déjà équipés avec des réseaux multiplexés, mais l'information qui circule sur le bus n'est pas critique vis-à-vis de la sécurité et n'a pour but que l'optimisation de la performance globale. Une défaillance au niveau de la communication (par exemple, la coupure des câbles) n'affecterait pas le fonctionnement essentiel des sous-systèmes locaux connectés. De plus, la capacité de communication du réseau multiplexé est très

faiblement exploitée (une charge d'environ 5%), afin d'éviter le conflit entre les messages et de diminuer tout risque de blocage ou de saturation dû à l'envoi d'éventuels messages d'erreur. Notons également que le nombre de sous-systèmes connectés au bus est relativement petit, et que la longueur du réseau ne dépasse pas généralement quelques mètres.

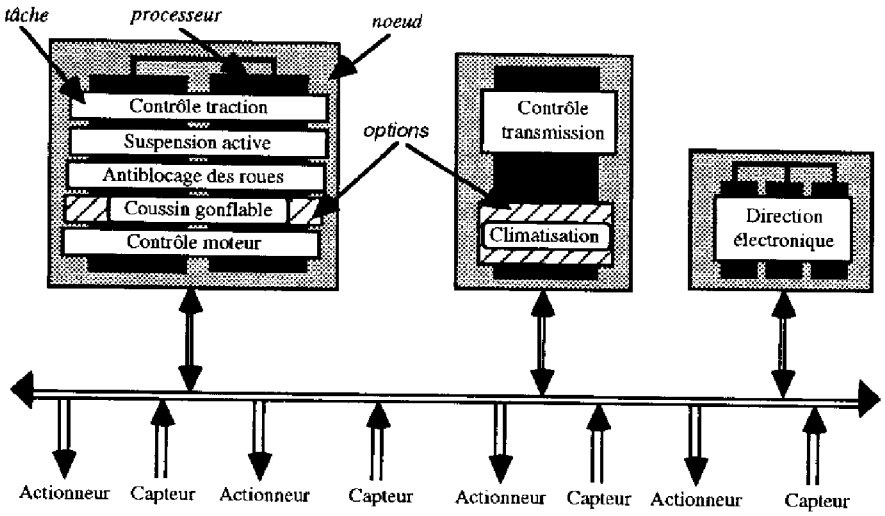


De	Processeurs dédiés	
Sp	Accès spécifique aux entrées-sorties	
He	Redondance hétérogène	
In	Intégration fonctionnelle	
Di/Ce	Distribution des processeurs	Centralisation des processeurs

*Figure 2-4 : Architectures DeSpHeIn**

L'architecture de la *Figure 2-4* illustre une application possible du protocole TTP qui autoriserait l'échange d'informations critiques vis-à-vis de la sécurité et du temps, et qui inclurait également la tolérance aux fautes au niveau de chaque station du réseau ainsi que de la communication.

La présence d'un réseau multiplexé permet non seulement la communication inter-système, mais aussi la connexion de tous les capteurs et actionneurs, soit en les regroupant sur des concentrateurs, soit en les équipant individuellement d'une interface multiplexée (architectures DePaHe**, *Figure 2-5*).



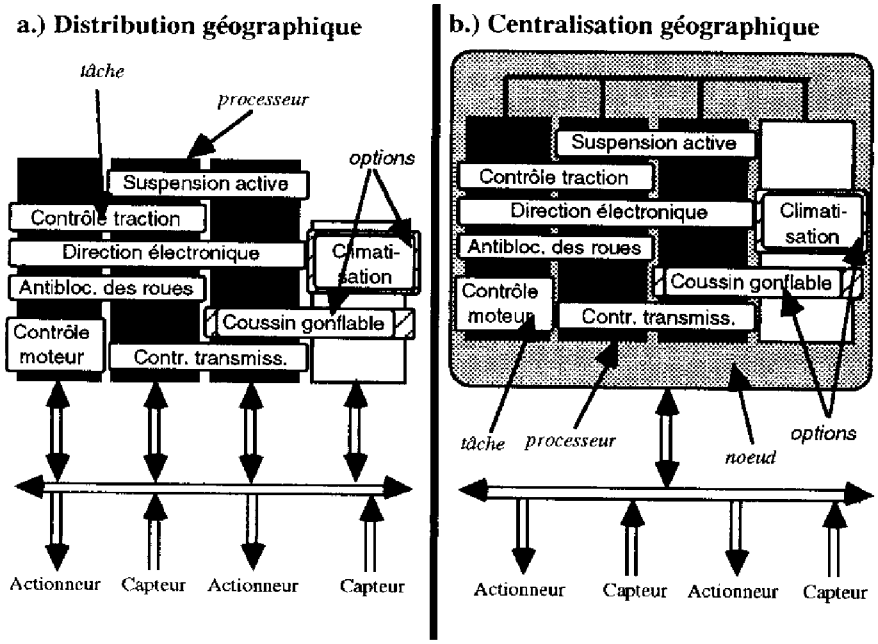
De	Processeurs dédiés	
Pa	Accès partagé aux entrées-sorties	
He	Redondance hétérogène	
Fe/In	Fédération fonctionnelle	Intégration fonctionnelle
Di/Ce	Distribution des processeurs	Centralisation des processeurs

*Figure 2-5 : Architectures DePaHe***

Cela aurait l'avantage d'un partage d'informations facilité, permettant en outre, dans le cadre d'une démarche globale à la sûreté de fonctionnement, de regrouper plusieurs fonctions sur un même module tolérant aux fautes, comme cela est illustré sur la figure. Le regroupement des fonctions suivant une criticité semblable a pour but de faire passer les interactions entre niveaux de criticité différents sur le réseau multiplexé qui représente alors une barrière matérielle de confinement d'erreur. Par contre, les barrières de confinement d'erreur entre niveaux de criticité semblables

doivent être implantées localement par des mécanismes matériels et logiciels. C'est pour cela que nous avons considéré le cas (extrême) de la fédération fonctionnelle qui élimine l'interaction entre les fonctions. Le regroupement des fonctions suivant leur criticité devrait également permettre de réduire les coûts. L'ensemble des fonctions ainsi regroupées peut être vu comme une macro-fonction, c'est pour cela que ce n'est pas un critère de distinction dans le spectre.

Une plus grande flexibilité de configuration est donnée lorsque l'on peut allouer les processeurs à n'importe quelle fonction, indépendamment de sa criticité, ce qui veut dire que l'on banalise les processeurs (architectures BaPa***, *Figure 2-6* et *Figure 2-7*).



Ba	Processeurs banalisés	
Pa	Accès partagé aux entrées-sorties	
He	Redondance hétérogène	
Fe/In	Fédération fonctionnelle	Intégration fonctionnelle
Di/Ce	Distribution des processeurs	Centralisation des processeurs

*Figure 2-6 : Architectures BaPaHe***

Ici, un système global est spécifié et conçu qui réalise toutes les fonctions, tenant compte des interactions entre les différentes fonctions, du partage des informations

capteurs et de l'accès partagé aux actionneurs. Les architectures BaPa*** sont conçues pour réduire le nombre de processeurs, donc diminuer les coûts, le poids et le volume. La démarche de sûreté de fonctionnement est entièrement globale, dans le but d'améliorer et d'harmoniser le comportement du système en cas de défaillance. On pourrait très bien imaginer par exemple de pouvoir reconfigurer la répartition fonctionnelle du système pendant sa vie opérationnelle après l'occurrence d'une défaillance. Dans ce contexte, la coexistence de modules logiciels avec différentes criticités sur un même processeur ainsi que leurs interactions doivent être prises en compte. Un problème est de dimensionner les capacités de traitement et de mémoire de telle façon que les fonctions optionnelles peuvent être implantées sans difficultés majeures. En cas d'une forte interdépendance de la fonction optionnelle avec les autres fonctions, une partie de la gestion globale de tolérance aux fautes doit être revue. Les architectures BaPaHe** (*Figure 2-6*) illustrent la banalisation des processeurs, le degré de redondance étant spécifique pour chaque tâche. Dans l'exemple de la *Figure 2-6*, les fonctions optionnelles sont implantées en partie sur un quatrième processeur supplémentaire. Mais il est également possible de prévoir suffisamment de mémoire et de capacité de calcul sur les autres processeurs afin d'y pouvoir accueillir entièrement les fonctions optionnelles.

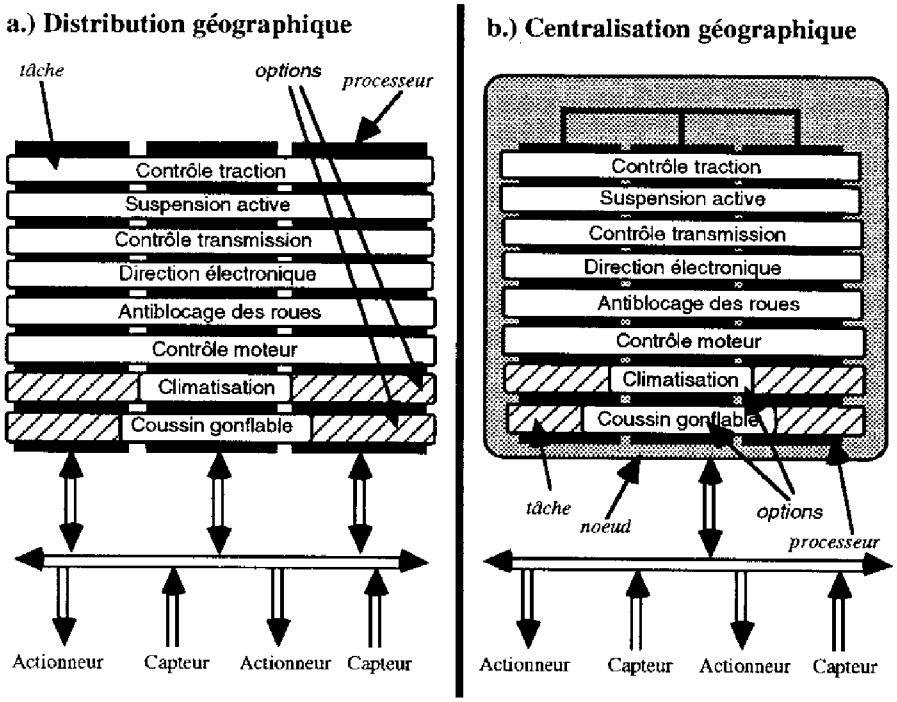
Dans le cas des architectures BaPaHo** (*Figure 2-7*), l'allocation homogène des redondances crée un surcoût de redondance pour certaines fonctions, mais la gestion de la redondance est alors plus facile que dans le cas de la redondance hétérogène. La capacité de calcul et de mémoire ne peut être accrue qu'en augmentant la capacité de chaque canal redondant. Cela signifie que, dès la conception initiale du système, l'accueil d'éventuelles fonctions optionnelles doit être pris en compte.

Les architectures BaPa*** permettent chacune de distribuer ou de centraliser les processeurs.

En cas de distribution géographique, les processeurs sont individuellement connectés au réseau. Le problème principal qui se pose avec une telle distribution géographique est dû aux interférences pouvant affecter les transmissions. Cela conduit à limiter de façon importante la bande passante du système de communication entre processeurs (qui serait alors généralement mis en œuvre par un bus série). Cette limitation de la bande passante entre processeurs a pour conséquence que les messages non-fonctionnels liés à la gestion de la redondance pourraient prendre une importance rédhibitoire.

En cas de centralisation géographique, le logiciel se trouve sur trois ou quatre processeurs regroupés dans un boîtier. Le câblage est réduit aux connexions entre le calculateur central tolérant aux fautes, les capteurs et les actionneurs. Les interférences électromagnétiques affectent la communication inter-processeurs moins que dans le cas de la distribution géographique. Le regroupement des processeurs

dans un boîtier permet alors d'installer un bus parallèle et d'augmenter ainsi le débit de communication inter-processeurs. De plus, une horloge commune tolérante aux fautes serait relativement facile à réaliser et des mécanismes redondants microsynchronisés pourraient être implantés. La gestion globale de la redondance a donc une influence mineure sur la performance globale.



Ba	Processeurs banalisés	
Pa	Accès partagé aux entrées-sorties	
Ho	Redondance homogène	
Fe/In	Fédération fonctionnelle	Intégration fonctionnelle
Di/Ce	Distribution des processeurs	Centralisation des processeurs

*Figure 2-7 : Architectures BaPaHo***

II.5.3 Analyse des coûts

Dans ce sous-paragraphe, nous allons élaborer un modèle qui permet de calculer approximativement le coût de chaque architecture du spectre. Le coût d'un système

informatique est déterminé par deux facteurs principaux, à savoir le coût de développement et le coût récurrent.

Le coût de développement (noté *Cd*) est essentiellement lié au temps ainsi qu'au nombre de personnes nécessaires pour le développement. Nous faisons ici abstraction des frais généraux, tels que la location d'un bâtiment ou le support d'outils de développement spécifiques. En général, on exprime le coût de développement par des *hommes-années*, le coût d'un homme-année étant connu. Dans notre modèle, nous rapportons le coût de développement au nombre de pièces produites (noté *N*) afin de pouvoir juger de son influence sur le coût total de fabrication, ce coût étant appelé coût de développement rapporté (noté *Cdr*).

Le coût récurrent (noté *Cr*) est directement lié à la complexité matérielle et logicielle du système. Afin de calculer le coût récurrent, nous divisons le système en plusieurs sous-systèmes :

1) Le coût du processeur ou du sous-ensemble de processeurs redondants (noté *C1*) est exprimé en fonction de deux paramètres, à savoir le degré de redondance et la complexité intrinsèque du processeur. Le degré de redondance est introduit comme simple facteur multiplicatif, et en ce qui concerne la complexité, nous posons l'hypothèse simplificatrice que le coût est une fonction linéaire de la vitesse de calcul.

2) Le coût de la mémoire (noté *C2*) est également exprimé en fonction de deux paramètres, à savoir le degré de redondance et la taille mémoire (nous faisons l'hypothèse que le coût est une fonction linéaire de la taille mémoire).

3) Le coût de l'interface (noté *C3*) : ici, nous distinguons entre une interface ordinaire et une interface multiplexée. Cela permet d'étudier l'influence du multiplexage sur le coût total du système. Les coûts du câble et du connecteur associés sont pris en compte dans le coût de l'interface.

4) Le coût du boîtier (noté *C4*) étant supposé indépendant du système, il n'influe sur le coût total que par le nombre de boîtiers nécessaires.

Le *Tableau 2-2* illustre, à titre d'exemple, le calcul du coût pour le système de contrôle-moteur de l'architecture DeSpHeInDi, l'interconnexion avec les autres systèmes étant réalisée par des liaisons point-à-point. En se basant sur un système réel, nous avons estimé les différents paramètres, tels que le nombre d'interfaces ou la capacité mémoire. Les paramètres indépendants du système sont indiqués dans la légende du tableau, tandis que les paramètres spécifiques au système se trouvent dans la partie A du tableau. L'ensemble des paramètres permet de calculer le coût total *Ct* qui est la somme du coût de développement rapporté *Cdr* et du coût récurrent (*Cr=C1+C2+C3+C4*) (voir parties B et C du tableau). Les coûts sont exprimés dans une monnaie fictive que nous appelons *UP* (Unité de Prix). Lors du choix numérique

des paramètres de coût, nous nous sommes basés sur des prix de catalogue. Il est évident que ces prix sont approximatifs, en réalité ils dépendent entre autres du choix du fabricant et de la technologie interne du composant. De plus, lors d'une réalisation d'un système en grande série, les prix des composants sont généralement renégociés. Le choix de paramètres approximatifs est justifié par le fait que nous nous intéressons non pas au coût absolu d'une architecture, mais à la comparaison de différents choix d'architectures. Ayant posé les mêmes hypothèses simplificatrices pour chaque architecture, nous pouvons dégager quelques conclusions générales quant aux coûts relatifs des architectures.

Tableau 2-2 : Coût du système de contrôle-moteur

prix de base du processeur : *proc_base* = 120 *UP* ;
prix par unité de vitesse : *proc_fact* = 1,5 *UP/Mips* ;
prix de base de la mémoire : *mem_base* = 40 *UP* ;
prix par unité de taille : *mem_fact* = 0,2 *UP/Koctets* ;
prix de l'interface ordinaire : *int* = 1,5 *UP* ;
prix de l'interface multiplexée : *int_mux* = 40 *UP*

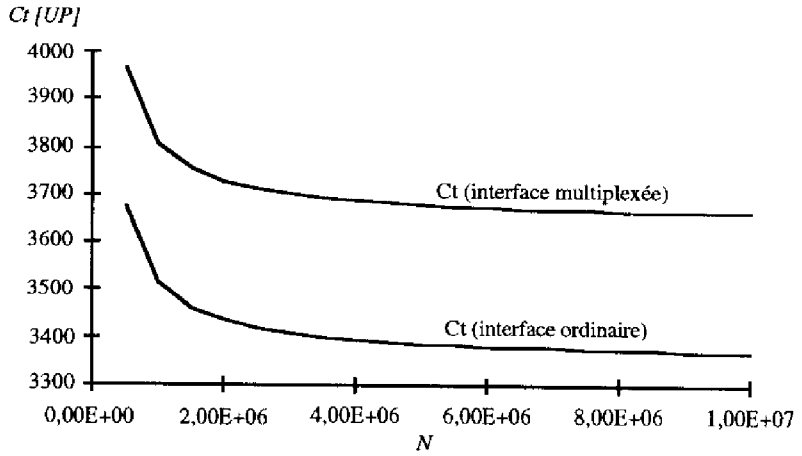
	Cd = Coût de développement [*UP*]	2,00E+07
	N = Nombre de pièces produites	1,00E+06
	Mem = Capacité mémoire, sans redondance [*Koctets*]	50
A	*Red* = Degré de redondance	2
	Vit = Vitesse processeur [*Mips*]	20
	N_int = Nombre d'interfaces	43
	N_int_mux = Nombre d'interfaces multiplexées	0
	Cdr = *Cd* / *N* [*UP*]	20
	C1 = Coût Processeur = (*proc_base* + *proc_fact***Vit*)**Red* [*UP*]	300
B	*C2* = Coût Mémoire = (*mem_base* + *mem_fact***Mem*)**Red* [*UP*]	100
	C3 = Coût Interface = *int***N_int* + *int_mux***N_int_mux* [*UP*]	64,5
	C4 = Coût Boîtier [*UP*]	50
C	*Cr* = Coût récurrent = *C1* + *C2*+ *C3* + *C4* [*UP*]	514,5
	Ct = Coût total = *Cdr* + *C1* + *C2* + *C3* + *C4* [*UP*]	534,5

A partir du modèle de coût développé ici, chaque architecture du spectre peut être analysée. Dans ce sous-paragraphe, nous nous limitons aux architectures DeSpHeInDi et BaPaHoInCe qui se trouvent aux deux extrémités du spectre.

II.5.3.1 Coût de l'architecture DeSpHeInDi

Pour le calcul du coût total de l'architecture DeSpHeInDi, il suffit d'établir un tableau — tel que le *Tableau 2-2* — pour chacune des 8 fonctions réalisées par cette architecture (cf. *Figure 2-4*) et de calculer la somme des différents coûts totaux. Nous prenons l'hypothèse simplificatrice que, parmi les paramètres spécifiques de chaque système, le coût de développement et le nombre de pièces produites sont les mêmes pour chaque fonction. Les valeurs numériques de autres paramètres spécifiques aux systèmes sont basées sur une analyse de systèmes réels.

A l'heure actuelle, les interconnexions entre les différents systèmes de l'architecture DeSpHeInDi peuvent être matérialisées par des liaisons point-à-point ou par un réseau multiplexé. Ces deux réalisations se distinguent non seulement d'un point de vue technique, mais également au niveau du coût, le réseau multiplexé induisant actuellement un surcoût. C'est pour cela que nous calculons le coût total de l'architecture DeSpHeInDi en distinguant ces deux principes de communication. Les systèmes embarqués sur l'automobile étant produits en grande série, on s'intéresse de plus à l'influence du nombre de pièces produites N sur le coût total Ct (voir *Figure 2-8*). Nous vérifions que le multiplexage crée un surcoût qui, dans notre exemple de calcul, est d'environ 10%. Nous vérifions également que le coût total Ct est inversement proportionnel au nombre de pièces produites, la valeur asymptotique étant le coût récurrent Cr.



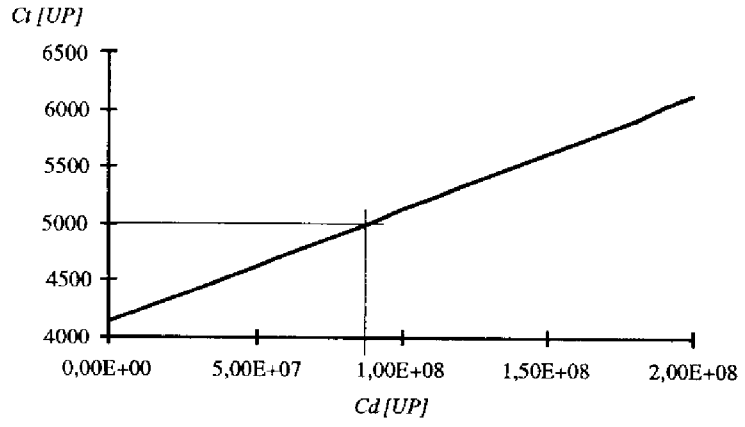
Ct = Coût total [*UP*] ; *N* = nombre de pièces produites

Figure 2-8 : Coût de l'architecture DeSpHeInDi

II.5.3.2 Coût de l'architecture BaPaHoInCe

Pour le calcul du coût total de l'architecture BaPaHoInCe (cf. *Figure 2-7*), il est important de noter que nous avons choisi de multiplexer chaque capteur et actionneur individuellement. Le coût de développement d'une telle architecture dans le domaine de l'automobile est relativement peu connu à l'heure actuelle. Par conséquent, nous étudions ici l'influence du coût de développement sur le coût total et nous prenons l'hypothèse que la valeur numérique du coût de développement se trouve entre deux bornes : la borne inférieure est le coût de développement d'un seul système relativement peu complexe tel que le contrôle-moteur ; la borne supérieure est le nombre de fonctions multiplié par le coût de développement d'un système. De manière analogue au paragraphe précédent, le coût total dépend également de N (nombre de pièces produites). Dans notre exemple de calcul, nous supposons que N est constant et relativement petit ($N = 10^5$), car une telle architecture serait seulement implantée sur les véhicules haut de gamme.

Concernant l'influence de Cd sur Ct, il pourrait être intéressant de se fixer une valeur maximale pour Ct et de conclure sur la valeur maximale de Cd. La *Figure 2-9* illustre l'évolution de Ct en fonction de Cd et elle nous permet de constater que l'on dépasse par exemple l'objectif $Ct = 5000$ UP pour $Cd > 9*10^7$ *UP*.



Cd = Coût de développement [*UP*] ; *Ct* = Coût total [*UP*] (*N* = 100 000)

Figure 2-9 : Coût de l'architecture BaPaHoInCe

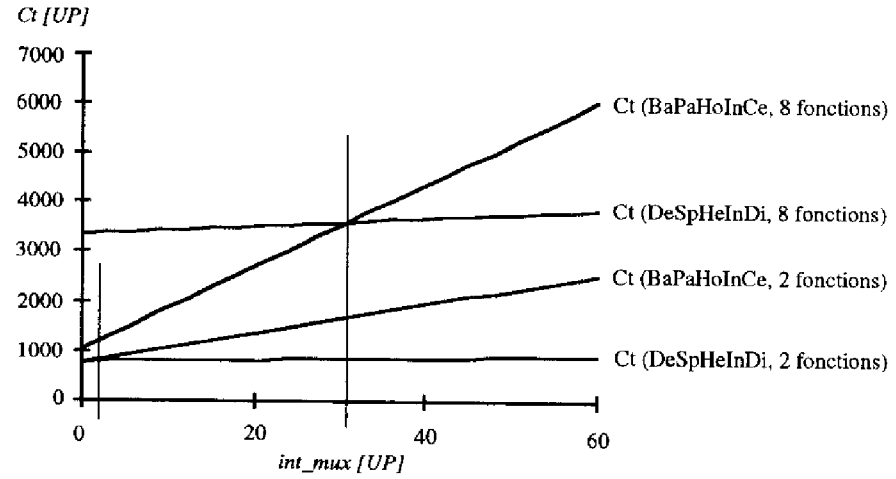
II.5.3.3 Comparaison de coût des architectures DeSpHeInDi et BaPaHoInCe

Dans ce paragraphe, nous allons comparer les coûts de l'architecture DeSpHeInDi (l'interconnexion des systèmes étant réalisée par un réseau multiplexé) et de l'architecture BaPaHoInCe. Le but de cette comparaison est de pouvoir donner des

recommandations concernant le choix d'une de ces architectures à chaque extrémité du spectre. Nous supposons que l'influence du prix de l'interface multiplexée *int_mux* est importante dans ce choix et nous calculons par conséquent le coût total *Ct* pour différentes valeurs de *int_mux*. Dans le cas de l'architecture DeSpHeInDi, le multiplexage se limite à la communication inter-système et nous nous attendons à une influence relativement faible de *int_mux*. Pour l'architecture BaPaHoInCe, l'influence de *int_mux* est supposée très grande, dû au grand nombre de capteurs et actionneurs multiplexés individuellement.

D'une façon qualitative, nous pouvons dire que la complexité fonctionnelle est directement liée au nombre d'interfaces nécessaires et influe alors sur le coût total *Ct*. C'est pour cela que la relation entre *int_mux* et *Ct* est étudiée non seulement pour le cas où toutes les huit fonctions sont implantées, mais également pour le cas où seulement deux fonctions parmi ces huit fonctions sont implantées.

La *Figure 2-10* regroupe l'évolution des coûts totaux des deux architectures en fonction de *int_mux*, le nombre de fonctions étant égal à 8 ou à 2 respectivement ($N = 10^7$). D'une manière générale, on voit que l'architecture BaPaHoInCe présente un avantage en coût par rapport à l'architecture DeSpHeInDi, lorsque le prix de l'interface multiplexée est inférieur à une certaine valeur. Nous vérifions que cette valeur dépend fortement du nombre de fonctions implantées sur l'architecture, elle augmente lorsque le nombre de fonctions augmente. Nous en concluons que l'architecture BaPaHoInCe est intéressante lorsque l'on veut y implanter un grand nombre de fonctions, car, dans ce cas, elle garde un avantage en coût jusqu'à un prix relativement élevé de l'interface multiplexée.



int_mux = prix d'une interface multiplexée [*UP*]; *Ct* = coût total [*UP*] ($N = 10^7$)

Figure 2-10 : Coût des architectures DeSpHeInDi et BaPaHoInCe

Conclusion

Dans ce chapitre, nous avons argumenté en faveur d'une définition globale des futures architectures afin de mieux remplir les objectifs de fonctionnalité, de sûreté de fonctionnement et de coût. Cela signifie, entre autres, une répartition flexible des fonctions sur les différents processeurs et une approche globale à la tolérance aux fautes. L'analyse de différentes architectures possibles montre que la conception entièrement intégrée des systèmes embarqués automobiles est souhaitable, mais difficile à atteindre. A l'heure actuelle, une première étape vers de telles architectures est commencée en interconnectant les différents calculateurs. Compte tenu de la criticité des tâches réalisées par l'ensemble des systèmes embarqués sur l'automobile, la sûreté de fonctionnement des voies de communication a une importance particulière. Lorsque la communication est réalisée par le biais d'un réseau multiplexé, deux objectifs de sûreté de fonctionnement peuvent être soulignés : d'une part, le temps de latence des messages doit être constant et prévisible ; d'autre part, le protocole implanté doit disposer de mécanismes de tolérance aux fautes. Nous avons montré que les protocoles existants sur le marché automobile ne correspondent qu'en partie à ces objectifs souhaités et qu'il est intéressant d'étudier de nouvelles approches telles que le protocole TTP.

La comparaison qualitative des architectures embarquées sur l'automobile qui a été présentée dans ce chapitre doit être approfondie en ce qui concerne l'aspect de sûreté de fonctionnement. Il serait souhaitable d'étudier l'ensemble des 16 architectures du spectre afin de mieux guider le choix d'architecture. Le nombre relativement important des différentes solutions d'architecture nous conduit à développer dans les chapitres III et IV une méthode permettant d'évaluer quantitativement la sûreté de fonctionnement de ces architectures d'une façon structurée et rigoureuse. L'application de cette méthode sera illustrée sur un sous-ensemble du spectre d'architectures.

Chapitre III

Choix d'une méthode d'évaluation de la sûreté de fonctionnement

L'évaluation de la sûreté de fonctionnement consiste à analyser les défaillances des composants et leurs conséquences sur la sûreté de fonctionnement du système [Laprie *et al.* 1996]. Dans le cadre de nos travaux, l'évaluation de la sûreté de fonctionnement est la partie principale de l'analyse et de la comparaison des architectures présentées dans le chapitre II. L'évaluation à la fois qualitative et quantitative permet de valider les objectifs que l'on se fixe lors de la conception d'un système et de choisir l'architecture "optimale" en vue de ces objectifs. Comme déjà mentionné au paragraphe I.4.1, on peut distinguer entre l'évaluation ordinale et l'évaluation probabiliste. Dans les deux premiers paragraphes de ce chapitre, nous présentons différentes techniques d'évaluation ordinale et probabiliste, en focalisant sur la technique d'évaluation probabiliste par modélisation markovienne. La méthode de modélisation que nous avons développée et suivie permet de générer des chaînes de Markov en passant par une **modélisation modulaire** en Réseaux de Petri Stochastiques Généralisés. Cette méthode, qui est décrite dans le troisième paragraphe, consiste à construire différents modèles globaux en se basant sur un ensemble de sous-modèles donné. Le paragraphe III.4 détaille les réseaux de base développés ainsi que, à titre d'exemple, différents réseaux globaux formés à partir de ces réseaux de base.

III.1 Evaluation ordinale

L'évaluation ordinale est destinée à identifier, classer et ordonner les défaillances. Plusieurs méthodes et techniques d'évaluation ordinale sont intégrées dans le développement et la production d'un système afin d'éviter les défaillances en vie opérationnelle. Ces méthodes sont employées à tous les stades de la conception afin de minimiser les coûts dus aux éventuelles modifications de la conception. Dans ce paragraphe, nous allons présenter les méthodes qui nous semblent les plus importantes, les sous-paragraphes III.1.1 et III.1.2 décrivant les méthodes les plus couramment utilisées dans le domaine de l'automobile.

III.1.1 AMDEC

Le principe fondamental de l'AMDEC (Analyse des Modes de Défaillance, de leurs Effets et de leur Criticité) est de déterminer de manière systématique pour chaque composant d'un système l'ensemble des modes de défaillance du composant et les conséquences de ces défaillances au niveau système. En général, l'AMDEC est un tableau dans lequel on détaille pour chaque mode de défaillance ses causes possibles, ses effets, sa criticité, les moyens de détection utilisables et les actions correctrices à mettre en œuvre. Il s'agit alors d'une méthode progressant des causes vers les effets, c'est pour cela qu'on l'appelle parfois méthode inductive. Le tableau d'AMDEC fait également apparaître les points importants qui doivent être testés et permet d'associer une ou plusieurs causes à un effet lorsqu'une défaillance se présente. L'efficacité de l'approche par AMDEC est limitée par la complexité du système impliquant la manipulation d'un volume important d'informations, souvent il est difficile de déterminer tous les modes de défaillance des composants du système. Cela est surtout vrai dans le cas des systèmes actuels embarqués sur l'automobile qui contiennent un nombre important de composants numériques (microprocesseur, mémoire, ...) et qui peuvent présenter une multitude de modes de défaillance. De plus, l'AMDEC ne permet pas de prendre en compte les défaillances multiples.

III.1.2 Arbre de fautes

La méthode d'Arbre de fautes (AdF) consiste à rechercher les combinaisons d'événements pouvant conduire à un événement indésirable, telle qu'une défaillance catastrophique. L'AdF permet alors de prendre en compte des combinaisons de défaillances qui échappent à l'AMDEC, c'est pour cela qu'il est souvent utilisé en complément de l'AMDEC.

Le principe de l'AdF est de représenter graphiquement les combinaisons de défaillances au moyen d'une structure arborescente. Par conséquent, un AdF est formé de niveaux successifs d'événements reliés par des portes (opérateurs logiques "ET", "OU", ...), l'événement indésirable constituant la racine de l'arbre. Chaque événement situé en sortie d'une porte donnée est obtenu par la combinaison des événements situés en entrée de cette porte. Le principe de construction d'un AdF consiste à décomposer chaque événement rencontré, en partant de l'événement racine, jusqu'à arriver sur des événements jugés élémentaires. Si les événements élémentaires sont indépendants entre eux et si les probabilités d'occurrence de ces événements peuvent être estimées, le calcul de la probabilité d'occurrence de l'événement indésirable peut être envisagé. Ce calcul se simplifie lorsqu'on trouve les *coupes minimales* : une coupe est un ensemble d'événements entraînant l'événement indésirable ; une coupe est dite minimale lorsqu'elle n'en contient aucune autre. La structure de l'arbre de fautes pouvant être exprimée par des relations booléennes, on

applique les règles de l'algèbre de Boole afin de simplifier (ou réduire) l'équation reliant l'événement indésirable aux événements élémentaires et de trouver ainsi la coupe minimale (parfois, on utilise l'expression "chemin critique" pour "coupe minimale"). En résumé, l'AdF est une méthode progressant des effets vers les causes, c'est pour cela qu'on l'appelle aussi méthode déductive.

III.1.3 Table de vérité

La méthode de la table de vérité (MTV) [Villemeur 1988] consiste à recenser toutes les combinaisons d'états de fonctionnement et de défaillance des composants du système. Pour chaque combinaison d'états, on étudie les effets et la criticité. Cette méthode combine donc la méthode d'AMDEC avec la méthode d'AdF, elle est théoriquement la plus complète que l'on puisse s'imaginer. Cependant, la MTV n'est plus applicable dès que le nombre de composants est grand, étant donné le nombre important de combinaisons à considérer.

III.2 Evaluation probabiliste

L'évaluation probabiliste est destinée à évaluer en termes de probabilités le degré de satisfaction de certains des attributs de la sûreté de fonctionnement (par exemple, la sécurité). L'évaluation probabiliste s'appuie sur deux méthodes, à savoir la méthode analytique et la méthode expérimentale :

1) **La méthode analytique** consiste à construire un modèle axiomatique du système à partir des processus stochastiques élémentaires qui modélisent le comportement des composants du système et leurs interactions. Le traitement de ce modèle permet d'obtenir les valeurs des mesures de la sûreté de fonctionnement du système.
   On peut également élaborer un modèle empirique afin de pouvoir simuler le comportement physique du système. L'injection de fautes dans ce modèle pendant la simulation permet d'évaluer la sûreté de fonctionnement du système.
   La méthode analytique souffre d'une limitation principale : on ne peut pas modéliser tous les aspects d'un système, car il manque souvent l'information détaillée concernant la mise en œuvre. Par ailleurs, la complexité des modèles est limitée afin qu'ils puissent rester exploitables. Par conséquent, il est difficile d'avoir une confiance suffisante dans les résultats des évaluations menées.

2) **La méthode expérimentale** consiste à injecter des fautes dans un modèle physique, un prototype, ou un système réel produit en grande série. L'effort expérimental est nécessaire afin de tester le comportement, en présence de fautes, des mécanismes de tolérance aux fautes du système évalué. Les

résultats obtenus lors de l'injection de fautes permettent donc de dégager des conclusions concernant la sûreté de fonctionnement du système. Le défaut principal de la méthode expérimentale est surtout l'obligation de l'appliquer tardivement dans le processus de développement, la découverte de fautes de conception demande en général des corrections coûteuses et fastidieuses. De plus, lors de l'injection de fautes il est impossible de couvrir entièrement le domaine des fautes qui pourraient avoir lieu pendant la vie opérationnelle du système réel.

Les deux méthodes sont nécessairement complémentaires, car l'évaluation de la sûreté de fonctionnement est une approche itérative qui doit étroitement mêler évaluations analytique et expérimentale.

Dans le cadre de notre travail de recherche, nous ne disposons ni de système réel, ni de modèle empirique des systèmes embarqués. Par conséquent, nous nous limitons aux méthodes analytiques basées sur des modèles axiomatiques. Certaines de ces méthodes sont décrites dans les sous-paragraphes III.2.1, III.2.2 et III.2.3. Dans l'industrie informatique, de telles méthodes ont été employées lors de la conception de systèmes tolérant les fautes du matériel (voir [Siewiorek & Swarz 1992]), influençant ainsi le choix de l'architecture ou de la politique de maintenance. Dans l'industrie automobile, par contre, les évaluations probabilistes ne sont que très rarement employées actuellement.

La complexité globale des modèles développés rend l'emploi d'un outil informatique souvent indispensable. C'est pour cela que des logiciels spécifiques aux méthodes ont été développés, assistant les ingénieurs de conception dans leur tâche d'évaluation. Les méthodes d'évaluation tiennent compte essentiellement des fautes physiques en opération, et elles sont basées sur l'hypothèse de *fiabilité stabilisée*, c'est-à-dire que les taux de défaillance sont supposés constants. Notons cependant que quelques travaux prennent en compte des fautes de conception et de la croissance de fiabilité dans les modèles axiomatiques [Laprie *et al.* 1990, Kanoun *et al.* 1993].

Dans le cadre de l'évaluation probabiliste de la sûreté de fonctionnement des systèmes embarqués sur l'automobile, nous souhaitons quantifier différentes mesures. Le paragraphe III.2.4 expose les mesures probabilistes les plus couramment utilisées dans la littérature, avec un effet de loupe sur les mesures dans le domaine de l'automobile. Quelques outils informatiques, permettant une modélisation par RdPSG, sont présentés dans le paragraphe III.2.5.

III.2.1 Diagrammes de fiabilité

La méthode des diagrammes de fiabilité (en anglais : Reliability Block Diagram Method) est utilisée dans de nombreux domaines industriels pour les systèmes non réparables afin d'évaluer leur fiabilité (voir aussi [Villemeur 1988]). Elle consiste à

construire un modèle qui est basé sur une analyse du fonctionnement du système. Ce modèle met en évidence les entités (composants, sous-systèmes, fonctions, etc.) et leurs relations, car il est structuré en plusieurs blocs liés entre eux, chaque bloc représentant une entité. Deux règles principales de construction du modèle peuvent être formulées :

1) Un bloc dont la défaillance entraîne la défaillance du système, est placé en série avec l'ensemble des autres blocs.

2) Un bloc dont la défaillance ne provoque la défaillance du système qu'en combinaison avec d'autres blocs, est placé en parallèle avec l'ensemble de ces blocs.

Sous l'hypothèse que l'on peut estimer les taux de défaillance de chaque bloc et que les défaillances des blocs sont indépendantes entre elles, on peut calculer la fiabilité du système. Sous certaines conditions, les diagrammes de fiabilité permettent également de calculer la disponibilité et la maintenabilité des systèmes réparables.

III.2.2 Chaînes de Markov

La méthode des chaînes de Markov (voir également [Siewiorek & Swarz 1992]) permet d'analyser et d'évaluer la sûreté de fonctionnement des systèmes, elle est particulièrement adaptée aux systèmes réparables. Lors de la construction d'une chaîne de Markov, deux aspects doivent être pris en compte :

a) Les processus stochastiques relatifs à la défaillance des composants, c'est-à-dire à l'occurrence d'une faute dans le système.

b) Les processus stochastiques relatifs au traitement de fautes et d'erreurs, c'est-à-dire à l'éventuelle restauration du service par des mécanismes de tolérance aux fautes.

Par conséquent, les chaînes de Markov sont constituées de *sommets* représentant l'état de fonctionnement ou de défaillance d'un système ainsi que d'*arcs* entre ces sommets, représentant les processus stochastiques de défaillance et de réparation. Les *chaînes de Markov homogènes* sont construites sous les hypothèses que les processus stochastiques suivent une distribution exponentielle et que les taux de défaillance (et de réparation) sont constants. Il existe également d'autres méthodes dérivées, par exemple les *chaînes de Markov non-homogènes* ou les *processus semi-markoviens*, pour lesquelles ces hypothèses ne sont pas nécessaires. Nous nous limitons par la suite aux chaînes de Markov homogènes que l'on appelle tout simplement chaînes de Markov.

A titre d'exemple, la *Figure 3-1* montre une chaîne de Markov pour un système constitué de deux composants redondants, appelés A et B, sans contrainte sur le nombre de réparations concomitantes. Les arcs sont étiquetés avec les taux de

défaillance et de réparation spécifiques au composant, les sommets contiennent l'information sur l'état de chaque composant (on parle aussi de *graphe d'état*).

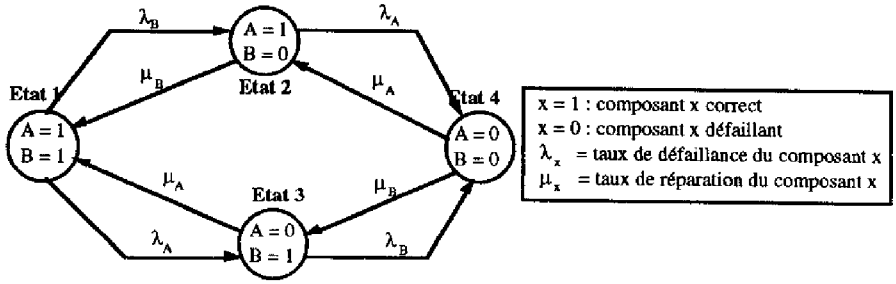


Figure 3-1 : Chaîne de Markov d'un système avec deux composants redondants

Le problème principal de la construction des chaînes de Markov concerne la maîtrise de l'explosion du nombre d'états, car le processus de modélisation implique l'énumération de tous les états possibles et de toutes les transitions entre ces états. Par exemple, le graphe d'état associé à un système avec N composants redondants peut contenir jusqu'à 2^N états (chaque composant pouvant être dans 2 états diférents).

D'une manière générale, les équations associées au graphe d'état permettent d'exprimer l'évolution temporelle de la probabilité d'être dans un état. Sous l'hypothèse que les processus de défaillance et de réparation suivent une loi exponentielle à taux constant, ces équations peuvent être formulées de manière suivante :

$$\left[\frac{dP_1(t)}{dt}, \frac{dP_2(t)}{dt}, \ldots\ldots, \frac{dP_n(t)}{dt}\right] = \left[P_1(t), P_2(t), \ldots\ldots, P_n(t)\right] * \Lambda$$

$$\Rightarrow \mathbf{P}(t) = \mathbf{P}(0) * e^{\Lambda t}$$

- $P_i(t)$ est la probabilité d'être dans l'état i à l'instant t. La résolution de l'équation nécessite la connaissance des probabilités initiales $P_i(0)$.
- n est le nombre d'états du graphe.
- Λ est une matrice carrée de dimension n, dont les éléments λ_{ij} $(i \neq j)$ correspondent aux taux de transition de l'état i vers l'état j. Les éléments λ_{ii} sont égaux à l'opposé de la somme des taux de transition de l'état i vers tous les autres états.

Il s'agit alors d'un système linéaire d'équations différentielles du premier ordre dont la résolution est classique en analyse numérique. Les mesures de sûreté de fonctionnement peuvent être obtenues à partir de ce système d'équations. La probabilité d'être dans un état (ou sous-ensemble d'états) peut être définie comme mesure de sécurité, de fiabilité, de disponibilité, etc. Cette définition est faite par le

modéliseur qui détermine les états sûrs et les états fiables d'un système. Dans l'exemple de la *Figure 3-1*, nous supposons que le fonctionnement correct d'un des deux composants redondants est nécessaire et suffisant pour accomplir la fonction spécifiée. Par conséquent, l'ensemble des états 1, 2 et 3 représente le fonctionnement correct du système global (la fonction est accomplie). L'état 4 correspond alors à la défaillance du système global, c'est l'état de non-fiabilité du système. Les effets de la défaillance doivent être analysés afin de savoir si le non-accomplissement de la fonction a des conséquences sur la sécurité des utilisateurs du système. Au cas où la perte de la fonction est une défaillance catastrophique (pour ses utilisateurs), l'état 4 serait un état de non-sécurité.

La modélisation par chaînes de Markov permet également de superposer des *structures de récompenses* [Howard 1971]. L'établissement d'une structure de récompense consiste à affecter des *taux de récompense* à certains états et des *bonus* à certains événements. Cela veut dire que le temps de séjour dans les états appartenant à la structure de récompense est cumulé sur un "compte" et récompensé par le taux d'intérêt affecté. Lors de l'occurrence d'un événement appartenant à la même structure de récompenses, on verse le bonus affecté sur le même compte. Cette approche permet d'évaluer des mesures de coût-sûreté de fonctionnement [Moreira de Souza & Landrault 1981] ou de performabilité [Meyer 1982].

III.2.3 Les Réseaux de Petri

Les Réseaux de Petri Stochastiques Généralisés (RdPSG) constituent un moyen adéquat pour la construction des chaînes de Markov. La possibilité de générer une chaîne de Markov à partir d'un RdPSG peut donner lieu à des évaluations quantitatives (études paramétriques) de la sûreté de fonctionnement et constitue, en plus, une étape dans la validation des modèles RdPSG développés. Les RdPSG sont directement dérivés des Réseaux de Petri classiques qui permettent de modéliser et de visualiser des comportements comportant du parallélisme, de la compétition, de la synchronisation et du partage de ressource. Dans le but de mieux introduire les RdPSG, nous présentons dans ce paragraphe différentes classes de Réseaux de Petri qui constituent la base des RdPSG ou qui leur sont étroitement liées.

III.2.3.1 Les Réseaux de Petri classiques

Un réseau de Petri (RdP) est un *graphe orienté biparti* constitué de deux types de sommets, à savoir les *places* et les *transitions*, qui sont reliés par des *arcs*. Chaque place peut contenir 0, 1 ou plusieurs *jetons*, on parle également de *marquage de la place*. Le *marquage du réseau* est donné par le marquage de toutes les places.

Un réseau de Petri a un comportement dynamique, c'est-à-dire que son marquage peut évoluer. L'évolution du marquage est déterminé par les *transitions tirables*

(aussi : *transitions franchissables*). Une transition est tirable si et seulement si chacune des places reliées en amont de la transition contient au moins w jetons, w étant le *poids* (anglais : *weight*) de l'arc correspondant (sur la représentation graphique du réseau de Petri, le poids d'un arc n'est indiqué que lorsqu'il diffère de la valeur 1). Lors du tir d'une transition, on enlève des jetons des places en amont et on insère des jetons dans les places en aval. Les nombres de jetons enlevés et insérés sont égaux aux poids des arcs correspondants. Lorsqu'une place est liée en amont d'une transition par un arc de poids nul (appelé *arc inhibiteur*), la transition n'est tirable que si la place ne contient aucun jeton. Le réseau est *pur* s'il n'y a pas de transition ayant une place en amont qui soit également une place en aval de cette transition. On parle de *conflit* entre transitions lorsque le tir d'une transition désensibilise une autre transition. Par exemple, si une place est en amont de deux ou plusieurs transitions et s'il n'y a qu'un jeton dans cette place, il y a un conflit entre les transitions. Une seule transition pouvant être franchie, il faut alors pouvoir prendre une décision (par exemple, par l'intermédiaire de priorités).

A partir d'un marquage initial donné, on obtient donc un ensemble de marquages successifs (cet ensemble pouvant être vide). La connaissance de tous les marquages accessibles à partir d'un marquage initial permet de construire le *graphe de marquages accessibles* (appelé également *séquence de tir*).

Formellement, un Réseau de Petri peut être défini de la manière suivante :

- $P = \{p_1, p_2, ... p_{np}\}$ est l'ensemble des places, avec np = nombre de places
- $T = \{t_1, t_2, ... t_{nt}\}$ est l'ensemble des transitions, avec nt = nombre de transitions
- **Pre** est la matrice d'incidence avant. Elle fait correspondre à chaque place de P un sous-ensemble de T appelé *transitions suivantes*.
- **Post** est la matrice d'incidence arrière. Elle fait correspondre à chaque transition de T un sous-ensemble de P appelé *places suivantes*.
- **C** = **Post** - **Pre** est la matrice d'incidence, de dimension nt x np.
- $\mathbf{M}_i = (M_i(p_1), M_i(p_2), ... M_i(p_{np}))^T$ est un vecteur colonne qui contient le marquage i de chaque place du réseau (un nombre entier, positif ou nul). En particulier, $\mathbf{M}_0$ est le vecteur de marquage initial.

La relation de tir (ou de marquage suivant) de la transition t_i peut s'écrire sous forme matricielle :

$$\text{si } \mathbf{M}_k - \mathbf{Pre}^T * \mathbf{U}_{t_i} \geq 0 \quad \text{alors} \quad \mathbf{M}_{k+1} = \mathbf{M}_k + \mathbf{C}^T * \mathbf{U}_{t_i}$$

(sinon, t_i n' est pas tirable à partir de $\mathbf{M}_k$)

avec :

- $\mathbf{U}_{t_i}$ = vecteur de dimension *nt* où seule la *i*-ème composante est égale à 1, les autres composantes étant égales à 0.
- $\mathbf{M}_{k+1}$ = Marquage après le tir de t_i
- $\mathbf{M}_k$ = Marquage avant le tir de t_i

Si un marquage $\mathbf{M}_k$ est accessible depuis le marquage initial $\mathbf{M}_0$, il existe alors une séquence de tir de transitions depuis $\mathbf{M}_0$ à $\mathbf{M}_k$, ce qui revient à formuler l'équation :

$$\mathbf{M}_k = \mathbf{M}_0 + \mathbf{C}^T * \mathbf{Z}$$
$$\text{avec : } \mathbf{Z} = (\mathbf{U}_{t_1} + \mathbf{U}_{t_2} + \ldots + \mathbf{U}_{t_{nt}})$$

Dans ce contexte, nous définissons plusieurs propriétés des réseaux de Petri :

- On parle d'un *réseau borné* lorsque l'ensemble des marquages accessibles à partir du marquage initial $\mathbf{M}_0$ est fini, ce qui revient à dire que le marquage de chaque place est borné. Un cas particulier du réseau borné est le *réseau sauf* où il y a au plus un jeton dans chaque place.
- Un *blocage* est un marquage tel qu'aucune transition ne peut être franchie.
- Le réseau est *vivant* pour un marquage initial $\mathbf{M}_0$ si toutes les transitions sont toujours franchissables à partir de $\mathbf{M}_0$.
- Le réseau est *quasi-vivant* pour un marquage initial $\mathbf{M}_0$ si toutes les transitions sont tirées au moins une fois à partir de $\mathbf{M}_0$.

A partir de l'équation $\mathbf{M}_k = \mathbf{M}_0 + \mathbf{C}^T * \mathbf{Z}$, on définit un *invariant de tir* par le vecteur colonne **S** (de dimension *nt*) tel que :

$$\mathbf{M}_i = \mathbf{M}_i + \mathbf{C}^T * \mathbf{S} \quad (\Rightarrow \mathbf{C}^T * \mathbf{S} = 0)$$

S représente alors une séquence de tir qui ramène à un marquage de départ $\mathbf{M}_i$; la séquence de tir peut être répétée indéfiniment, c'est pour cela qu'on l'appelle également *séquence répétitive*. Les invariants de tir peuvent donner une idée du comportement "cyclique" d'un réseau, mais leur interprétation est difficile lorsque plusieurs séquences répétitives sont possibles. Néanmoins, on peut conclure que lorsque toutes les transitions figurent dans les invariants de tir, le réseau est vivant.

Un *invariant de places* est le vecteur colonne **Q** (de dimension *np* ; chaque q_i est un *poids*, donc un nombre entier positif ou nul) qui vérifie l'équation :

$$\mathbf{Q}^T * \mathbf{M}_i = \mathbf{Q}^T * \mathbf{M}_0 = \mathbf{K} \quad (\Rightarrow \mathbf{C} * \mathbf{Q} = 0)$$

pour tout marquage $\mathbf{M}_i$ accessible à partir de $\mathbf{M}_0$. **K** est une constante appelée *invariant de marquage*. Le sous-ensemble de places à poids non nul figurant dans **Q** est appelé *composante conservative*. Les invariants de marquage permettent de

vérifier que le nombre de jetons dans la composante conservative est constant et, par conséquent, borné. Souvent, ils peuvent être interprétés comme une liste des circuits ou chemins bloquants qui peuvent être parcourus par un jeton, ce qui leur donne en général une signification physique. C'est pour cela que les invariants de marquage sont davantage utilisés que les invariants de tir.

III.2.3.2 Les Réseaux de Petri Stochastiques

Les Réseaux de Petri Stochastiques (notés aussi RdPS) sont obtenus à partir des Réseaux de Petri classiques, en associant une durée de franchissement aléatoire aux transitions [Florin & Natkin 1978, Molloy 1982]. La durée de temps qui s'écoule entre la sensibilisation et le tir effectif d'une transition est distribuée selon une loi exponentielle négative de taux λ. Il importe de noter qu'une place peut être suivie de plusieurs transitions temporisées, et que le tir de ces transitions est basé sur le modèle compétitif [Howard 1971]. Le marquage du RdPS est un processus markovien, c'est-à-dire que le graphe de marquages associé au RdPS correspond à une chaîne de Markov. On étiquette les arcs par leurs taux de franchissement, ces taux pouvant être constants ou des fonctions du marquage de certaines places.

III.2.3.3 Les Réseaux de Petri Stochastiques Généralisés

Les RdPS ont été étendus aux Réseaux de Petri Stochastiques Généralisés (notés RdPSG) par [Marsan *et al.* 1984] en introduisant des transitions instantanées qui sont tirées sans délai et qui sont prioritaires par rapport aux transitions temporisées à délai aléatoire. Lors de la description graphique d'un RdPSG, les transitions instantanées sont représentées par un rectangle noir et les transitions temporisées par un rectangle blanc. La présence de deux types de transitions amène à définir deux types de marquages, à savoir les *marquages tangibles* qui sensibilisent des transitions temporisées et les *marquages transitoires* qui sensibilisent des transitions instantanées. Lorsqu'un marquage transitoire sensibilise plusieurs transitions instantanées, on définit une *sélection aléatoire* qui consiste à associer à chaque transition tirable une probabilité de tir telle que la somme de ces probabilités soit égale à un. Le passage du graphe de marquages à la chaîne de Markov se fait en ne conservant que les marquages tangibles. Les taux associés aux arcs de la chaîne de Markov se calculent de la façon suivante : on multiplie le taux associé à la transition temporisée dont le tir génère un marquage transitoire par la probabilité de chemin à travers des marquages transitoires jusqu'au prochain marquage tangible. Si un marquage tangible est suivi d'un autre marquage tangible, le taux associé est directement dérivé de la transition temporisée. Notons également que l'on peut associer une structure de récompenses aux RdPSG qui permet une génération automatique des chaînes de Markov avec une structure de récompense.

III.2.3.4 D'autres classes de Réseaux de Petri Stochastiques

A part les RdPSG, d'autres classes de RdPS ont été proposées, qui peuvent être caractérisées en fonction du type de la durée de franchissement qu'elles peuvent manipuler. On peut citer par exemple : les Réseaux de Petri Stochastiques à distributions discrètes [Holliday & Vernon 1987] ; les Réseaux de Petri Stochastiques Etendus [Dugan *et al.* 1984] qui permettent des fonctions de distribution quelconques ; les Réseaux de Petri Stochastiques et Déterministes [Marsan & Chiola 1986] qui combinent des délais exponentiellement distribués et des délais constants (pouvant être égaux à zéro) ; les Réseaux de Petri Temporisés Stochastiques (RdPTS) introduits par les travaux de [Atamna 1994] au LAAS et qui permettent de décrire une grande variété de distributions temporelles.

III.2.4 Mesures probabilistes

Lorsque l'on évalue la performance et la sûreté de fonctionnement d'un système, on peut choisir entre un grand nombre de mesures. Parmi ces mesures, nous focalisons sur celles qui sont utilisées le plus couramment dans la littérature (voir Annexe pour les définitions mathématiques de ces mesures), et nous proposons dans le paragraphe III.2.4.1 une classification générale de ces mesures. Cette classification est un guide pour le choix des mesures, et nous présentons les mesures retenues pour le domaine de l'automobile dans le paragraphe III.2.4.2.

III.2.4.1 Classification générale des mesures

Concernant l'évaluation de la sûreté de fonctionnement, on peut regrouper les mesures en deux catégories. La *première catégorie* de mesures caractérise le comportement du système soumis seulement au processus de défaillance, sans tenir compte d'une éventuelle réparation. Parmi les mesures présentées dans l'Annexe, la fiabilité ponctuelle, la sécurité ponctuelle, la disponibilité avant défaillance catastrophique, le *MTTF* ainsi que le *MTFF* appartiennent à cette catégorie. Elles évaluent le temps d'accomplissement du service correct avant une défaillance (qui est considérée comme état absorbant). La *deuxième catégorie* de mesures caractérise le comportement du système soumis simultanément aux deux processus de défaillance et de réparation. Parmi les mesures présentées dans l'Annexe, la disponibilité (ou l'indisponibilité) ponctuelle et asymptotique, la maintenabilité ponctuelle, le *MUT*, le *MDT* ainsi que le *MTBF* appartiennent à cette catégorie. Elles évaluent le temps d'accomplissement ou de non-accomplissement du service correct, en tenant compte de l'alternance entre service correct et service incorrect.

Concernant l'évaluation de la performance, on introduit des mesures de type revenu qui permettent par exemple d'évaluer le temps moyen passé dans un certain état du système ou bien de compter le nombre d'occurrences d'un événement.

Il existe également des mesures qui généralisent les mesures de sûreté de fonctionnement en y associant des mesures de performance. Ces mesures, souvent dénommées mesures de performabilité, sont d'un grand intérêt pour les systèmes informatiques *multi-performants*. Parmi les systèmes multi-performants, on peut compter ceux qui ont différents modes de service corrects et incorrects. Par exemple, les systèmes multiprocesseurs présentent plusieurs niveaux de performance de par leur capacité à supporter des modes d'accomplissement dégradés, ils possèdent des mécanismes de tolérance aux fautes et se trouvent dans un mode dégradé lors de (et après) leur phase d'"auto-maintenance".

Le choix des mesures dépend considérablement de la fonction et de l'utilisation du système étudié. Les mesures de la première catégorie sont employées pour évaluer la sûreté de fonctionnement des fonctions pour lesquelles la réparation n'est pas envisagée ou envisageable (par exemple, pour certains systèmes spatiaux). L'emploi des mesures de la deuxième catégorie est très fréquent afin d'évaluer la sûreté de fonctionnement des fonctions qui sont destinées à une utilisation continue de longue durée. Cela est par exemple le cas pour certaines fonctions informatiques des systèmes de commutation téléphonique, des centres de contrôle de navigation aérienne, des postes d'aiguillage des trains ou bien de l'informatique bancaire.

Pour des systèmes à deux degrés de défaillance, à savoir la défaillance bénigne (le service du système est simplement dégradé) et la défaillance catastrophique, la sécurité est la mesure du temps de séjour dans l'état sûr (regroupant les états d'accomplissement du service correct et les états de défaillance bénigne) avant défaillance catastrophique. La mesure de sécurité constitue ainsi une mesure de type fiabilité, les états absorbants étant caractérisés par le mode de service incorrect catastrophique. La mesure de la disponibilité (c'est-à-dire, la mesure de la délivrance d'un service sûr par rapport à l'alternance "état sûr — défaillance catastrophique") ne serait pas très significative. C'est pour cela que nous sommes conduit à mesurer la délivrance du service correct par rapport à l'alternance "service correct — défaillance bénigne", avant occurrence d'une défaillance catastrophique. Cette mesure, dénommée disponibilité avant défaillance catastrophique, permet de quantifier l'habituel compromis entre fiabilité (ou disponibilité) et sécurité.

III.2.4.2 Mesures dans le domaine de l'automobile

L'automobile est un produit qui est utilisé de façon discontinue, le temps total de roulage étant relativement court (par exemple, 10 ans d'utilisation correspondent en moyenne à seulement 5000 heures de fonctionnement). Lors de l'évaluation de la sûreté de fonctionnement des systèmes informatiques embarqués sur l'automobile, on fait généralement abstraction des temps d'arrêt. Les mesures de la deuxième catégorie présentent donc peu d'intérêt, et nous nous limitons aux mesures de la première catégorie.

- D'une manière générale, les mesures du *MTTF* et du *MTFF* sont faciles à interpréter, mais moins riches en informations (et — ainsi — parfois trompeuses) que les autres mesures de la première catégorie ; par conséquent, nous ne poursuivons pas leur analyse.
- La fiabilité ponctuelle peut être une mesure intéressante pour les systèmes qui sont sollicités en permanence et dont la défaillance n'a généralement pas de conséquences catastrophiques. Dans le passé, une grande part des systèmes électroniques embarqués sur l'automobile étaient des systèmes de ce type (par exemple, le système de contrôle moteur).
- A l'heure actuelle, les constructeurs automobile offrent de plus en plus de systèmes électroniques dont la défaillance pourrait avoir des effets catastrophiques (par exemple, le coussin gonflable ou l'antiblocage des roues). Pour ces systèmes, les conséquences d'une défaillance catastrophique sont généralement d'une ampleur telle que la restauration du service n'est pas primordiale. En effet, la restauration du service devient secondaire par rapport au coût et à la durée de la réparation des conséquences de la catastrophe. Par conséquent, l'évaluation de la sûreté de fonctionnement de tels systèmes est essentiellement basée sur les mesures de la sécurité ponctuelle.
- Dans le futur, de plus en plus de systèmes informatiques embarqués sur l'automobile seront des systèmes à différents degrés de défaillance. C'est pour cela que la mesure de la disponibilité avant défaillance catastrophique — actuellement non utilisée dans le domaine de l'automobile — pourrait gagner de l'intérêt.

Afin d'évaluer la fiabilité ou la sécurité des systèmes informatiques embarqués, le constructeur automobile s'intéresse généralement au nombre de systèmes défaillants pendant un an d'utilisation (c'est-à-dire, 500h de fonctionnement). Ce nombre est mis en rapport au nombre de systèmes produits pendant ce temps et est multiplié par 10^6. On obtient ainsi une expression que nous proposons de noter n, et qui est exprimée dans l'unité *ppm* (partie par million de systèmes produits). En multipliant par 10^6 la valeur de la non-sécurité d'un système donné à l'instant t = 500h, nous obtenons n dans l'unité *ppm*. Dans le cadre de nos travaux, nous allons évaluer la sûreté de fonctionnement de deux fonctions ayant un impact sur la sécurité, à savoir le coussin gonflable et la direction électronique. Le coussin gonflable est un système qui possède deux modes de défaillance catastrophique (le non-gonflement lors de la sollicitation et le gonflement intempestif) et un mode appelé défaillance potentiellement catastrophique où il n'est pas prêt à se gonfler lors d'un accident éventuel. C'est pour cela que nous proposons d'évaluer non seulement n, mais également une mesure que nous appelons *indisponibilité sachant qu'il n'y pas*

accident et qui sera notée *UAA* (anglais : UnAvailability before Accident). En ce qui concerne la direction électronique, seulement *n* sera évalué.

On note ici que, dans d'autres secteurs industriels telle l'industrie aéronautique, on exprime l'objectif de sécurité d'un système généralement par la probabilité d'occurrence d'une défaillance catastrophique pendant une unité de temps ; cette mesure est le taux de défaillance catastrophique du système λ_c, exprimé le plus souvent dans l'unité *1/h*. Ce taux regroupe généralement un ensemble de défaillances élémentaires conduisant à la défaillance catastrophique du système global. On suppose généralement que le temps avant défaillance catastrophique suit approximativement une distribution exponentielle, le taux λ_c peut alors être obtenu en calculant la pente à l'origine de la sécurité ponctuelle.

III.2.5 Outils basés sur les RdPSG et les chaînes de Markov

L'utilisation des méthodes markoviennes induit souvent des modèles de grande dimension et, par conséquent, des calculs très complexes. C'est pour cela que des outils informatiques sont indispensables afin d'évaluer la sûreté de fonctionnement. Ces outils sont destinés à fournir des valeurs numériques pour des mesures de sûreté de fonctionnement, voire de performance. Les premiers programmes informatiques d'évaluation des graphes d'états sont apparus à partir du milieu des années 1970, notamment dans le domaine nucléaire. Dans [Arlat 1988, Villemeur 1988, Geist & Trivedi 1990, Siewiorek & Swarz 1992], on peut trouver une liste de plusieurs logiciels permettant l'évaluation de la sûreté de fonctionnement. Certains de ces outils basent leurs calculs sur des simulations (on parle aussi de simulation "Monte-Carlo"), d'autres créent la chaîne de Markov associée à un RdPSG afin de calculer les mesures spécifiées. Ces derniers peuvent être intéressants, car ils permettent de valider la structure du Réseau de Petri en analysant la chaîne de Markov associée. Cette validation est néanmoins limitée à des modèles markoviens d'une certaine complexité en-dessous de laquelle il est possible d'effectuer une analyse "manuelle" du modèle markovien et d'obtenir des formules analytiques exprimant par exemple le taux de défaillance catastrophique d'un système.

Nous nous limitons ici aux deux outils que nous avons utilisés, à savoir TOMSPIN et SURF-2.

III.2.5.1 TOMSPIN

L'outil TOMSPIN a été conçu et développé par les laboratoires de recherche de la société SIEMENS à Munich [SIEMENS 1993]. Il permet d'analyser des RdPSG, ces réseaux étant saisis d'une façon textuelle dans un fichier en respectant une syntaxe bien définie. Un aspect très intéressant dans la saisie est qu'il est possible de

décrire le réseau hiérarchiquement (jusqu'à deux niveaux). Dans la plus récente version de TOMSPIN, un éditeur graphique soutient la saisie textuelle. De plus, TOMSPIN donne la possibilité d'affecter des variables aux taux / probabilités de transitions et aux marquages initiaux des places, ce qui permet en théorie d'effectuer une analyse paramétrique. Ici, nous avons découvert le désavantage principal de TOMSPIN (au moins, dans le cadre de nos travaux) qui est l'impossibilité de stocker et d'observer plus qu'une seule analyse transitoire à la fois ; une analyse transitoire paramétrique devient alors très lourde. De plus, on ne peut pas regrouper des places sous des ensembles non fiables et non sûrs, les calculs de TOMSPIN permettent uniquement d'observer l'évolution de la probabilité de marquage de chaque place indépendamment. Afin de calculer par exemple la sécurité, il faudrait alors faire la somme des probabilités d'occurrence des états sûrs.

III.2.5.2 SURF-2

L'outil SURF-2 a été conçu et développé au LAAS [Béounes *et al.* 1993]. Il possède un éditeur graphique qui permet de définir la topologie d'un RdPSG ou d'une chaîne de Markov ainsi que de spécifier ses paramètres (marquage initial, taux et probabilités de transition). Ces paramètres peuvent être des variables dont les valeurs numériques sont définies lors de l'analyse du modèle, ce qui permet en particulier de faire des analyses paramétriques. SURF-2 permet également de définir des *partitions* : l'ensemble des états est subdivisé en les deux sous-ensembles *propre* et *impropre*, l'ensemble des états impropres pouvant contenir un sous-ensemble *catastrophique*. En ce qui concerne les Réseaux de Petri, un état du modèle se traduit par le marquage d'une ou plusieurs places. Lorsque le modèle est dans un des états propres, le système modélisé est considéré comme sûr, disponible et fiable. Dès que le modèle n'est plus dans un des états propres, le système devient par définition non fiable, et il devient non sûr lorsqu'un des états catastrophiques est occupé. Les mesures de sûreté de fonctionnement sont basées sur ces partitions. Par exemple, la disponibilité est définie comme la probabilité d'être dans les états propres et la sécurité est la probabilité de ne pas être dans les états catastrophiques. Afin d'analyser et vérifier la structure du RdPSG, l'outil SURF-2 permet de calculer les invariants de tir et les invariants de marquage. Lors de l'analyse numérique du réseau, SURF-2 transforme le RdPSG en un graphe de marquages. Le calcul du graphe de marquages permet entre autres de vérifier que le réseau est borné et constitue — jusqu'à une certaine complexité — un moyen de validation du réseau. Le graphe de marquages est réduit en une chaîne de Markov qui est la base du calcul des mesures. Cette chaîne de Markov peut être également éditée et stockée. Notons enfin que SURF-2 permet la superposition aux modèles d'une structure de récompenses.

III.3 Méthode de modélisation modulaire par RdPSG

Parmi les différentes techniques d'évaluation de la sûreté de fonctionnement présentées dans les paragraphes III.1 et III.2, la modélisation par chaînes de Markov permettrait d'exprimer fidèlement la logique du comportement du système ainsi que les processus stochastiques temporels de défaillance et de réparation. Etant donné le nombre de fonctions et les possibilités d'architectures présentées dans le chapitre II, il est évident qu'il est très difficile et fastidieux de créer un modèle markovien complet pour chaque combinaison d'architectures et de fonctions. De plus, l'analyse markovienne d'un système complexe peut donner lieu à des graphes de grande taille qui sont difficiles à vérifier et à interpréter. C'est pourquoi il faut absolument utiliser des modèles permettant, d'une part, de fournir des représentations concises et structurées et, d'autre part, de résoudre le problème crucial de l'explosion de l'espace d'états. Sachant que les différentes fonctions et architectures ont des caractéristiques à la fois communes mais aussi distinctes, nous sommes donc conduits à développer une méthode de modélisation modulaire et systématique. Cette approche doit permettre de créer plusieurs modèles globaux basés sur l'interconnexion d'un petit nombre de modèles de base. De plus, le changement d'un modèle de base doit pouvoir s'effectuer sans changer les autres modèles de base. La modélisation par Réseaux de Petri Stochastiques Généralisés (RdPSG) nous semble alors l'approche la plus appropriée, car une caractéristique importante des RdPSG est de pouvoir coupler différents modèles de base (relativement peu complexes) dans le but de générer une chaîne de Markov qui décrit le fonctionnement global d'un système et qui peut être très complexe.

Dans le paragraphe III.3.1, nous justifions le choix des modèles de base nécessaires pour la construction des modèles globaux. Le paragraphe III.3.2 explique ensuite les différentes techniques de couplage entre les modèles de base, et dans le paragraphe III.3.3 nous présentons la méthode de validation des modèles de base. En pratique, les étapes de construction, de couplage et de validation des modèles de base sont étroitement liées et sont effectuées de façon itérative.

III.3.1 Choix des modèles de base

Concernant le choix des modèles de base et compte tenu de notre objectif de comparaison des architectures, les aspects suivants peuvent être dégagés :

- Pour chaque fonction, les effets de défaillance sont spécifiques. Cela signifie que les stratégies de signalisation d'erreur et de réparation diffèrent. La modélisation doit alors permettre de décrire le comportement de plusieurs fonctions soumises aux processus de défaillance et de réparation.

- Plusieurs architectures sont imaginables pour implanter un ensemble de fonctions données. Il serait alors souhaitable de pouvoir modéliser les aspects fonctionnels indépendamment des aspects d'architecture, tout en donnant la possibilité de lier n'importe quel modèle fonctionnel avec n'importe quel modèle d'architecture.
- L'aspect d'intégration fonctionnelle amène à considérer l'allocation d'un processeur (ou d'un sous-ensemble de processeurs) à plusieurs fonctions. On doit ainsi pouvoir modéliser l'impact d'une architecture (au niveau de la sûreté de fonctionnement) sur plusieurs fonctions à la fois.
- Chaque système embarqué sur l'automobile n'est en opération que durant 500h par an et est soumis à un processus cyclique d'arrêt et de redémarrage. Le redémarrage des systèmes est important car l'autotest qui est effectué permet de détecter des erreurs latentes. Par conséquent, nous devons modéliser les actions d'arrêt et de redémarrage du système. De plus, les actions de réparation et de maintenance doivent être prises en compte.

Les aspects évoqués ici nous amènent à développer trois types de modèles de base à partir desquels nous pouvons construire les modèles globaux qui nous intéressent (cf. *Figure 3-2* pour une illustration) :

- Un modèle de la fonction. Chaque fonction a un comportement spécifique, par conséquent chaque fonction a son propre modèle fonctionnel. Nous distinguons essentiellement entre les états de fonctionnement correct, de défaillance potentiellement catastrophique et de défaillance catastrophique. Ces états permettent également la définition des différentes mesures de sûreté de fonctionnement.
- Un modèle d'activation du véhicule. Ici, nous modélisons des états tels que le roulage, l'accident, la réparation et le redémarrage. Le modèle du véhicule est unique pour tous les modèles globaux.
- Un modèle du calculateur. C'est un modèle qui tient compte du degré de redondance matérielle employé. Nous modélisons une unité de traitement qui peut être simple (architecture Simplex), doublée (architecture Duplex) ou triplée (architecture TMR, "Triple Modular Redundancy"). Lors de la modélisation du calculateur, nous tenons compte des fautes physiques transitoires et permanentes ainsi que du recouvrement des différents types d'erreurs associées.

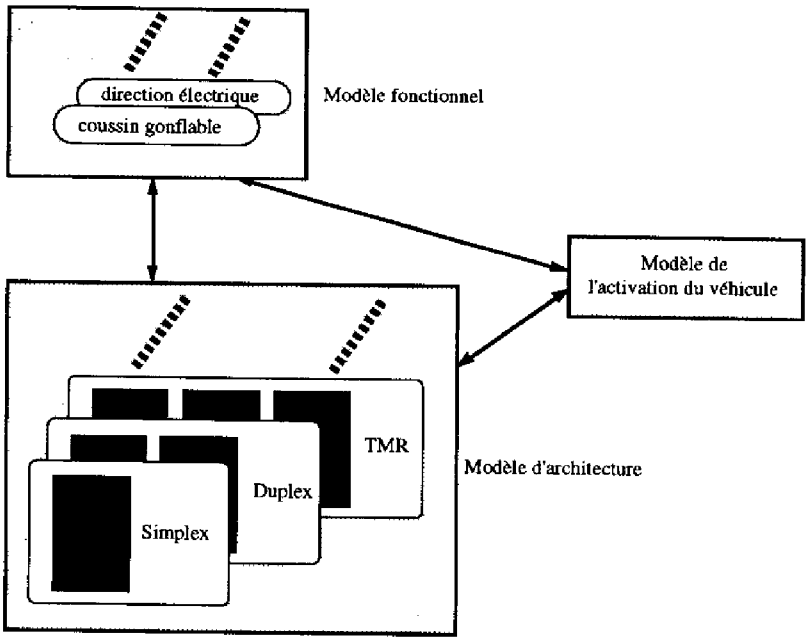


Figure 3-2 : Approche modulaire

La modélisation modulaire par RdPSG est traitée également dans la littérature [Ammar *et al.* 1987, Szczerbicka 1992, Ciardo & Trivedi 1993, Specker 1993], les travaux les plus récents peuvent être trouvés dans [Borrel 1996]. Notre approche se distingue de ces travaux essentiellement par la définition des mesures sur un modèle fonctionnel indépendant du modèle d'architecture ainsi que par la représentation indépendante des processus d'activation du véhicule. Nous distinguons entre fautes permanentes et fautes transitoires au niveau des procédures de recouvrement du calculateur. Une autre caractéristique de nos modèles est de différencier plusieurs types d'erreur et leurs défaillances associées. Cependant, à la différence de [Borrel 1996], nos modèles d'architecture ne différencient pas entre fautes du matériel et fautes du logiciel.

III.3.2 Couplage des modèles de base

Tout modèle global est formé à partir des trois réseaux de base décrits dans le paragraphe précédent. Nous avons choisi de distinguer trois façons de coupler les modèles de base que nous allons présenter successivement dans les trois sous-paragraphes suivants (voir aussi *Figure 3-3*).

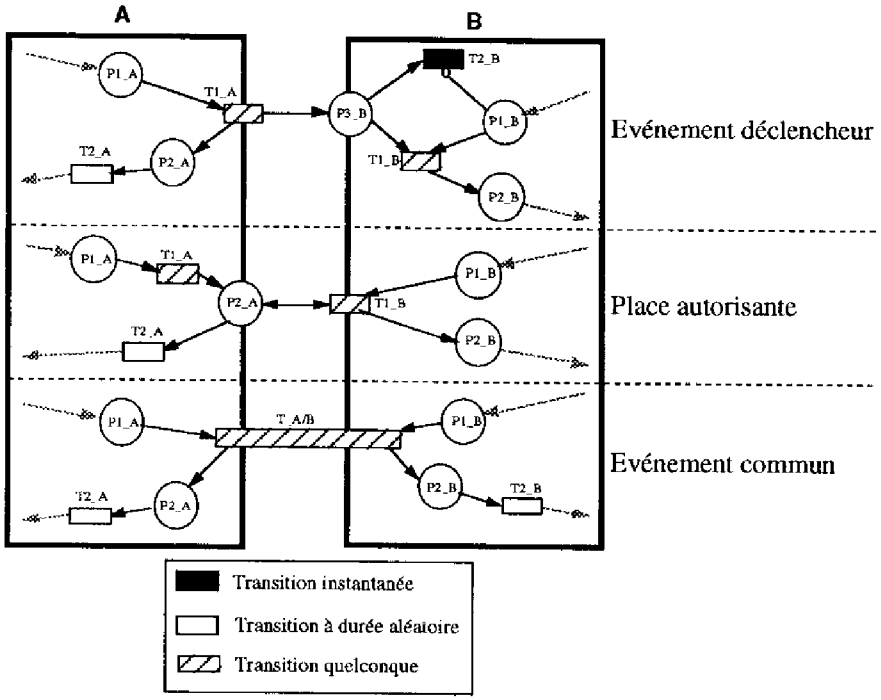


Figure 3-3 : Couplage des modules

III.3.2.1 L'événement déclencheur

L'*événement déclencheur* est une transition d'un module A qui, lors de son franchissement, génère un jeton en destination d'un module B. Le jeton généré véhiculant l'information sur l'occurrence d'un événement, nous l'appelons *jeton d'événement*. Le module B doit disposer d'une place pour recevoir le jeton d'événement. En général, l'unique rôle de cette place est de passer l'information véhiculée dans son propre module, c'est pour cela qu'elle est appelée *place de réception*. En fait, l'événement déclencheur fournit une information sur un événement du module A qui est lue par le module B. Au niveau du module B, l'arrivée d'une information peut être vue comme une interruption qui — en fonction de l'état du module B — est soit masquée, soit autorisée à influencer l'évolution de B. Lors du couplage de deux modules A et B par un événement déclencheur du module A, l'évolution dans B peut alors être influencée dans certains cas par A, sans que l'évolution de A ne soit affectée par celle de B. C'est pour cela que l'événement déclencheur de A est une interface de sortie, tandis que la place de réception de B est l'interface d'entrée correspondante. Dans l'exemple de la *Figure 3-3*, la transition

(temporisée ou instantanée) T1_A du module A est un événement déclencheur qui met un jeton d'événement dans la place de réception P3_B du module B. Deux possibilités se présentent alors :

a) La place P1_B contient un jeton, le module B est déjà en attente de l'événement déclencheur (prise en compte de l'interruption). Le marquage de la place de réception autorise alors le tir de la transition (temporisée ou instantanée) T1_B, l'état du module B évolue.

b) La place P1_B ne contient pas de jeton, le module B n'est pas en attente de l'événement déclencheur (masquage de l'interruption). Dans ce cas, la place P3_B est "vidée" par le tir de la transition instantanée T2_B, l'état du module B n'évolue pas. Donc, on ne prend pas en compte l'événement déclencheur et on choisit d'éviter l'accumulation de jetons dans la place de réception.

Nous conseillons de s'assurer que le franchissement de l'événement déclencheur du module A amène à un état tangible dans son propre module. Cela permet de garantir que toutes les transitions instantanées tirables du module B peuvent être tirées sans être en conflit avec d'éventuelles transitions instantanées du module A. C'est pour cela que, dans l'exemple de la figure, la place P2_A est suivie d'une transition temporisée. Une deuxième règle que nous proposons est de vider la place de réception immédiatement si seulement la place P3_B est marquée (dans l'exemple de la figure, cela est assuré par la transition instantanée T2_B). Il est également important que le processus déclenché dans le module B ne soit pas en concurrence avec d'autres processus intrinsèques au module B. Dans l'exemple de la figure, cela signifie que le jeton dans place P1_B peut être enlevé seulement par la transition T1_B.

III.3.2.2 La place autorisante

Le couplage par *place autorisante* correspond à lier une place d'un module A avec une transition d'un module B par un arc bidirectionnel. Le marquage de la place autorisante conditionne le tir de la transition reliée, c'est pour cela que nous parlons également de *transition conditionnée*. En fait, le marquage de la place autorisante fournit une information sur l'état du module A que l'on prend en compte ou non dans le module B. De façon analogue au couplage par événement déclencheur, on note que l'évolution du module A peut influencer l'évolution du module B, sans que l'évolution dans A ne soit affectée par celle de B. Par conséquent, la place autorisante du module A peut être vue comme une interface de sortie, et la transition conditionnée du module B comme une interface d'entrée. Dans l'exemple de la *Figure 3-3*, la place P2_A du module A est liée avec la transition (temporisée ou instantanée) T1_B du module B par un arc bidirectionnel. En cas de marquage de la place P2_A, deux cas de figure peuvent se présenter :

a) Le place P1_B contient un jeton, le module B est déjà en attente du marquage de la place autorisante. Par conséquent, la transition conditionnée est franchie, l'état du module B évolue. La présence d'un arc bidirectionnel garantit que la place autorisante ne perde pas son jeton lors du franchissement de la transition conditionnée.

b) La place P1_B ne contient pas de jeton, le module B n'est pas en attente du marquage de la place autorisante. Par conséquent, l'évolution du modèle A continue (franchissement de T2_A) sans influencer celle de B, le marquage de la place autorisante n'est pas pris en compte.

Nous conseillons de s'assurer que le marquage de la place autorisante corresponde à un état tangible du module A. Cela permet d'éviter un conflit entre d'éventuelles transitions instantanées des deux modules. C'est pour cela que, dans l'exemple de la figure, la place P2_A est suivie d'une transition temporisée. Il est également important que le processus autorisé dans le module B ne soit pas en concurrence avec d'autres processus intrinsèques au module B. Dans l'exemple de la figure, cela signifie que le jeton dans la place P1_B peut être enlevé seulement par la transition T1_B.

III.3.2.3 L'événement commun

L'*événement commun* (appelé aussi *rendez-vous*) consiste à construire une synchronisation réciproque des évolutions de deux modules. Le franchissement de l'événement commun est conditionné par l'évolution des deux modules et il détermine également l'évolution "future" dans chacun des deux modules, il est alors à la fois interface d'entrée et interface de sortie pour les deux modules. L'exemple de la *Figure 3-3* illustre que les évolutions dans A et dans B sont réciproquement dépendantes par la transition (temporisée ou instantanée) T_A/B. Le tir de la transition T_A/B ne s'effectue que lorsque les places P1_A et P1_B en amont de T_A/B sont marquées, et il entraîne le marquage des places P2_A et P2_B, c'est-à-dire la poursuite de l'évolution dans chaque module.

Nous conseillons que le franchissement de l'événement commun doit amener à un marquage tangible dans au moins un des deux modules afin d'éviter des conflits entre d'éventuelles transitions instantanées des deux modules. Dans l'exemple de la figure, cela signifie qu'au moins une des deux places P2_B et P2_A doit être suivie d'une transition temporisée. Il est également important que l'événement commun ne soit pas en concurrence avec d'autres événements dans le module A ou module B. Dans l'exemple de la figure, cela signifie que les jetons dans les places P1_A et P1_B peuvent être enlevés seulement par la transition T_A/B.

III.3.2.4 Règles et recommandations d'emploi de couplage

Lors de la construction d'un modèle global à partir de différents modèles de base, le modéliseur a souvent le choix entre les différentes façons de couplage présentées dans le paragraphe précédent. Ce choix étant souvent spécifique au système modélisé, il est difficile de formuler des règles précises concernant l'emploi de tel ou tel couplage. Néanmoins, sous certaines hypothèses, quelques règles et recommandations peuvent être dégagées.

Un point de vue important dans le choix du couplage est la sémantique "physique" du modèle considéré. Par exemple, lorsqu'un événement dans un système déclenche une action dans un autre système, le couplage par événement déclencheur est la façon la plus appropriée de modéliser cette relation. Au cas où l'on souhaite synchroniser deux évolutions indépendantes, le couplage par événement commun est employé, car il exprime fidèlement la réalité physique. Lorsque l'évolution d'un système se fait sous condition que d'autres systèmes soient dans un certain état, le couplage par place autorisante est le plus approprié.

Mis à part le point de vue "physique", des aspects relatifs au langage de modélisation (ici, les réseaux de Petri) doivent être pris en compte. Les couplages par événement déclencheur et par place autorisante ont la caractéristique principale de pouvoir distinguer clairement pour chaque module entre ses interfaces d'entrée et ses interfaces de sortie. Cela a l'avantage que chaque module puisse être conçu avec ses interfaces (notamment les interfaces d'entrée), le couplage avec les autres modules étant relativement faible. Par contre, un couplage par événement commun étant à la fois interface d'entrée et de sortie pour les modules concernés, il induit une dépendance plus forte entre les évolutions couplées. C'est pour cela que nous focalisons par la suite sur l'emploi de l'événement déclencheur ainsi que de la place autorisante (voir également *Figure 3-4* pour une illustration).

Dans la constellation 1 de la figure, plusieurs transitions (amenant à des états tangibles) d'un module A peuvent chacune déclencher la même action dans un module B. Dans ce cas, l'emploi du couplage par événements déclencheurs est équivalent au couplage par états autorisants. Néanmoins, lorsque chacun des événements amène à un état tangible différent, le couplage par place autorisante nécessite l'extension de la transition à déclencher du module B en une construction de type OU de plusieurs transitions. Le couplage par place autorisante nous semblant alors plus complexe à réaliser, nous recommandons par conséquent l'emploi du couplage par événement déclencheur.

La constellation 2 de la figure est un cas particulier de celui présenté précédemment. Ici, tous les événements déclencheurs amènent au même état tangible du module A. De manière analogue à la constellation 1, on peut choisir entre un couplage par événements déclencheurs ou par place autorisante. Dans une telle

constellation, le couplage par place autorisante permet de réduire le nombre de liens entre les modules et ainsi de diminuer la complexité du modèle global. Cela est surtout vrai lorsque plusieurs événements indépendants peuvent être déclenchés (ici, les transitions T1_B et T2_B). Par contre, lorsque seulement une de plusieurs transitions est franchissable (ici, une des transitions T1_B et T3_B), un couplage par événement déclencheur peut être avantageux, car la présence d'une seule place de réception est suffisante.

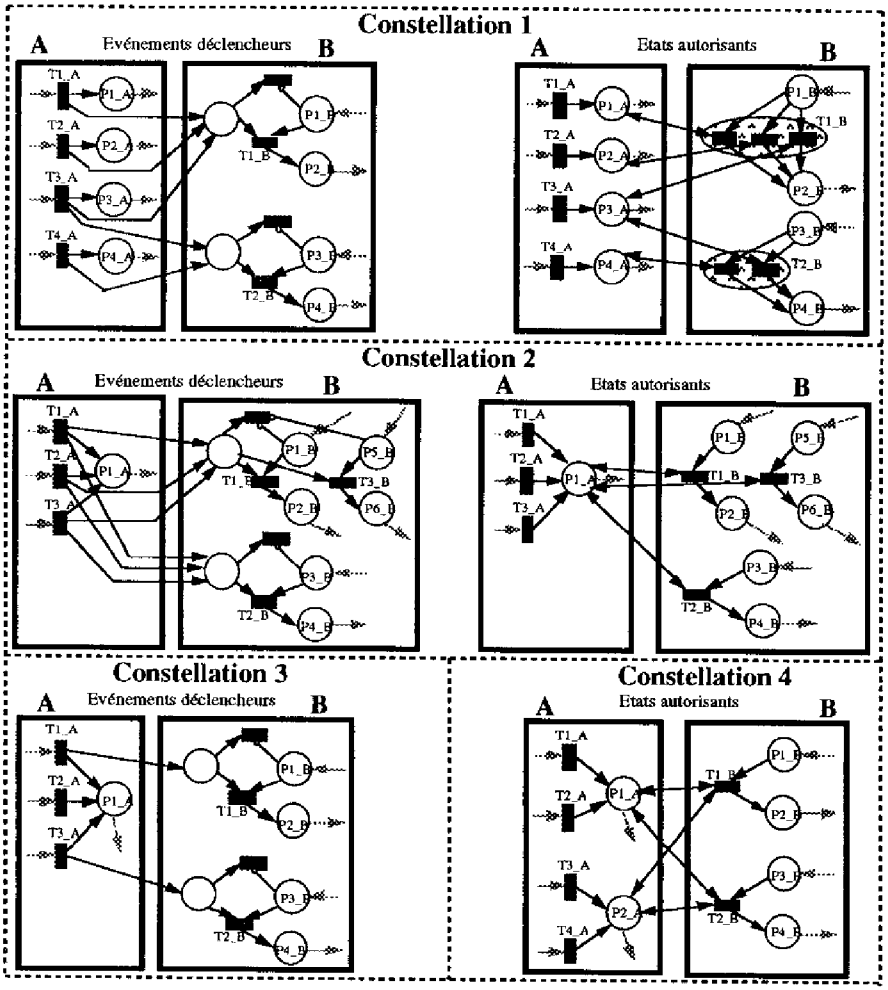


Figure 3-4 : Règles et recommandations d'emploi de couplage

La constellation 3 de la figure illustre le cas où plusieurs transitions amènent à un état tangible, mais seulement quelques-unes parmi elles correspondent à des

événements déclencheurs. Dans ce cas, l'occupation de l'état tangible n'est pas une information équivalente à l'occurrence des événements déclencheurs. Par conséquent, la seule façon de coupler les modules est l'événement déclencheur.

La constellation 4 da la figure illustre le cas où l'occupation simultanée de plusieurs états tangibles constitue la condition nécessaire et suffisante pour le franchissement d'une ou de plusieurs transitions. Cette construction de type ET est seulement faisable en employant le couplage par places autorisantes.

III.3.3 Validation des modèles de base

D'une manière générale, la validation d'un système vise à augmenter la confiance dans l'aptitude du système à délivrer un service conforme à l'accomplissement de sa fonction [Laprie *et al.* 1996]. La validation apparaît dans les activités d'élimination des fautes et de prévision des fautes ; l'élimination des fautes correspond à valider le système à un instant donné ("Le système est-il bon ?"), tandis que la prévision des fautes correspond à valider le système pour une durée donnée ("Pour combien de temps le système sera-t-il bon ?"). La modélisation du comportement d'un système par RdPSG et sa mise en œuvre à l'aide d'un outil tel que SURF-2 est une façon d'effectuer la prévision des fautes. Il est important aussi que l'on puisse avoir confiance dans la démarche de modélisation afin d'augmenter la confiance dans les résultats obtenus. C'est pour cela que toute conception d'un modèle RdPSG est obligatoirement accompagnée d'une validation spécifique (il s'agit d'une *validation de la validation*). La validation exhaustive d'un modèle global pouvant être trop difficile pour des raisons de complexité, la technique de modélisation modulaire a été développée en vue de pouvoir valider chaque sous-modèle indépendamment des autres.

Dans un premier temps, l'éditeur graphique de SURF-2 est un éditeur syntaxique qui permet, dès la saisie du modèle, de vérifier certaines propriétés relatives à la topologie du réseau. Par exemple, il est impossible de relier plusieurs places entre elles ou de connecter plusieurs transitions entre elles. Il est également interdit d'avoir des arcs redondants, et les sommets du graphe (c'est-à-dire, les places et les transitions) doivent avoir tous des noms différents. SURF-2 valide aussi la cohérence des structures de partitions et de revenus par rapport au réseau saisi.

Dans un deuxième temps, nous proposons de construire un simulateur d'environnement, afin de stimuler le fonctionnement du sous-modèle à travers son interface d'entrée. Cette stimulation doit permettre de calculer les invariants de marquage et de transitions, dont l'analyse et l'interprétation constituent une étape importante dans la validation du modèle (cf. paragraphe III.2.3.1). Un autre objectif du simulateur d'environnement est de déclencher toutes les transitions d'un module et de pouvoir ainsi vérifier visuellement le graphe de marquage et / ou la chaîne de

Markov associés. Lors de la construction de ce simulateur, les règles suivantes doivent être respectées :

1) Les événements déclencheurs des autres sous-modèles sont remplacés par des transitions temporisées générant des jetons à un certain taux.

2) Quant aux places autorisantes, on suppose que l'attente du sous-modèle pour le marquage des places autorisantes des autres sous-modèles puisse être simulée par une transition temporisée.

3) En ce qui concerne les événements en commun, il s'agit de simuler l'attente du sous-modèle de l'autorisation de tir venant des autres sous-modèles. C'est pour cela que les événements communs sont simplement représentés par des transitions temporisées.

Mis à part ces règles, il est courant que l'on inclue une connaissance approximative et partielle de l'évolution des sous-modèles couplés au sous-modèle à valider. Par exemple, les événements déclencheurs peuvent être inhibés par le marquage d'une certaine place, ou bien le franchissement de l'événement commun est conditionné par l'évolution d'un sous-modèle simplifié. Nous verrons un exemple d'un tel simulateur d'environnement dans le paragraphe III.4.1.2.

La méthode de validation proposée ici étant effectuée antérieurement au traitement des modèles, elle peut être complétée par une validation postérieure au traitement des modèles. Cette dernière consiste par exemple à vérifier les plages et les sens de variation des résultats numériques obtenus (nous reviendrons sur ce point dans le chapitre IV).

III.4 Exemples de modélisation

Dans ce paragraphe, nous présentons quelques exemples de modélisation appliqués à l'automobile en respectant les objectifs que nous nous sommes fixés et en suivant la méthode développée pour le couplage des modèles. Afin de comparer différentes architectures entre elles, nous avons choisi de modéliser deux fonctions différentes, chacune étant réalisée sur différentes architectures.

Concernant les **fonctions**, notre choix porte sur le coussin gonflable et la direction électronique.

- Le coussin gonflable est une fonction particulièrement intéressante du point de vue de la modélisation pour ses différents modes de défaillance, à savoir le non-gonflement lors de la sollicitation et le gonflement intempestif. De plus, le coussin gonflable est actuellement largement répandu sur les véhicules.
- La direction électronique est une fonction qui a un niveau de criticité très élevé et dont le fonctionnement continu doit être assuré (généralement, elle nécessite

donc une architecture à compensation d'erreur). Dans le futur, elle pourrait éventuellement remplacer la direction mécanique.

Les **architectures** choisies sont les plus courantes actuellement parmi les systèmes embarqués, à savoir le Simplex, le Duplex et le TMR.

- Lors de la modélisation de l'architecture Simplex, nous avons réalisé deux politiques de signalisation différentes. La politique 1 consiste à signaler au conducteur seulement les fautes permanentes (nous parlerons de façon simplifiée de "Simplex-1"), tandis que la politique 2 (appelée désormais "Simplex-2") correspond à signaler à la fois les fautes transitoires et permanentes. De plus, l'architecture Simplex simplifiée, où le recouvrement automatique est supposé instantané (dénommée "Simplex-i"), est modélisée.
- La modélisation de l'architecture Duplex tient compte de deux stratégies différentes de traitement d'erreur. La stratégie 1 (dénommée "Duplex-1") consiste à mettre la fonction en position sûre en cas de détection d'erreur par l'autotest local de chaque processeur ainsi que lors de la détection d'erreur par le comparateur. La stratégie 2 (dénommée "Duplex-2") permet de compenser l'erreur lorsque seulement un autotest local d'un processeur détecte une erreur, en permutant sur le processeur n'ayant pas détecté d'erreur.
- Concernant l'architecture TMR, on note que, à notre connaissance, elle n'a pas encore été implantée dans l'automobile. Elle est néanmoins intéressante à étudier, car elle pourrait accueillir des fonctions très critiques nécessitant une compensation d'erreur (par exemple, une future direction électronique). Dans les systèmes embarqués de l'avionique, cette architecture est très souvent utilisée à l'heure actuelle.

Le nombre de fonctions (2) et le nombre d'architectures (3+2+1) présentées ici permet en théorie d'étudier 12 modèles, dont les plus intéressants sont traités dans ce paragraphe. Nos évaluations portent essentiellement sur l'étude du coussin gonflable, dont la fonctionnalité et les mesures associées sont présentées dans le paragraphe III.4.1. Trois réseaux de base — à savoir le modèle fonctionnel du coussin gonflable, le modèle de l'activation du véhicule et le modèle du calculateur Simplex — sont présentés successivement dans les paragraphes III.4.2, III.4.3 et III.4.4. Le paragraphe III.4.5 explique ensuite comment construire différents réseaux globaux à partir de ces trois réseaux de base. Le paragraphe III.4.6 montre comment on peut modéliser l'architecture Duplex à partir d'un modèle Simplex et illustre son couplage avec le modèle fonctionnel du coussin gonflable et le modèle de l'activation du véhicule.

III.4.1 Le coussin gonflable et ses mesures

Pour le coussin gonflable, nous devons distinguer entre trois modes de défaillance :

1) Le gonflement du coussin sans avoir été sollicité (appelé gonflement intempestif).
2) La défaillance potentiellement catastrophique, où le coussin n'est pas prêt à réagir à une sollicitation éventuelle.
3) Le non-gonflement du coussin lors d'une sollicitation.

Compte tenu des différents modes de défaillance, nous nous intéressons à évaluer trois mesures de sûreté de fonctionnement :

1) La probabilité que le coussin se gonfle sans être sollicité (gonflement intempestif).
2) La probabilité conditionnelle de non-gonflement du coussin sachant qu'il est sollicité.
3) La probabilité d'une sollicitation du coussin non satisfaite.

L'introduction du coussin gonflable a pour but d'augmenter la sécurité des passagers et de diminuer leur risque de blessure, donc de diminuer le taux de défaillance catastrophique λ_c. Avec le coussin gonflable, λ_c peut être décomposé en deux composantes :

$\lambda_c = \lambda_{gi} + \lambda_{ng}$, avec :

λ_{gi} = le taux de défaillance dû au gonflement intempestif du coussin. Ce taux exprime le fait que la présence du coussin gonflable peut créer un risque supplémentaire.

λ_{ng} = taux de non-gonflement du coussin lors d'un accident. Ce taux peut s'exprimer en fonction du taux d'accident (noté λ_{acc}) et de la probabilité conditionnelle de gonflement du coussin lors de la sollicitation (notée θ) :

$\lambda_{ng} = (1-\theta)*\lambda_{acc}$

$\Rightarrow \lambda_c = \lambda_{gi} + (1-\theta)*\lambda_{acc}$

Lors de l'introduction du coussin gonflable, il faut respecter généralement deux contraintes :

1) Le risque ajouté par la présence du coussin (soit λ_{gi}) doit être très petit devant le risque sans coussin (soit λ_{acc}) :

$\lambda_{gi} \ll \lambda_{acc}$

2) Le taux de défaillance catastrophique avec le coussin (soit λ_c) doit être plus petit que le taux de défaillance catastrophique sans le coussin (soit λ_{acc}) :

$\lambda_c < \lambda_{acc}$, soit : $\lambda_{gi} + (1-\theta)*\lambda_{acc} < \lambda_{acc}$, soit : $\lambda_{gi} < \theta*\lambda_{acc}$

$\Rightarrow \theta > \lambda_{gi} / \lambda_{acc}$

On voit que si la contrainte (1) est respectée, la valeur minimale imposée sur θ par la contrainte (2) est très peu contraignante. Par conséquent, la contrainte (1) est la condition principale concernant l'introduction du coussin gonflable. Sachant que le taux de défaillance catastrophique influence directement le nombre de véhicules défaillants, nous allons étudier n_{ng}, n_{gi} et $n = n_{ng} + n_{gi}$, où n_{ng} et n_{gi} sont le nombre de véhicules accidentés (en *ppm*) avec respectivement non-gonflement du coussin ou gonflement intempestif. Nous vérifions également que n_{gi} soit très petit devant n_{acc}, le nombre de véhicules accidentés (en *ppm*) sans prendre en compte le coussin gonflable.

En prenant en compte la signalisation d'erreur, le taux de défaillance catastrophique peut être décomposé d'une autre façon en deux composantes :

$\lambda_c = \lambda_{fabr} + \lambda_{chauf}$, avec :

λ_{fabr} = le taux de défaillance exprimant le fait que la défaillance catastrophique est sous la responsabilité du fabricant.

λ_{chauf} = le taux de défaillance exprimant le fait que la défaillance catastrophique est sous la responsabilité du chauffeur.

Les défaillances catastrophiques qui sont sous la responsabilité du fabricant comprennent non seulement le gonflement intempestif, mais aussi une partie des cas de non-gonflement. En effet, parmi les erreurs qui causent un non-gonflement du coussin lors de la sollicitation, certaines peuvent être signalées suite à une détection. Nous introduisons le facteur β qui représente la proportion d'erreurs signalées. Nous définissons que les erreurs signalées sont sous la responsabilité du chauffeur, tandis que les erreurs non signalées sont sous la responsabilité du fabricant. Les équations suivantes peuvent alors être formulées :

$\lambda_{fabr} = \lambda_{gi} + (1-\beta)*\lambda_{ng}$

$\lambda_{chauf} = \beta*\lambda_{ng}$

De façon analogue au cas précédent, nous allons étudier λ_{fabr} et λ_{chauf} par le biais de leur impact sur le nombre de véhicules accidentés exprimé en *ppm*, soit n_{fabr}, n_{chauf} ainsi que $n = n_{fabr} + n_{chauf}$.

III.4.2 Modèle fonctionnel du coussin gonflable

Le modèle du coussin gonflable est développé dans l'objectif de décrire l'alternance entre les états de fonctionnement correct et incorrect, cette alternance étant gouvernée par les modèles du calculateur et de l'activation du véhicule. Notre objectif est de pouvoir définir les mesures de sûreté de fonctionnement sur les états du modèle fonctionnel et — éventuellement — du modèle de l'activation du véhicule, en faisant abstraction des autres modèles couplés. Dans les sous-paragraphes suivants, nous allons présenter successivement la construction ainsi que la validation du modèle fonctionnel du coussin gonflable.

III.4.2.1 Construction du modèle

La *Figure 3-5* illustre le modèle fonctionnel que nous avons développé pour le coussin gonflable. Le couplage des interfaces d'entrée avec les autres modèles de base y est illustré par des traits pointillés, les interfaces de sortie n'étant pas indiquées.

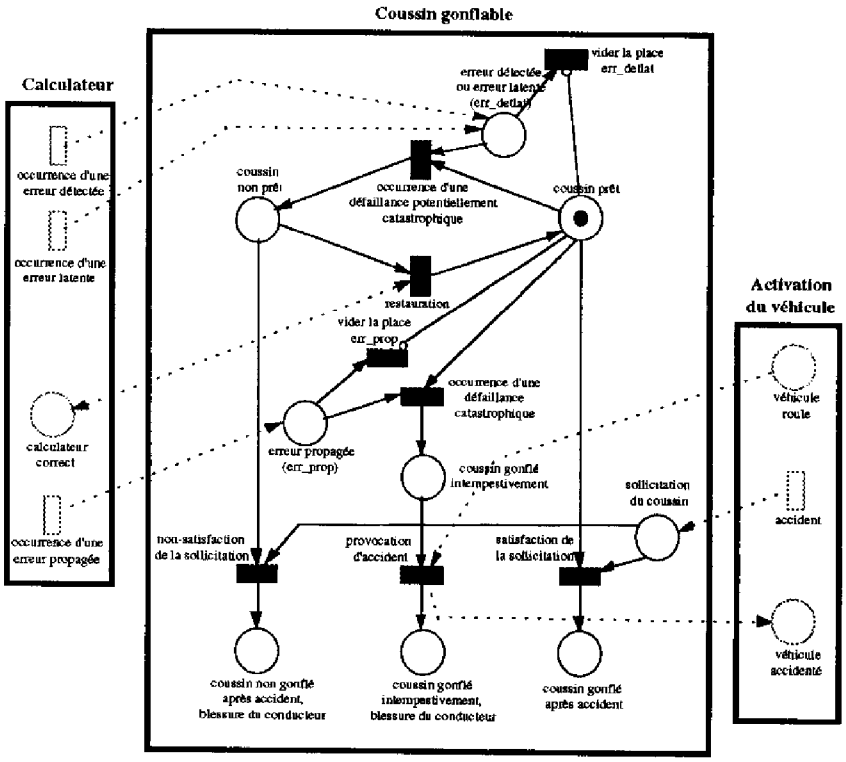


Figure 3-5 : Modèle fonctionnel du coussin gonflable

D'une manière générale, l'évolution du modèle du coussin gonflable est déterminée par des transitions instantanées qui sont conditionnées par des états du modèle du calculateur et du modèle de l'activation du véhicule.

A l'initialisation, un jeton se trouve dans la place "coussin prêt". A partir de cet état initial, nous avons représenté trois possibilités de déplacer le jeton :

1) Lors d'un accident, le coussin se gonfle tel que prévu (tir de la transition "satisfaction de la sollicitation"). Le jeton est mis dans la place absorbante "coussin gonflé après accident".

2) Une erreur se produit dans le calculateur et se propage de façon à provoquer immédiatement une défaillance catastrophique (franchissement de la transition "occurrence d'une défaillance catastrophique"), c'est-à-dire que le coussin se gonfle intempestivement. Un tel cas correspond à une provocation d'accident, c'est pour cela que le modèle du véhicule est également influencé par cet événement.

3) Une erreur se produit dans le calculateur et ne se propage pas (elle reste latente ou est détectée). Au niveau fonctionnel, cela signifie l'occurrence d'une défaillance potentiellement catastrophique, le coussin gonflable n'est plus prêt à se gonfler lors d'une sollicitation. A partir de l'état "coussin non prêt", il y a deux possibilités de déplacer le jeton :
   - Un accident a lieu avant que la fonction soit restaurée. Dans ce cas, le coussin ne se gonfle pas (tir de la transition "non-satisfaction de la sollicitation") et on suppose que le conducteur soit blessé.
   - L'état "calculateur correct" permet de restaurer le modèle fonctionnel à l'état "coussin prêt" avant que le coussin soit sollicité par un accident. Par exemple, la réparation ou la maintenance régulière du calculateur peuvent déclencher la restauration de la fonction.

Le modèle développé respecte la méthodologie de couplage que nous avons présentée dans le paragraphe III.3.2. Néanmoins, une exception est faite pour l'événement déclencheur "accident", pour lequel la place de réception associée ("sollicitation du coussin") ne nécessite pas d'être vidée. En effet, lors de la présence d'un jeton dans cette place, nous supposons que le jeton du modèle fonctionnel est dans une des places "coussin prêt" ou "coussin non prêt". Cela signifie que le jeton de la place "sollicitation du coussin" est consommé dans tous les cas de figure. Une fois que le jeton du modèle fonctionnel se trouve dans une des trois places absorbantes, il faut éviter l'arrivée de nouveaux jetons dans la place "sollicitation du coussin". Il faut alors s'assurer que l'événement "accident" ne puisse pas avoir lieu après une sollicitation du coussin ou une provocation d'accident, ce qui doit être garanti par la structure du modèle de l'activation du véhicule.

Dans l'objectif d'obtenir les mesures de sûreté de fonctionnement en utilisant l'outil SURF-2, nous devons définir l'ensemble des marquages impropres et des marquages catastrophiques. Dans le cas du coussin gonflable, seuls les marquages des places "coussin prêt" et "coussin gonflé après accident" représentent le fonctionnement correct. Les marquages de toutes les autres places forment l'ensemble des marquages impropres, c'est-à-dire le système est non disponible et non fiable lors de la présence d'un jeton dans une de ces places. Parmi ces marquages se trouvent les marquages de défaillance catastrophique, à savoir les marquages des places "coussin non gonflé après accident, blessure du conducteur (cnga_bc)" et "coussin gonflé intempestivement, blessure du conducteur (cgi_bc)". Lorsque le jeton se trouve dans une de ces places, le système est considéré non sûr. Cela signifie que la mesure n correspond aux défaillances associées au marquage des places cnga_bc ou cgi_bc. En respectant le partage proposé dans le paragraphe III.4.1, la mesure de n_{ng} correspond à la place cnga_bc, tandis que n_{gi} correspond à la place cgi_bc. La distinction entre les erreurs signalées et non-signalées ne peut être respectée qu'en incluant le modèle de l'activation du véhicule qui sera décrit ultérieurement, dans le paragraphe III.4.3. Quant à la mesure de *UAA*, nous devons diviser le temps moyen de marquage de la place "coussin non prêt" par la somme des temps moyens de marquage des places "coussin prêt" et "coussin non prêt".

III.4.2.2 Validation du modèle

La validation du modèle fonctionnel se fait en respectant les règles développées dans le paragraphe III.3.3. La *Figure 3-6* illustre la validation du modèle fonctionnel du coussin gonflable à l'aide d'un simulateur d'environnement. Afin de faciliter la lecture du modèle, nous avons remplacé les noms des places et des transitions par des abréviations.

Concernant le couplage au modèle du calculateur, les places de réception sont individuellement connectées à des transitions générant des jetons à un certain taux (par exemple, la place "err_prop" est liée au générateur de jetons "occ_err_prop"). La transition "rest" étant normalement couplée à une place autorisante, elle est remplacée ici par une transition temporisée. En ce qui concerne le couplage au modèle de l'activation du véhicule, nous avons opté pour un modèle simplifié. Ce modèle décrit l'évolution essentielle du modèle du véhicule par rapport au modèle fonctionnel, à savoir le passage de l'état de roulage à l'état d'accident. Une occurrence d'erreur ne pouvant pas avoir lieu après un accident du véhicule, les générateurs d'erreurs sont inhibés par l'état d'accident du véhicule.

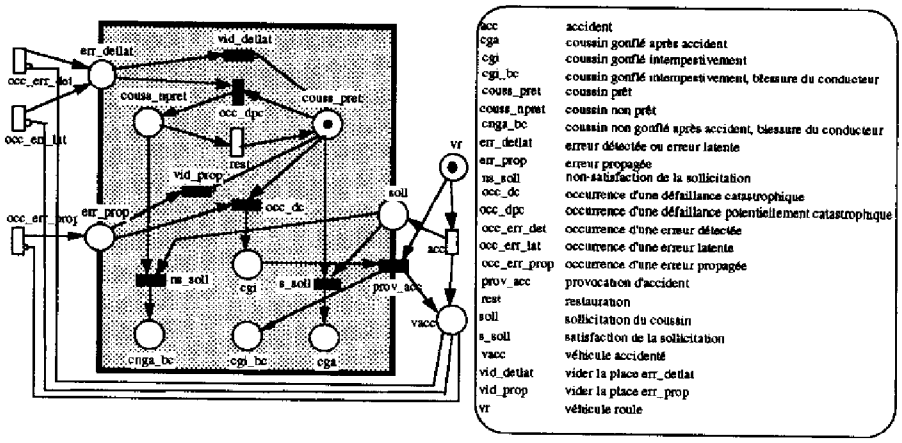


Figure 3-6 : Validation du modèle fonctionnel du coussin gonflable

La construction du simulateur d'environnement permet de calculer les invariants de marquage :

(1) $m(couss\_pret) + m(couss\_npret) + m(cnga\_bc) + m(cga) + m(cgi) + m(cgi\_bc) = 1$

(2) $m(vacc) + m(vr) = 1$

(3) $m(vr) + m(cnga\_bc) + m(soll) + m(cga) + m(cgi\_bc) = 1$

La première équation montre que le jeton du modèle fonctionnel est toujours dans une de ses propres places. La deuxième équation illustre que le jeton du modèle simplifié du véhicule se trouve soit dans "vr" soit dans "vacc". La troisième équation décrit la relation suivante : soit le véhicule roule, soit il y a une sollicitation du coussin qui est satisfaite ou non, soit il y a un accident dû à un gonflement intempestif. La présence de ces trois invariants nous conforte donc par rapport à la correction du modèle.

La *Figure 3-7* montre le graphe de marquage ainsi que la chaîne de Markov issus de la validation. Les états entourés par un cercle épais correspondent aux marquages tangibles, la première ligne présente la place où se trouve le jeton du modèle fonctionnel, la deuxième ligne indique si le véhicule roule ou s'il est accidenté, et la troisième ligne contient l'éventuel marquage des places de couplage. L'analyse du graphe de marquage permet de vérifier que toutes les transitions du modèle fonctionnel sont tirées. On s'assure notamment du fonctionnement correct des transitions "vid_detlat" et "vid_prop" qui, dans le cas des marquages M9 et M10, doivent vider les places "err_detlat" et "err_prop" respectivement et ainsi reboucler sur le marquage M6. Lors de la génération de la chaîne de Markov, les *rebouclages* (anglais : *self-loops*) M6-M9 et M6-M10 sont supprimés par l'outil SURF-2 (nous

les avons indiqués dans la figure afin de faciliter l'interprétation de la chaîne de Markov).

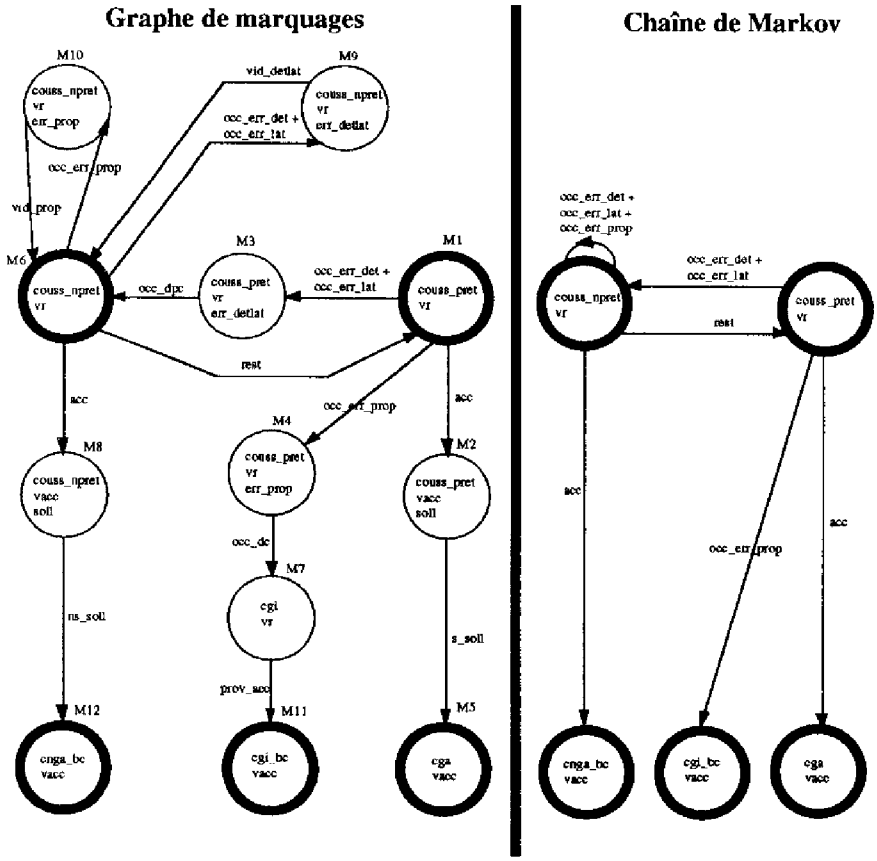


Figure 3-7: Graphe de marquages et chaîne de Markov pour le modèle fonctionnel du coussin gonflable

III.4.3 Modèle de l'activation du véhicule

Le modèle de l'activation du véhicule est un modèle de base qui est utilisé dans tous les modèles globaux sans que sa structure soit changée. Il décrit l'alternance et les relations temporelles entre les différents états tels que le roulage, l'arrêt normal et la réparation. Un point important de cette description est le choix d'une référence temporelle. En moyenne, l'automobile n'est en service que pendant 500 h par an, c'est-à-dire environ 5% du temps d'utilisation annuelle (1 an = 8760 h). Les temps d'arrêt normal ou de réparation sont très variables et par conséquent difficiles à moyenner. Nous faisons l'hypothèse simplificatrice qu'il n'y a pas de dégradation

des systèmes embarqués électroniques pendant les périodes de hors service. Par conséquent, nous choisissons les heures de roulage comme référence temporelle pour le modèle de l'activation du véhicule, les heures d'arrêt étant supposées de durée nulle.

Dans le premier sous-paragraphe, nous expliquons la construction du modèle, le deuxième paragraphe est dédié à la validation du modèle.

III.4.3.1 Construction du modèle

Le modèle de l'activation du véhicule (*Figure 3-8*) est composé de deux processus concurrents. Le **premier processus** (dénommé "Roulage") décrit l'alternance entre le roulage, la réparation, l'arrêt normal et l'accident du véhicule. Le **deuxième processus** (dénommé "Maintenance") décrit l'alternance entre le besoin ou non de maintenance régulière du véhicule. De manière analogue au modèle fonctionnel, seulement le couplage des interfaces d'entrée est illustré sur la figure (par des traits pointillés).

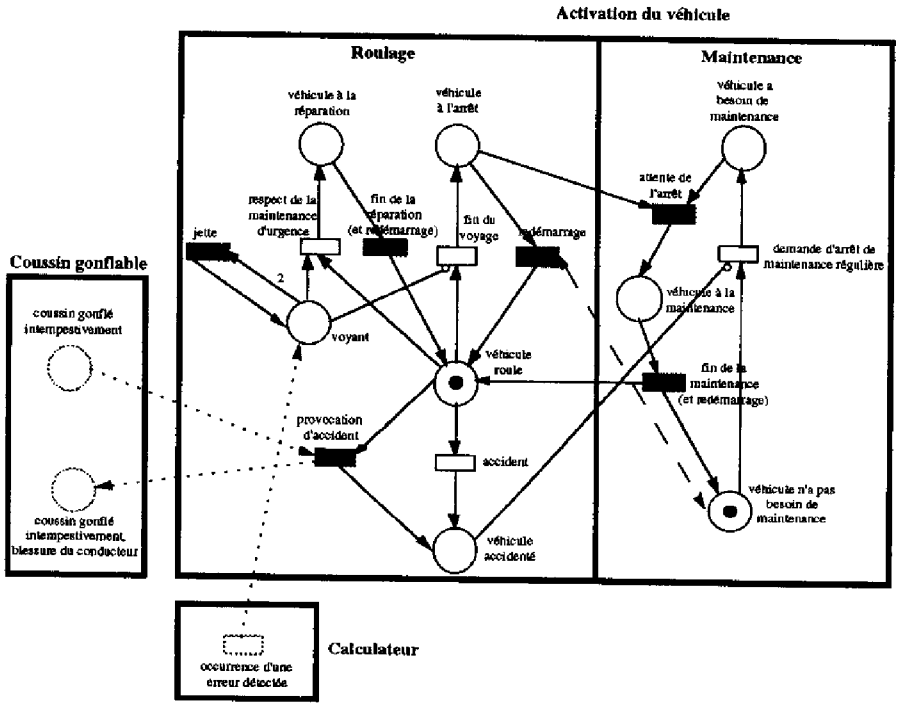


Figure 3-8 : Modèle de l'activation du véhicule

(1) Processus "Roulage" : à l'initialisation, le jeton se trouve dans la place "véhicule roule". L'état initial peut être quitté par quatre transitions :

- La transition temporisée fin du voyage représente l'arrêt régulier du véhicule. La place correspondante est quittée par le redémarrage qui peut avoir un effet positif (mais fortuit) sur l'état du calculateur. Cela est justifié par le fait que, lors d'un redémarrage, chaque calculateur effectue un autotest qui lui permet de détecter des erreurs préalablement latentes et de les recouvrir lorsqu'elles sont dues à des fautes transitoires. Le redémarrage du véhicule est synchronisé avec la place "véhicule n'a pas besoin de maintenance", c'est-à-dire que l'on ne peut pas redémarrer lorsque le véhicule est à la maintenance.
- La transition temporisée respect de la maintenance d'urgence est armée par l'allumage d'un voyant qui signale au conducteur l'occurrence d'une erreur détectée dans le calculateur. Le respect de l'arrêt d'urgence exclut le processus d'arrêt normal, c'est pour cela qu'il y a un arc inhibiteur entre "voyant" et "fin du voyage". Au cas où le voyant est déjà allumé lors d'une arrivée d'un jeton, ce deuxième jeton est enlevé (arc de poids 2 et transition "jette"). L'introduction d'un voyant enrichit la mesure de la sécurité par la notion de responsabilité : la signalisation d'erreur permet d'engager la responsabilité sur le conducteur qui est tenu à effectuer une maintenance d'urgence afin de faire réparer le système défaillant. Si un accident (sollicitant le coussin gonflable) a lieu avant que le conducteur ait fait réparer son véhicule, le non-gonflement du coussin (et la blessure du conducteur) lors de l'accident est sous la responsabilité du conducteur. Le fabricant est tenu de concevoir son système de telle façon que toutes les erreurs qui pourraient causer des défaillances catastrophiques soient signalées. Par conséquent, le non-gonflement du coussin (et la blessure du conducteur) lors d'un accident suite à une erreur non signalée est sous la responsabilité du fabricant. Un gonflement intempestif (et la blessure du conducteur) ne pouvant pas être signalé au conducteur, il est par définition sous la responsabilité du fabricant. La place "voyant" nous permet alors de distinguer entre les mesures n_{fabr} et n_{chauf} : n_{chauf} est la probabilité qu'il y ait un jeton dans la place "voyant" et dans la place "cnga_bc" du modèle fonctionnel, tandis que n_{fabr} est la probabilité que la place "voyant" soit vide et qu'une des places "cnga_bc" ou "cgi_bc" du modèle fonctionnel soit marquée.
- La transition temporisée accident amène à l'état "véhicule accidenté" et génère la sollicitation du coussin. Il s'agit d'un état absorbant (et par conséquent tangible), c'est-à-dire que les modèles n'évoluent plus après son marquage. C'est pour cette raison que la demande d'arrêt de maintenance régulière est inhibée par la place "véhicule accidenté".

- La transition instantanée provocation d'accident est un événement commun avec le modèle fonctionnel, elle enlève le jeton de "véhicule roule" et le met dans la place "véhicule accidenté".

Le choix de la référence temporelle étant les heures de roulage, le marquage des places "véhicule à la réparation" et "véhicule à l'arrêt" correspondent par conséquent à des états fugitifs qui sont quittés par les transitions instantanées "fin de la réparation" et "redémarrage". Ces transitions ont une conséquence sur l'évolution du modèle du calculateur, elles peuvent ramener son état à "calculateur correct" (on note que l'événement "fin de la réparation" signifie également le redémarrage du véhicule). Respectant la méthodologie de couplage, nous vérifions que ces deux événements amènent à la place stable "véhicule roule".

(2) Processus "Maintenance" : après la demande d'arrêt de maintenance régulière, on attend le prochain arrêt du véhicule pour effectuer la maintenance. La durée de la maintenance est égale à zéro à cause du choix de la référence temporelle ; par conséquent, l'état "véhicule à la maintenance" est quitté par la transition instantanée "fin de la maintenance". Cette transition remet le jeton dans la place "véhicule n'a pas besoin de maintenance" et remet également le modèle de roulage dans l'état tangible "véhicule roule" (on note que l'événement "fin de la maintenance" signifie également le redémarrage du véhicule). On suppose alors que la fin de la maintenance inclut le redémarrage du véhicule. Puisque les transitions "attente de l'arrêt" et "fin de la maintenance" sont instantanées, nous pouvons les fusionner en une seule transition appelée "fin de la maintenance", en faisant disparaître la place à marquage fugitif "véhicule à la maintenance".

III.4.3.2 Validation du modèle

L'évolution du modèle du véhicule est seulement influencée par l'événement commun "provocation d'accident" et par l'événement déclencheur "détection d'erreur". La validation respecte cette donnée et elle tient également compte du fait que l'accident inhibe l'occurrence d'autres erreurs (arc inhibiteur entre "vacc" et "occ_err_det"). La *Figure 3-9* illustre la validation, les noms des places et transitions étant remplacés par des abréviations pour une meilleure lisibilité.

Les invariants de marquage issus de la validation sont :

$$(1)\ m(vacc) + m(vr) + m(vrep) + m(var) = 1$$

$$(2)\ m(vnbm) + m(vbm) = 1$$

Ces équations soulignent l'existence de deux processus parallèles : le processus de l'activation du véhicule et le processus de maintenance.

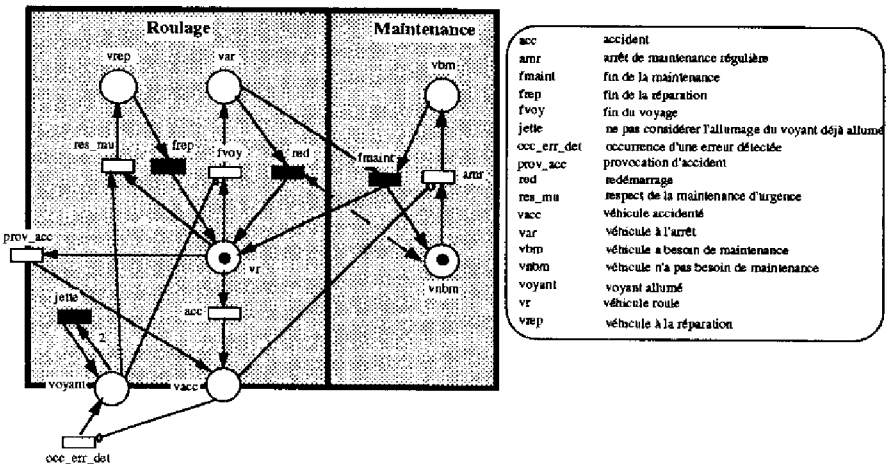


Figure 3-9 : Validation du modèle de l'activation du véhicule

Le graphe de marquage et la chaîne de Markov associés à la validation sont dessinés dans la *Figure 3-10*, les marquages tangibles étant indiqués par un cercle épais.

Le graphe de marquage présente une certaine symétrie par rapport à l'arrêt de maintenance régulière, les marquages M1, M3, M5, M9, M10 et M11 (véhicule n'a pas besoin de maintenance) sont reproduits dans M4, M7, M8, M12, M13 et M14 (véhicule a besoin de maintenance).

Le circuit M1-M2 correspond à l'utilisation normale du véhicule, le circuit M1-M4-M6 représente l'arrêt de maintenance régulière, les circuits M1-M5-M9 et M4-M8-M12 illustrent l'allumage du voyant suivi de la maintenance d'urgence et de la réparation. Les chemins M5-M10, M1-M3, M8-M13 et M4-M7 sont parcourus lors d'un accident ou d'une provocation d'accident ; ils mettent le modèle dans un état absorbant. Les circuits M5-M11 et M8-M14 résultent de la nécessité de la transition "jette" qui enlève d'éventuels jetons excédentaires dans "voyant".

Les circuits M1-M2, M5-M11 et M8-M14 correspondent à des événements qui ne font pas changer l'état du modèle, et dans le cas du circuit M1-M2, l'événement "red" peut affecter l'état d'autres modèles couplés.

III.4.4 Modèle du calculateur Simplex

Le modèle du calculateur est destiné à décrire les processus de création de faute ainsi que les procédures de restauration associés. Le déroulement de ces procédures dépend du caractère temporel des fautes ainsi que du type d'erreur associé.

Concernant le caractère temporel des fautes, nous distinguons entre fautes transitoires et fautes permanentes.

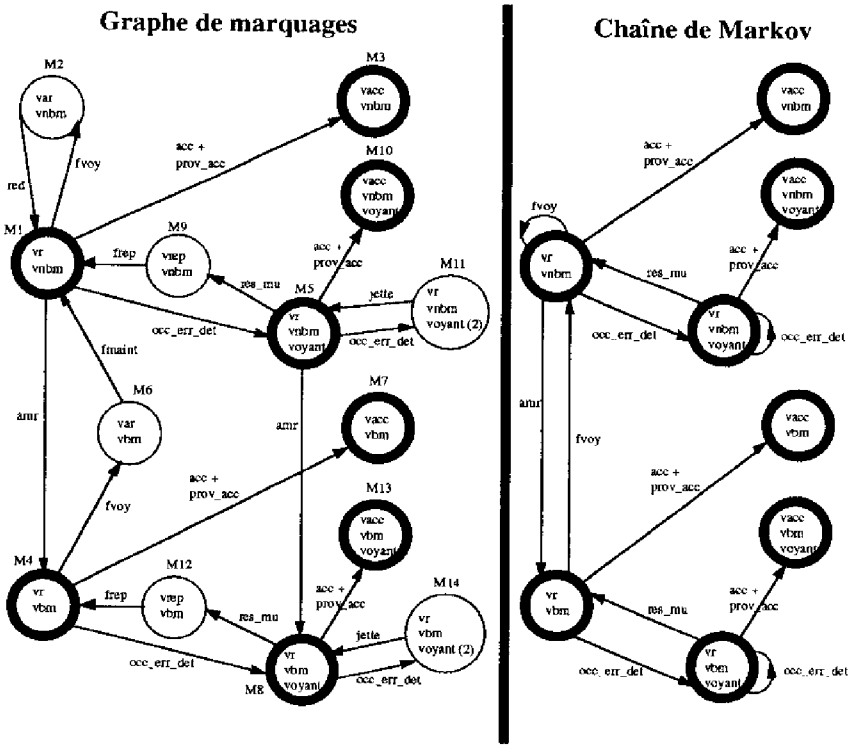


Figure 3-10 : Graphe de marquages et chaîne de Markov pour le modèle de l'activation du véhicule

Afin de pouvoir décrire et distinguer les différentes procédures de recouvrement d'erreur, nous introduisons la notion de *couverture* qui est une des caractéristiques quantitatives principales de la tolérance aux fautes. La couverture est classiquement définie comme la probabilité conditionnelle que, étant donné une faute dans le système, le système parvient à la tolérer. Elle permet alors de caractériser l'efficacité des différentes étapes du traitement d'erreur et de faute. La couverture imparfaite de la tolérance aux fautes résulte de deux grands types d'imperfections :

1) Les fautes affectant la conception et la mise en œuvre des mécanismes de tolérance aux fautes par rapport aux hypothèses de fautes considérées, ce qui se traduit par un manque de couverture des mécanismes de traitement des erreurs et des fautes.

2) La prise en compte d'hypothèses de fautes différentes de celles qui peuvent se produire en opération, ce qui résulte en un manquement de la couverture des hypothèses de fautes.

La modélisation des mécanismes de traitement d'erreur et de fautes peut être enrichie par la prise en compte des processus dynamiques de création et de détection d'erreurs. En effet, une faute dans le système peut rester dormante tant qu'elle n'est pas activée par un processus de traitement. L'activation de la faute conduit à une erreur qui peut rester latente tant qu'elle n'a pas été reconnue en tant que telle ; cette reconnaissance de l'erreur peut être soit la détection par un dispositif ou algorithme de détection, soit la propagation de l'erreur à l'interface du système résultant en une défaillance du système.

Les aspects dynamiques d'activation de faute et de reconnaissance d'erreur peuvent éventuellement être pris en compte par l'introduction de couvertures sous la forme de fonctions de répartition des variables aléatoires caractérisant les instants où se produisent effectivement les événements correspondants [Dugan & Trivedi 1989]. Ici, nous avons voulu tenir compte surtout des conséquences de ces imperfections dans les mécanismes de traitement d'erreur sans complexifier à outrance les modèles correspondants. Par conséquence, nous avons choisi de modéliser le processus de création d'erreurs de la façon suivante :

- Toute faute produit immédiatement une erreur (pas de fautes dormantes).
- L'erreur produite est :
  - soit détectée immédiatement ;
  - soit propagée immédiatement (conduisant à une défaillance du composant) ;
  - soit latente jusqu'à sa découverte éventuelle par une inspection de maintenance ou sa manifestation en tant que défaillance lors d'une sollicitation.

De manière analogue aux paragraphes III.4.1 et III.4.2, nous allons présenter tout d'abord le modèle pour ensuite parler de sa validation.

III.4.4.1 Construction du modèle

La prise en compte des différents phénomènes décrits précédemment induit un modèle du calculateur qui est relativement complexe. C'est pour cela que nous illustrons la construction du modèle sur un modèle simplifié (*Figure 3-11*).

A partir de l'état initial "calculateur correct", il y a l'occurrence d'une faute qui est soit permanente soit transitoire. Les arcs bidirectionnels entre "véhicule roule" et respectivement "faute permanente" et "faute transitoire" expriment la nécessité que le

véhicule roule (ce qui correspond à l'allumage des calculateurs) pour que les processus de faute du calculateur puissent avoir lieu.

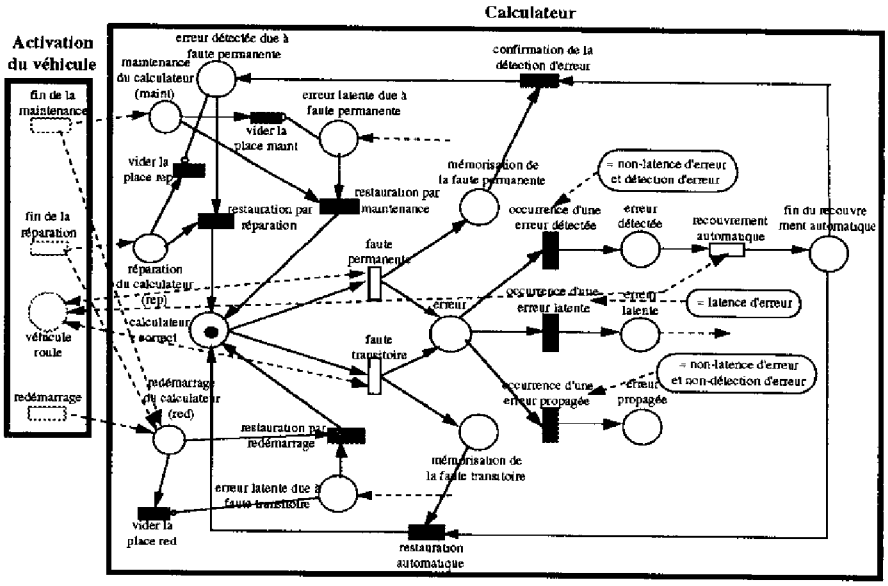


Figure 3-11 : Modèle simplifié d'un calculateur Simplex

On suppose que toute faute devienne active, le calculateur présentant ainsi une erreur. Pour des besoins de modélisation des mécanismes de restauration, on mémorise l'aspect temporel de la faute. Un sélecteur aléatoire permet ensuite de distinguer d'une façon instantanée entre les types d'erreur considérés, à savoir "erreur détectée", "erreur latente" ou "erreur propagée" :

a) Une erreur détectée est suivie par une tentative de recouvrement automatique. Le couplage de la transition correspondante avec la place autorisante "véhicule roule" exprime le fait que la tentative ne peut avoir lieu que lorsque le calculateur est allumé, c'est-à-dire lorsque le véhicule roule. S'il s'agit d'une faute transitoire, la tentative de recouvrement automatique se termine avec succès et le calculateur revient au marquage de la place "calculateur correct" en franchissant la transition "restauration automatique". En cas de faute permanente, on marque la place "erreur détectée due à faute permanente". Dans ce cas, le calculateur peut être remis à l'état correct par une action de réparation. On note que l'événement déclencheur "fin de la réparation" inclut l'événement de redémarrage, c'est pour cela qu'il est lié aux deux places de réception correspondantes du modèle calculateur.

b) Une erreur latente due à une faute permanente est supposée recouverte seulement par une action de maintenance. L'événement déclencheur "fin de la maintenance" inclut également l'événement "redémarrage", par conséquent il est lié aux deux places de lecture correspondantes. En fait, bien que les erreurs latentes dues à des fautes permanentes soient souvent détectées lors du prochain redémarrage, nous préférons ne pas modéliser cette possibilité afin de conserver l'hypothèse la plus pessimiste. Par contre, une erreur latente due à une faute transitoire est recouverte par le redémarrage.

c) Une erreur propagée ne peut pas être recouverte dans le cas de l'architecture Simplex. Elle entraîne immédiatement une défaillance catastrophique et amène à un état absorbant.

Le modèle complet du calculateur est dessiné sur la *Figure 3-12*.

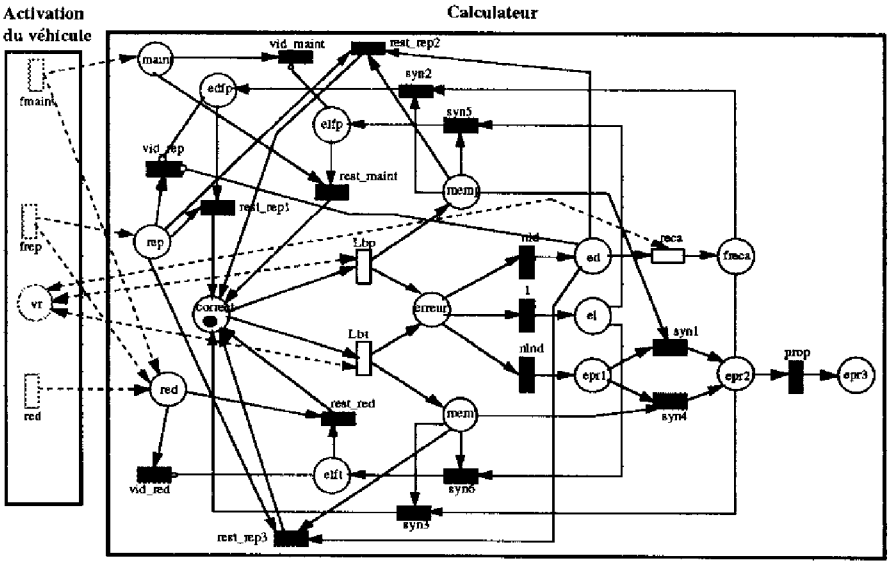


Figure 3-12 : Modèle complet d'un calculateur Simplex

III.4.4.2 Validation du modèle

Nous validons le modèle complet du calculateur Simplex comme indiqué dans la *Figure 3-13*, ce qui nous permet d'obtenir les invariants de marquage :

(1) $m(correct) + m(erreur) + m(ed) + m(el) + m(epr1) + m(edfp)$
$+ m(elft) + m(elfp) + m(freca) + m(epr3) + m(epr2) = 1$

(2) $m(correct) + m(edfp) + m(memp) + m(memt) + m(elft) + m(elfp) + m(epr3) + m(epr2) = 1$

$\Rightarrow$ (1) − (2) : $m(erreur) + m(ed) + m(el) + m(epr1) + m(freca) = m(memp) + m(memt)$

La première équation permet de vérifier que l'ensemble des places du modèle, sauf les places "memp" et "memt", contient au maximum un jeton à la fois. La soustraction (1)-(2) confirme ce fait, une présence d'un jeton dans "memp" ou "memt" induit une présence d'un deuxième jeton dans une des places "erreur", "ed", "el", "epr1" ou "freca".

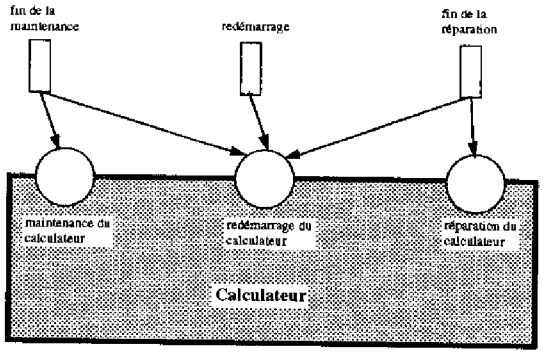


Figure 3-13 : Validation du modèle d'un calculateur Simplex

Le graphe de marquages issu de la validation contient 49 états et son analyse dépasserait le cadre de ce paragraphe (nous vérifions seulement que toutes les transitions du modèle sont tirées). Par conséquent, nous analysons ici seulement la chaîne de Markov (*Figure 3-14*) qui exhibe bien la sémantique souhaitée.

Uniquement l'état E1 correspond au fonctionnement correct du calculateur, les autres états représentent les différentes formes d'erreur ainsi que les procédures de recouvrement associées. Parmi les états de fonctionnement incorrect, quelques états méritent un commentaire à part. L'état E3 (erreur détectée, faute transitoire) peut être quitté par la procédure de recouvrement automatique ou par l'action de réparation qui sont deux processus concurrents. L'action de réparation n'est possible que lorsque l'on signale l'erreur (marquage de la place "voyant" du modèle de l'activation du véhicule). La signalisation de l'erreur (et la réparation) n'est pas obligatoire, car le recouvrement automatique suffit pour recouvrir l'erreur qui est due à une faute transitoire. La situation est différente dans le cas des états E2 et E6. Ici, l'erreur détectée est due à une faute permanente, seulement la réparation permet de recouvrir l'état erroné. Une signalisation d'erreur est alors obligatoire, elle peut être effectuée avant la tentative de recouvrement automatique (retour de l'état E2 vers E1) ou après l'échec de cette dernière (retour de l'état E6 vers E1).

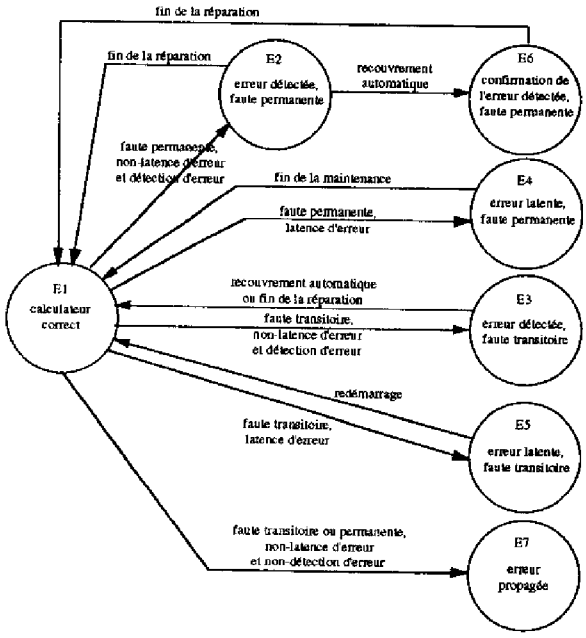


Figure 3-14 : Chaîne de Markov pour la validation du modèle d'un calculateur Simplex

III.4.5 Modèle global du coussin gonflable avec un calculateur Simplex

Le modèle global consiste à coupler le modèle fonctionnel du coussin gonflable et les modèles de l'activation du véhicule et du calculateur. Afin de faciliter la compréhension du modèle global et de montrer l'aspect modulaire de notre approche, nous ne décrivons que les liens entre les modèles de base. L'approche modulaire permet de concevoir aisément différents modèles globaux qui sont basés sur les mêmes modèles de base ; seules les interconnexions entre ces modèles de base changent. Dans le premier sous-paragraphe, nous allons illustrer la modélisation de deux politiques différentes de signalisation d'erreur. Le deuxième sous-paragraphe montre le réseau global lorsque la durée du recouvrement automatique est égale à zéro.

III.4.5.1 Deux politiques de signalisation d'erreur

La détection d'une erreur dans le calculateur a pour conséquence une défaillance potentiellement catastrophique au niveau fonctionnel et elle permet de tenter un recouvrement automatique à l'issue duquel on peut déterminer le caractère temporel

de l'erreur. Si la tentative échoue, on suppose que l'origine de l'erreur est une faute permanente. En cas de succès, on suppose que l'erreur était due à une faute transitoire. On peut imaginer deux politiques différentes pour signaler l'erreur détectée. La première politique consiste à signaler uniquement les erreurs détectées qui persistent après la tentative de recouvrement automatique, tandis que la deuxième consiste à signaler toute erreur détectée avant de tenter le recouvrement automatique. Le modèle global est illustré dans la *Figure 3-15*, les liens spécifiques aux deux politiques de signalisation sont étiquetés avec des numéros correspondants, les autres liens ne changent pas d'une politique à l'autre.

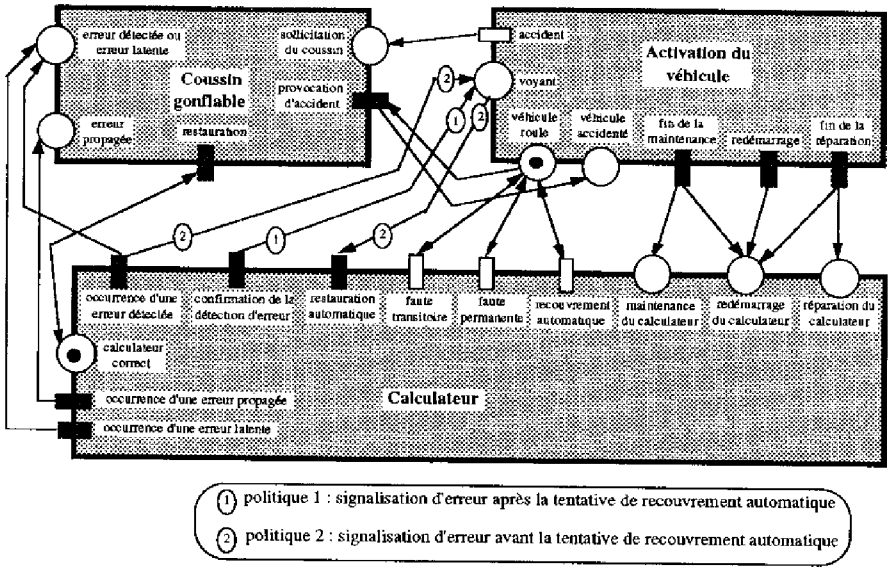


Figure 3-15 : Modèle global, deux politiques différentes de signalisation

Lors de l'application de la **politique 1**, on sait que l'allumage du voyant correspond à la détection d'une erreur qui est due à une faute permanente, le lien entre "confirmation de la détection d'erreur" et "voyant" illustre ce fait. Seulement une réparation permet de recouvrir l'erreur, le voyant est alors éteint par le respect de la maintenance d'urgence (événement propre au modèle d'activation, cf. *Figure 3-8*).

En cas d'utilisation de la **politique 2**, l'allumage du voyant est causé par la détection d'une erreur qui peut être due à une faute permanente ou à une faute transitoire (lien entre "occurrence d'une erreur détectée" et "voyant"). En cas de faute transitoire, le voyant peut être éteint par l'action de restauration automatique du calculateur (lien entre "voyant" et "restauration automatique") ou par le respect de la maintenance d'urgence. En cas de faute permanente, seulement la réparation permet de recouvrir l'erreur.

Le calcul des invariants de marquage du modèle global fournit les équations suivantes :

(1) $m(vacc) + m(vr) + m(vrep) + m(var) = 1$

(2) $m(vnbm) + m(vbm) = 1$

(3) $m(correct) + m(edfp) + m(memp) + m(memt) + m(elft) + m(elfp) + m(epr3) + m(epr2) = 1$

(4) $m(correct) + m(erreur) + m(ed) + m(el) + m(epr1)$
$+ m(edfp) + m(elft) + m(elfp) + m(freca) + m(epr3) + m(epr2) = 1$

(5) $m(couss\_pret) + m(couss\_npret) + m(cnga\_bc) + m(cga) + m(cgi) + m(cgi\_bc) = 1$

(6) $m(vr) + m(cnga\_bc) + m(soll) + m(cga) + m(cgi\_bc) + m(var) + m(vrep) = 1$

L'étude des invariants de marquage nous confirme que les trois modèles de base sont indépendants. On vérifie également que les invariants de marquage sont les mêmes pour les deux politiques de signalisation d'erreur, ce qui souligne la modularité de notre approche.

La chaîne de Markov associée au réseau global est illustrée dans la *Figure 3-16*, où l'on distingue également entre les deux politiques de signalisation d'erreur par un étiquetage correspondant (les états et les arcs non étiquetés avec le numéro de la politique sont valables pour les deux politiques).

L'analyse de la chaîne illustre la symétrie déjà révélée dans la chaîne de Markov associée à la validation du modèle de l'activation du véhicule. Le passage de la moitié supérieure à la moitié inférieure correspond à la demande d'arrêt de maintenance régulière. L'analyse de la chaîne de Markov montre également les différents processus de défaillance et de recouvrement. Les états E1, E2, E3 et E6 correspondent au fonctionnement correct du système, dans les 22 autres états le système est défaillant. Parmi les états de fonctionnement correct, les deux états absorbants E3 et E6 représentent le gonflement correct du coussin. Parmi les états de fonctionnement incorrect, on identifie 12 états absorbants de défaillance catastrophique, où le fonctionnement incorrect du coussin (le coussin ne s'est pas gonflé lors de l'accident ou il s'est gonflé intempestivement) a provoqué la blessure du conducteur. Ces états représentent alors les conséquences des défaillances potentiellement catastrophiques, ils contribuent à la non-sécurité du système.

Pour la **politique 1**, on vérifie que seulement l'échec du recouvrement automatique est signalé au conducteur. Ainsi, deux défaillances catastrophiques sont sous la responsabilité du chauffeur (états E25 et E26), les dix autres défaillances catastrophiques sont sous la responsabilité du fabricant. Pour la **politique 2**, on vérifie que le voyant est allumé avant la tentative de recouvrement automatique, c'est-à-dire dans les états E4, E5, E7 et E9. Par conséquent, ces états peuvent être également quittés par une réparation et la proportion des défaillances catastrophiques sous responsabilité du chauffeur est plus importante que dans la politique 1. Ici, six

défaillances catastrophiques sont sous la responsabilité du chauffeur, les six autres défaillances catastrophiques sont sous la responsabilité du fabricant.

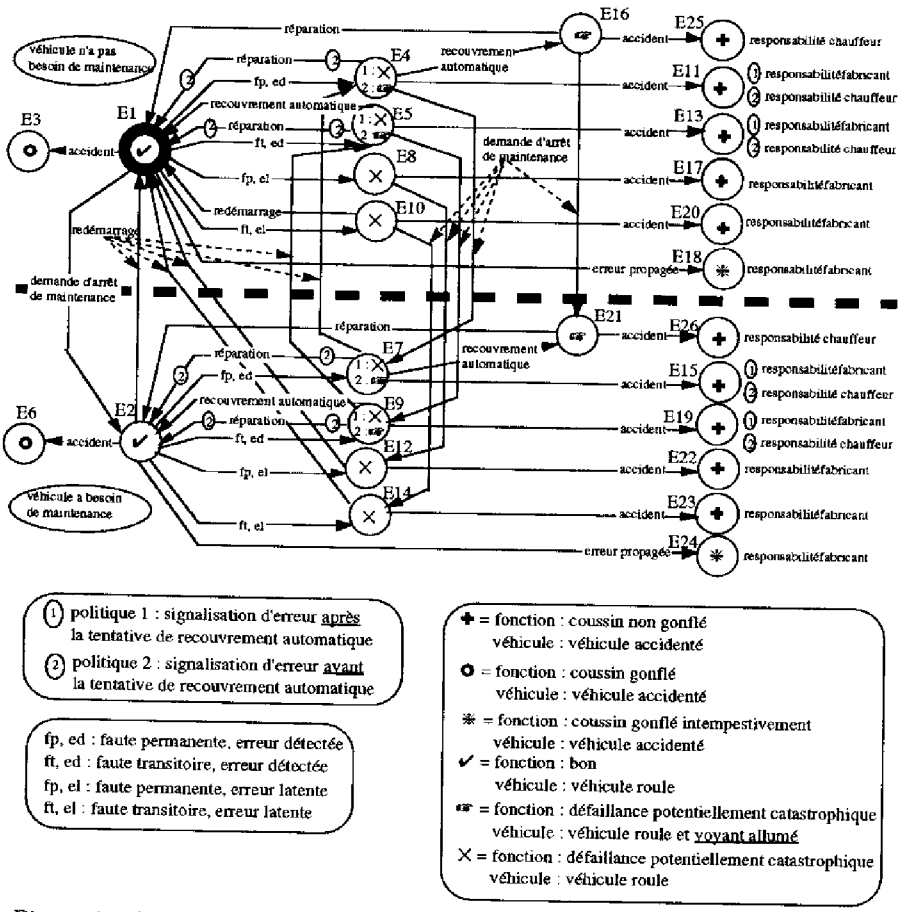


Figure 3-16 : Chaîne de Markov du modèle global, deux politiques différentes de signalisation

III.4.5.2 Recouvrement automatique instantané

La durée du recouvrement automatique est en général inférieure à une seconde ce qui revient à dire que le taux de transition est supérieur à 3600/h. Ce taux est très élevé par rapport au taux d'occurrence de faute qui se situe aux alentours de 10^{-3}/h. Par conséquent, nous allons représenter le recouvrement automatique par une transition instantanée (c'est le seul changement à effectuer dans le modèle du calculateur, les autres modèles de base restent inchangés). La *Figure 3-17* illustre le

modèle global correspondant. Le recouvrement n'est plus un processus de durée tangible, c'est pour cela que le couplage avec la place autorisante "véhicule roule" est supprimé. De plus, une distinction entre deux politiques de signalisation n'étant plus possible, nous avons choisi un couplage correspondant à celui de la politique 1.

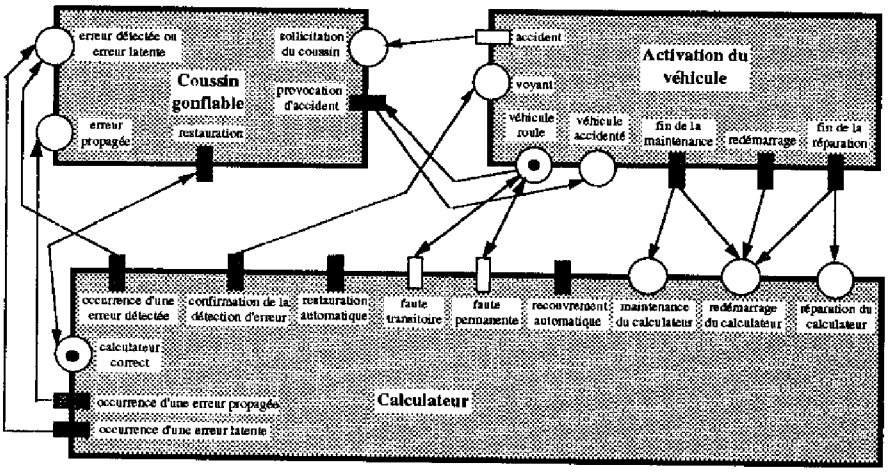


Figure 3-17 : Modèle global, recouvrement automatique instantané

La chaîne de Markov (*Figure 3-18*) illustre le fait que les états suivis par un recouvrement automatique sont devenus des états fugitifs et n'apparaissent plus dans la chaîne de Markov.

III.4.6 Modèle global du coussin gonflable avec un calculateur Duplex

Dans le cas d'une architecture Duplex (illustrée schématiquement dans la *Figure 3-19*), il y a deux calculateurs de type Simplex qui exécutent le même logiciel de façon synchrone. Chacun des deux calculateurs peut détecter certaines erreurs localement par l'intermédiaire de son autotest. La détection d'une éventuelle erreur par l'autotest déclenche une tentative de recouvrement locale qui se termine avec succès en cas de faute transitoire. En cas de faute permanente, on peut envisager deux stratégies :

1) Pour des fonctions de sécurité (par exemple, le coussin gonflable), on signale l'erreur locale au conducteur (peu importe l'état de l'autre calculateur) et on coupe le service global (la fonction ne peut plus être assurée). La réparation ramène le calculateur dans l'état correct.

2) Pour des fonctions de fiabilité (par exemple, la direction électrique), on commute directement sur l'autre calculateur si celui-là ne signale pas d'erreur

locale (suite à son propre autotest). Cela signifie que l'on tolère l'erreur et que la fonction est assurée. On signale l'erreur locale au conducteur et on suppose que le calculateur est ramené dans l'état correct par la réparation.

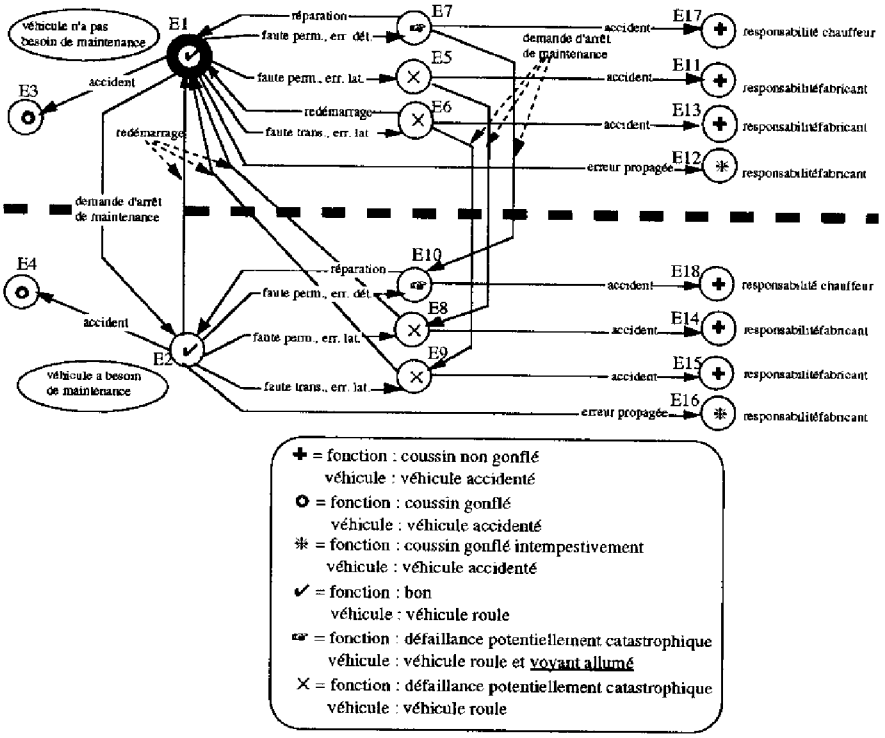


Figure 3-18 : Chaîne de Markov du modèle complet, recouvrement automatique instantané

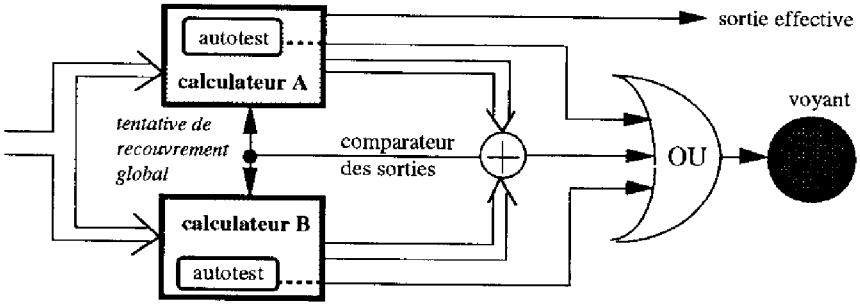


Figure 3-19 : Principe de l'architecture Duplex

Lorsque les deux autotests ne signalent pas d'erreur, on compare les sorties des deux calculateurs afin de détecter des erreurs qui se sont propagées vers les sorties sans avoir été détectées auparavant par l'autotest. La comparaison est effectuée non seulement sur les sorties effectives, mais aussi sur les sorties de contexte. Cela permet de détecter — en plus des erreurs propagées — des erreurs latentes. Cette détection n'est pas parfaite, certaines erreurs échappent au comparateur et peuvent entraîner une défaillance catastrophique du système. Au cas où le comparateur détecte une différence entre les sorties comparées, on ne peut pas localiser le calculateur défaillant. On doit donc tenter un recouvrement global des deux calculateurs qui se termine avec succès en cas de faute transitoire. En cas de non-succès de cette tentative, on sait que l'erreur est due à une faute permanente et on signale l'erreur au conducteur.

Le modèle des deux calculateurs est basé sur le modèle du Simplex. Le processus de recouvrement automatique local est représenté par une transition instantanée. Le modèle du calculateur est complexifié par la possibilité du recouvrement global qui permet, d'une part, de ramener le calculateur à l'état correct après une erreur latente due à une faute transitoire, et, d'autre part de signaler une erreur propagée due à une faute permanente. De façon analogue au raisonnement mené lors de la construction du modèle Simplex, nous représentons le processus de recouvrement global par une transition instantanée. La chaîne de Markov associée au réseau global comporte 176 états et est alors trop complexe pour être validée par inspection. Les validations du modèle fonctionnel et du modèle de l'activation du véhicule étant déjà effectuées auparavant, nous nous limitons à la validation du modèle du calculateur.

Le réseau global de l'architecture Duplex — couplée avec les modèles du coussin gonflable et de l'activation du véhicule — est illustré dans la *Figure 3-20*. Les liens spécifiques aux deux stratégies sont étiquetés avec des numéros correspondants, les autres liens ne changent pas d'une stratégie à l'autre, et les arcs bidirectionnels sont distingués par des traits pointillés. Le comparateur est représenté par un facteur de couverture de détection, la durée de comparaison est égale à zéro.

Les aspects suivants méritent d'être soulignés :

a) Lors d'une erreur propagée (à la sortie effective) dans un des calculateurs, nous ne distinguons pas entre faute transitoire et faute permanente. En cas de non-détection de l'erreur propagée par le comparateur, nous supposons que la conséquence est une défaillance intempestive (erreur propagée sur le modèle fonctionnel). Si le comparateur détecte l'erreur propagée, le voyant est allumé et seulement une réparation peut ramener les calculateurs à l'état correct.

b) Lors d'une erreur latente dans un des calculateurs, nous effectuons également une comparaison. Cette comparaison concerne les sorties de contexte.

En cas de détection, on donne l'ordre de recouvrement global aux deux calculateurs. Deux situations peuvent alors se produire : **1)** L'erreur latente est due à une faute permanente, ce qui signifie l'échec de la tentative de recouvrement global. Cette erreur doit être signalée et une réparation permet de remettre les calculateurs dans l'état correct. **2)** L'erreur latente est due à une faute transitoire et est recouverte par le recouvrement global. La durée de ce recouvrement étant égale à zéro, le modèle fonctionnel peut rester dans l'état "bon" et les calculateurs se retrouvent dans l'état correct.
En cas de non-détection, on ne peut pas effectuer le recouvrement global et au niveau fonctionnel on marque la place de réception "erreur détectée ou erreur latente" (ce qui amène le modèle fonctionnel dans l'état de défaillance potentiellement catastrophique). Les erreurs latentes sont recouvertes soit par un redémarrage du système (en cas de faute transitoire) soit par la prochaine maintenance (en cas de faute permanente).

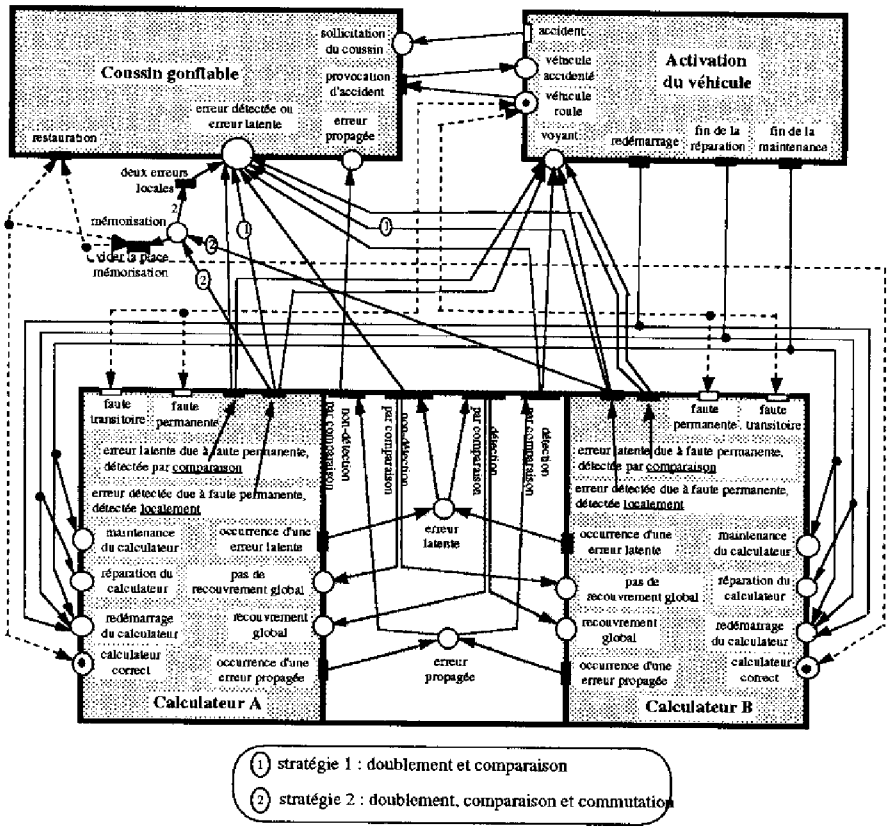


Figure 3-20 : Modèle global de l'architecture Duplex

c) La conséquence de la détection locale d'une erreur qui est due à une faute permanente dépend de la stratégie Duplex adaptée. Avec la stratégie 1, le modèle fonctionnel est mis dans l'état de défaillance potentiellement catastrophique — on considère que la fonction ne peut plus être assurée. La stratégie 2, par contre, consiste à assurer que la fonction reste dans l'état "bon" — on suppose que la fonction puisse être assurée par le calculateur non-défaillant. Au cas où celui-ci défaille également, le modèle fonctionnel est mis dans l'état de défaillance potentiellement catastrophique. Dans les deux stratégies, on allume toujours le voyant, ce qui permet de réparer les calculateurs. Cette réparation concerne non seulement le calculateur ayant affiché la défaillance, elle permet également de recouvrir des éventuelles erreurs latentes dans l'autre calculateur.

d) La détection locale d'une erreur qui est due à une faute transitoire n'est pas visible à l'interface du calculateur, car le recouvrement local permet de ramener instantanément le calculateur dans l'état correct.

Conclusion

Dans ce chapitre, nous avons présenté différentes techniques d'évaluation, parmi lesquelles nous avons ciblé sur l'évaluation probabiliste par Chaînes de Markov, en passant par une modélisation modulaire en Réseaux de Petri Stochastiques Généralisés. La méthode de modélisation développée permet de générer une multitude de modèles très complèts à partir de peu de modèles relativement simples, en respectant quelques règles de construction. Il s'agit d'une méthode très générale qui pourrait être appliquée à différents domaines où l'évaluation de la sûreté de fonctionnement rentre en jeu. L'illustration de la méthode à travers des exemples du domaine d'automobile nous a permis de démontrer son intérêt : elle permet de maîtriser aisément la complexité des modèles globaux et facilite leur compréhension. Nous avons choisi de développer trois types de modèles de base : le modèle fonctionnel, le modèle de l'activation du véhicule et le modèle du calculateur. A partir de ces modèles de base, il est en particulier possible de lier un modèle fonctionnel avec différents modèles d'architecture et le modèle de l'activation du véhicule.

Dans le cadre de nos travaux, la modélisation a pour but d'évaluer et de comparer la sûreté de fonctionnement de différentes architectures. Le chapitre IV reprend les modèles développés ici afin de quantifier, interpréter et comparer les mesures de la sûreté de fonctionnement. Il importe de dire ici que n'importe quelle modélisation repose sur des hypothèses souvent simplificatrices et ne peut décrire qu'une partie ou un aspect particulier d'un système. Par conséquent, les valeurs numériques issues de la modélisation doivent être interprétées avec prudence, elles dévient souvent de la réalité physique. Par contre, en appliquant systématiquement les mêmes hypothèses

et démarches de modélisation à un ensemble de systèmes, nous pouvons donner des conclusions issues de la comparaison des chiffres obtenus.

Chapitre IV

Analyse des résultats

Dans ce chapitre, nous présentons et analysons les résultats numériques obtenus pour les différentes mesures de sûreté de fonctionnement définies au chapitre III. L'objectif de ce chapitre est double :

- de pouvoir mettre ces résultats en relation avec des objectifs quantitatifs que l'on se fixe généralement lors de la réalisation d'un système réel,
- de comparer plusieurs architectures au niveau de la sûreté de fonctionnement.

Comme déjà indiqué au paragraphe III.2.4.2, les objectifs quantitatifs de sûreté de fonctionnement dans l'industrie automobile sont souvent exprimés en termes de n, c'est-à-dire le nombre de systèmes défaillants par million de systèmes pendant un an ; en général, les constructeurs souhaitent que n soit inférieur à 1 *ppm*. La spécification des objectifs de sûreté de fonctionnement dans le domaine de l'électronique automobile est également traitée dans [Kopetz 1995] qui formule les objectifs en termes de taux de défaillance : sachant qu'un véhicule automobile roule environ 500h par an, l'objectif n = 1 *ppm* reviendrait à fixer l'objectif du taux de défaillance d'un système donné à $2*10^{-9}$/h. Compte tenu de cela, nous proposons également de fixer l'objectif pour n à 1 *ppm*. Notons que cet objectif ne représente qu'un simple ordre de grandeur, car la modélisation markovienne des systèmes analysés repose sur des hypothèses nécessairement simplificatrices. En ce qui concerne la mesure de l'indisponibilité avant accident *UAA*, nous rappelons qu'elle n'est pas encore utilisée dans le domaine de l'automobile. Par conséquent, il est difficile de lui fixer un objectif quantitatif. Dans la pratique, la mesure de *UAA* pourrait consister à surveiller en continuité un système, de mesurer le temps pendant lequel le système est incapable de réagir à une sollicitation éventuelle et de calculer le rapport entre ce temps mesuré et le temps total de fonctionnement. Il est évident que ce rapport doit être le plus petit possible, et nous proposons par exemple un maximum de 10^{-4} ce qui correspond à 0,05 heures ou 3 minutes d'indisponibilité par an, compte tenu du roulage annuel de 500 heures.

Le traitement des modèles avec un outil informatique tel que SURF-2 ne peut être effectué qu'après avoir affecté les différents paramètres de valeurs numériques. Cet aspect est traité dans le paragraphe IV.1. Nous présentons ensuite dans le paragraphe

IV.2 les résultats obtenus pour différentes architectures du système de coussin gonflable. L'étude des différentes architectures pour la direction électronique est traitée dans le paragraphe IV.3.

IV.1 Choix des valeurs numériques pour les paramètres

Le choix des ordres de grandeur des valeurs numériques pour les paramètres des modèles étudiés doit être fait de telle sorte que le modèle représente au mieux le comportement du système réel. C'est pour cela que nous attribuons des valeurs nominales aux différents paramètres. Certains de ces paramètres sont modifiés dans les paragraphes suivants afin d'effectuer des études paramétriques. Les valeurs nominales des paramètres sont valables pour toutes les architectures et toutes les fonctions. Par exemple, le choix des valeurs numériques des paramètres du Duplex se fait en se basant sur l'hypothèse simplificatrice que l'architecture Duplex est composée de deux calculateurs Simplex et d'un comparateur. Lorsque la durée d'une procédure (par exemple, la durée du recouvrement automatique) est déterministe, on aurait pu faire appel aux Réseaux de Petri Stochastiques et Déterministes qui permettraient de modéliser également des délais constants. Ou bien, on pourrait utiliser les Réseaux de Petri Stochastiques Etendus qui permettraient d'approcher la durée déterministe par une distribution uniformément répartie entre deux valeurs limites ou par une distribution gaussienne. Cependant, nous choisissons de nous limiter à l'utilisation de distributions exponentielles à taux constant. Ce choix est motivé par la facilité de traitement de modèles purement markoviens. Par ailleurs, comme cela est montré dans [Laprie 1975], l'approximation d'autres distributions par une distribution exponentielle reste souvent valable dans les modèles de sûreté de fonctionnement, même dans le cas d'une durée déterministe (qui peut être estimée dans ce cas par l'inverse du taux de la distribution exponentielle).

Nous passons maintenant en revue les paramètres qui interviennent dans nos modèles afin de leurs attribuer des valeurs nominales et des plages de variation.

- Le taux de faute permanente d'un calculateur, λ_p : on admet qu'une faute permanente puisse se produire environ toutes les 100 000 heures de fonctionnement, la valeur nominale du taux λ_p est alors égale à 10^{-5}/h. La gamme des valeurs étudiées est 10^{-7}/h, 10^{-6}/h, 10^{-5}/h et 10^{-4}/h.
- Le taux de faute transitoire d'un calculateur, λ_t : compte tenu des interférences électromagnétiques, ce taux peut être très élevé. Nous supposons que la valeur nominale de λ_t est 10 fois plus grande que λ_p, soit 10^{-4}/h. Afin d'étudier la sensibilité par rapport à λ_t, les variations autour de la valeur nominale sont 10^{-6}/h, 10^{-5}/h, 10^{-4}/h et 10^{-3}/h.

- La durée moyenne du recouvrement automatique, τ_{reca} : nous supposons que la valeur nominale de τ_{reca} est de quelques centaines de millisecondes. Nous prenons une valeur de 0,36 secondes, ce qui correspond à un taux de transition de 10^4/h. L'influence de τ_{reca} sera étudiée pour les valeurs 0h, 10^{-4}h et 10^{-2}h, la première valeur correspondant à représenter τ_{reca} par une transition instantanée.
- La proportion d'erreurs latentes, l : parmi les fautes créées, certaines donnent lieu à des erreurs qui se propagent immédiatement, d'autres conduisent à des erreurs qui restent latentes. Nous supposons que seulement une petite part des fautes provoquent des erreurs latentes, la valeur nominale de l pouvant être fixée à 0,05. Nous étudions les différents modèles pour la gamme l = 0,0005 - 0,005 - 0,05 - 0,5.
- La couverture d'autodétection d'erreur, d : un calculateur dispose d'une autodétection qui permet de détecter la plus grande part des erreurs (c'est-à-dire, les fautes non latentes). Le paramètre d est fortement lié au calculateur et au logiciel d'autodétection. Nous supposons que sa valeur nominale puisse être fixée à 0,99, sachant qu'il s'agit d'une valeur relativement optimiste lorsque l'autodétection consiste simplement en un chien de garde (dans ce cas, on peut supposer que la valeur pour d est de l'ordre de 0,7 à 0,9). Les variations de d prises en compte sont 0,9 - 0,99 - 0,999 - 0,9999.
- La durée moyenne entre deux arrêts de maintenance régulière, τ_{amr} : en général, une maintenance est effectuée tous les 15000 km. Cela correspond en moyenne à un an d'utilisation, c'est-à-dire 500 heures de fonctionnement. La valeur nominale du taux correspondant est alors égale à 0,002/h. Les variations de τ_{amr} sont 50h, 500h et 5000h.
- La durée moyenne entre deux arrêts normaux, τ_{fvoy} : nous supposons que la durée de voyage en moyenne est d'une heure, le taux nominal est alors égal à 1/h. Les variations de τ_{fvoy} sont 0,1h - 1h - 10h.
- La durée moyenne du respect de la maintenance d'urgence, τ_{rmu} : il s'agit d'un paramètre qui dépend du conducteur et des circonstances, sa valeur est difficile à estimer. On peut supposer que, dans le pire des cas, le conducteur se rendra à une station service après une heure et dans le meilleur des cas après environ 30 secondes. La valeur nominale de τ_{rmu} se trouve entre ces deux bornes et nous proposons de la fixer à 6 minutes ou 0,1h (cela correspond à un taux nominal de 10/h). Notons que cette valeur est certainement optimiste dans un cas réel et représente plutôt une consigne donnée par le constructeur. C'est

pour cela que nous étudions une gamme de valeurs de τ_{rmu} qui est très large : 0,01h - 0,1h - 1h - 10h - 100h.

- Le taux d'accident, λ_{acc} : ce taux nous intéresse surtout dans le cas du coussin gonflable, car c'est suite à un accident que le coussin gonflable doit se déployer. On suppose que, pendant vingt ans d'utilisation (10000 heures de roulage), on a au maximum trois accidents qui nécessitent le gonflement du coussin. Cela correspond à un taux nominal de $3*10^{-4}$/h, une valeur dont les variations sont $1*10^{-4}$/h, $3*10^{-4}$/h, $5*10^{-4}$/h, $7*10^{-4}$/h et $9*10^{-4}$/h
- La couverture de détection par comparaison, *cd* : dans le cas d'une architecture Duplex, la présence d'un comparateur introduit ce nouveau paramètre ainsi que la possibilité de tenter un recouvrement global (voir chapitre III) suite à la détection d'erreur par le comparateur. Le comparateur permet de détecter des erreurs qui se propagent à la sortie effective du calculateur. De plus, des erreurs latentes peuvent être détectées par la comparaison de contextes (par exemple, les étapes de calcul internes ou bien le contenu de la mémoire). La valeur nominale de *cd* peut être fixée à 0,999, la gamme des valeurs étudiées est 0,9 - 0,99 - 0,999 - 0,9999.

IV.2 Analyse du coussin gonflable

Dans ce paragraphe, nous proposons d'évaluer la sûreté de fonctionnement du coussin gonflable. Le caractère sécuritaire de cette fonction induit de mesurer la sécurité (*S*) ainsi que l'indisponibilité avant accident (*UAA*).

A l'heure actuelle, les systèmes réels de coussin gonflable offerts sur le marché de l'automobile sont basés sur des architectures de type Simplex [Bergfried *et al.* 1992] ou Duplex [Bergfried *et al.* 1990]. C'est pour cela que nous présentons d'abord dans le paragraphe IV.2.1 plusieurs résultats obtenus à partir de l'architecture Simplex, où nous étudions par ailleurs les deux politiques de signalisation d'erreur (dénommées respectivement "Coussin-Simplex-1" et "Coussin-Simplex-2") ainsi que l'architecture simplifiée du Simplex (appelée "Coussin-Simplex-i"). Le paragraphe IV.2.2 est dédié aux résultats de l'architecture Duplex, les deux stratégies de traitement d'erreur étant comparées ("Coussin-Duplex-1" et "Coussin-Duplex-2"). Dans l'avenir, compte tenu de l'intégration des systèmes embarqués sur l'automobile, on pourrait imaginer une architecture du type TMR (cf. architecture de la *Figure 2-7*), c'est pour cela que le paragraphe IV.2.3 contient les résultats obtenus pour le "Coussin-TMR". Une comparaison plus approfondie des architectures Simplex, Duplex et TMR est donnée dans le paragraphe IV.2.4.

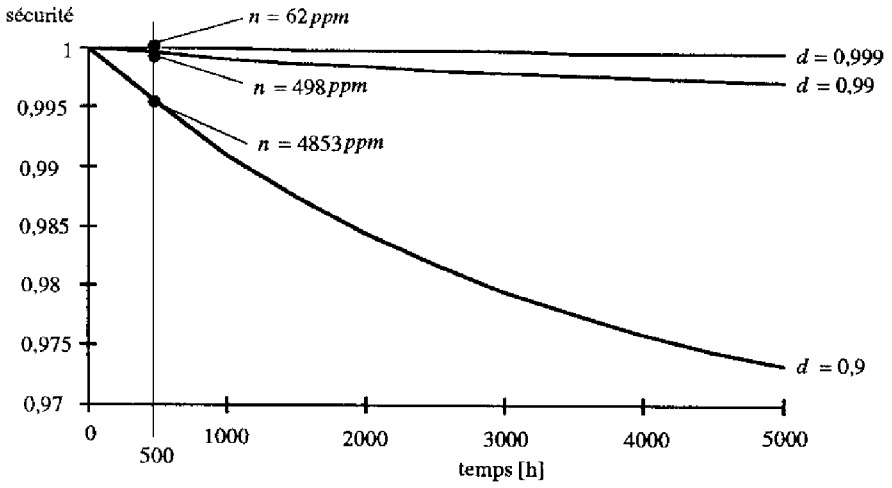
IV.2.1 Architecture Simplex

Dans les 7 premiers sous-paragraphes, nous allons analyser différents résultats générés à partir du modèle Coussin-Simplex-1. Le paragraphe IV.2.1.8 est ensuite dédié à la comparaison des architectures Coussin-Simplex-1 et Coussin-Simplex-2. L'architecture Coussin-Simplex-i est étudiée dans le dernier sous-paragraphe.

IV.2.1.1 Sécurité en fonction du temps

Afin de procurer une confiance accrue dans les modèles et les résultats correspondants, nous présentons dans ce paragraphe à titre d'exemple l'évolution de la sécurité en fonction du temps pour différentes valeurs des paramètres.

Dans un premier temps, nous montrons l'évolution temporelle de la sécurité en fonction de la couverture d'autodétection d (*Figure 4-1*), les autres paramètres étant fixés à leurs valeurs nominales (la figure illustre également les valeurs de n qui sont calculées en multipliant la non-sécurité à l'instant t = 500h par 1 million).



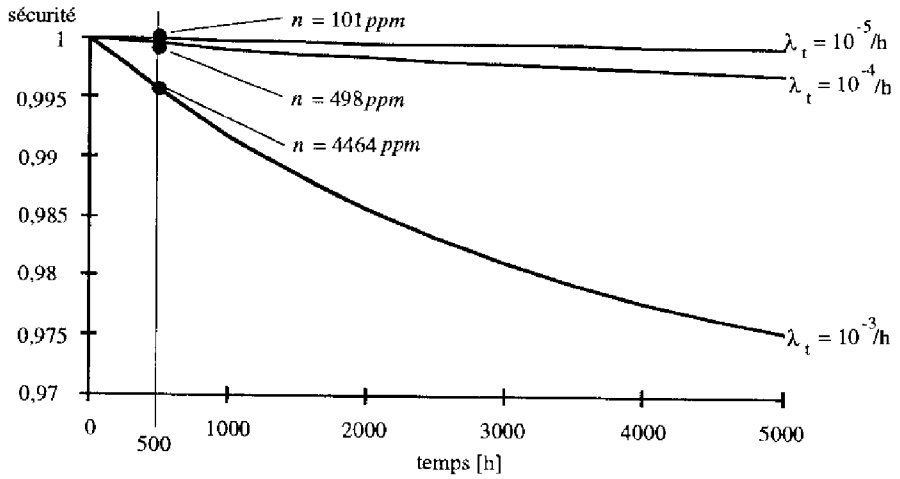
$\tau_{reca}=10^{-4}$h ; τ_{amr}=500h ; τ_{tvoy}=1h ; $\lambda_{acc}=3*10^{-4}$/h ; τ_{rmu}=0,1h ; i=0,05 ; $\lambda_t=10^{-4}$/h ; $\lambda_p=10^{-5}$/h

Figure 4-1 : Sécurité ponctuelle du Coussin-Simplex-1 en fonction de d

Mis à part que la sécurité initiale est égale à 1 et qu'elle baisse suivant une exponentielle négative, on note surtout que la valeur asymptotique de la sécurité est non nulle. Cela est dû au fait que le gonflement du coussin lors d'un accident est modélisé par un état absorbant qui n'est pas considéré comme un état catastrophique. La valeur asymptotique de la sécurité exprime alors "l'équilibre" entre le fonctionnement correct (c'est-à-dire, le gonflement lors d'un accident) et la défaillance catastrophique (c'est-à-dire, le non-gonflement lors de l'accident ou le gonflement intempestif). On vérifie que la sécurité asymptotique est proportionnelle à

la couverture d'autodétection. Globalement, les valeurs de n sont beaucoup trop élevées par rapport à l'objectif 1 *ppm*, même pour $d = 0{,}999$.

La couverture d'autodétection d ne pouvant pas dépasser environ 0,99 dans un système Simplex réel, une amélioration pourrait être apportée éventuellement par un blindage du système qui diminuerait le taux de faute transitoire λ_t. C'est pour cela que nous étudions ici également l'influence sur la sécurité du taux de fautes transitoires λ_t. La *Figure 4-2* illustre l'évolution temporelle de la sécurité en fonction de λ_t, les autres paramètres étant fixés à leurs valeurs nominales. On vérifie notamment que la sécurité augmente (c'est-à-dire, n diminue) lorsque λ_t diminue. Les valeurs de n sont toujours trop élevées par rapport à l'objectif. Nous allons revenir sur ce point dans le paragraphe IV.2.1.7.



$\tau_{reca}=10^{-4}$h ; $\tau_{amr}=500$h ; $\tau_{tvoy}=1$h ; $\lambda_{acc}=3*10^{-6}$/h ; $\tau_{rmu}=0{,}1$h ; $b=0{,}05$; $d=0{,}99$; $\lambda_p=10^{-5}$/h

Figure 4-2 : Sécurité ponctuelle du Coussin-Simplex-1 en fonction de λ_t

Il est intéressant de relier la valeur asymptotique de la sécurité à la mesure de l'indisponibilité avant accident *UAA*. La *Figure 4-3* rappelle la structure principale de la chaîne de Markov associée au modèle RdPSG du coussin gonflable. Nous rappelons que la sécurité S du système correspond à la probabilité d'être dans les états cp, cnp ou cg. L'indisponibilité avant accident *UAA* se calcule en divisant le temps moyen passé dans l'état cnp par la somme des temps moyens passés dans les états cnp et cp. La relation entre S et *UAA* peut être approximativement formulée de la façon suivante :

$$\lim_{t\to\infty} S(t) = (1-UAA) * \frac{\lambda_{acc}}{\lambda_{acc}+\lambda_{pr}} \quad (1)$$

avec : $\lambda_{pr} = (1-d)*(1-l)*(\lambda_t + \lambda_p)$

où le terme multiplicatif de (1-*UAA*) représente la proportion du flux de sortie de l'état cp conduisant à l'état sûr cg.

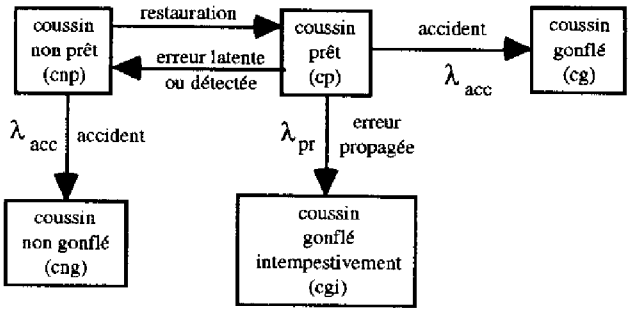


Figure 4-3 : Chaîne de Markov simplifiée du coussin gonflable

Le *Tableau 4-1* trace, en fonction de d et λ_t, les valeurs que nous obtenons pour $S(\infty)$ et *UAA*. Les valeurs de $S(\infty)$ issues du traitement du modèle par SURF-2 sont comparées avec celles obtenues par le calcul de l'équation (1) (notées $S'(\infty)$). Les valeurs nominales des paramètres variables sont indiquées en gras, cette convention étant adaptée dans la suite de ce chapitre.

Les résultats du tableau permettent de vérifier que l'écart entre $S(\infty)$ et $S'(\infty)$ est très faible, ce qui confirme la bonne approximation donnée par l'équation (1). Cette confirmation entre les résultats de $S(\infty)$ issus du modèle traité par SURF-2 et l'expression (1) obtenue par une approximation conforte notre confiance dans la validité du modèle et de son traitement.

IV.2.1.2 Influence de l'intervalle de maintenance régulière

Lors de la construction du modèle du calculateur, nous avons supposé que la maintenance permet de recouvrir des erreurs latentes qui sont dues à des fautes permanentes. Dans ce paragraphe, nous mesurons l'évolution de n en fonction de la proportion d'erreurs latentes l et du temps moyen entre arrêts de maintenance réguliers τ_{amr}. Le *Tableau 4-2* montre les résultats que nous obtenons pour le jeu de paramètres défini.

Tableau 4-1 : Valeurs de UAA, S(∞) et S'(∞) pour le Coussin-Simplex-1 en fonction de d et λ_t

τ_{reca}=10^{-4}h ; τ_{amr}=500h ; τ_{fvoy}=1h ; λ_{acc}=3*10^{-4}/h ; τ_{rmu}=0,1h ; *l*=0,05 ; λ_p=10^{-5}/h

d		λ_t 10^{-3}/h	**10^{-4}/h**	10^{-5}/h	10^{-6}/h
0,9	*UAA*	0,00027	0,00022	0,00022	0,00022
	S(∞)	0,76214	0,96765	0,99381	0,99650
	S'(∞)	0,75747	0,96612	0,99349	0,99631
0,99	*UAA*	0,00027	0,00022	0,00022	0,00022
	S(∞)	0,97017	0,99649	0,99919	0,99946
	S'(∞)	0,96875	0,99631	0,99915	0,99943
0,999	*UAA*	0,00027	0,00022	0,00022	0,00022
	S(∞)	0,99672	0,99946	0,99973	0,99976
	S'(∞)	0,99654	0,99943	0,99972	0,99975
0,9999	*UAA*	0,00027	0,00022	0,00022	0,00022
	S(∞)	0,99944	0,99976	0,99979	0,99979
	S'(∞)	0,99941	0,99974	0,99977	0,99978

Tableau 4-2 : Valeurs de n pour le Coussin-Simplex-1 en fonction de l et τ_{amr}

τ_{reca}=10^{-4}h ; τ_{fvoy}=1h ; λ_{acc}=3*10^{-4}/h ; τ_{rmu}=0,1h ; *d*=0,99 ; λ_t=10^{-4}/h ; λ_p=10^{-5}/h

τ_{amr} [h]	*l* 0,0005	0,005	**0,05**	0,5
50	510,5 *ppm*	508,6 *ppm*	489,1 *ppm*	293,4 *ppm*
500	510,6 *ppm*	509,5 *ppm*	498,4 *ppm*	387,4 *ppm*
5000	510,7 *ppm*	509,9 *ppm*	502,3 *ppm*	426,2 *ppm*

On vérifie d'abord que l'allongement de l'intervalle de maintenance procure une légère baisse de la sécurité, l'augmentation de *l* entraînant logiquement une influence plus forte de τ_{amr}. Comme on pouvait s'y attendre, l'influence de la durée de l'intervalle de maintenance sur la sécurité est négligeable pour la valeur nominale de *l*. Par conséquent, le paramètre τ_{amr} est par la suite fixé à sa valeur nominale (500h).

IV.2.1.3 Influence de la durée de voyage

Dans le modèle du calculateur, le redémarrage (après une fin de voyage) permet de recouvrir fortuitement des erreurs latentes qui sont dues à des fautes transitoires. De façon similaire au paragraphe précédent, nous mesurons *n* en fonction de la proportion d'erreurs latentes *l* et du temps moyen d'une durée de voyage τ_{fvoy}. Les résultats obtenus dans le *Tableau 4-3* permettent de vérifier d'abord que l'allongement de la durée de voyage est légèrement nuisible à la sécurité, cette influence devenant relativement forte pour des valeurs élevées de *l*. Pour des valeurs de *l* qui se situent aux alentours de la valeur nominale, l'influence de τ_{fvoy} sur *n* est

néanmoins très faible. Par conséquent, le paramètre τ_{fvoy} est par la suite fixé à sa valeur nominale (1h).

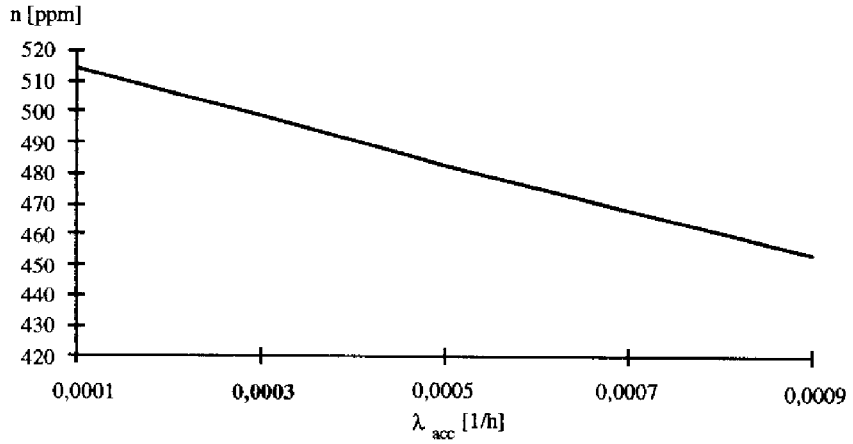
Tableau 4-3 : Valeurs de n pour le Coussin-Simplex-1 en fonction de l et τ_{fvoy}

$\tau_{reca}=10^{-4}$h ; $\tau_{amr}=500$h ; $\lambda_{acc}=3*10^{-4}$/h ; $\tau_{rmu}=0,1$h ; $d=0,99$; $\lambda_t=10^{-4}$/h ; $\lambda_p=10^{-5}$/h

τ_{fvoy} [h] \ l	0,0005	0,005	**0,05**	0,5
0,1	510,6 *ppm*	509,4 *ppm*	497,8 *ppm*	381,2 *ppm*
1	510,6 *ppm*	509,5 *ppm*	498,4 *ppm*	387,4 *ppm*
10	510,7 *ppm*	510,1 *ppm*	504,5 *ppm*	448,5 *ppm*

IV.2.1.4 Influence du taux d'accident

L'influence du taux d'accident sur la sécurité est illustrée sur la *Figure 4-4*. On constate qu'une augmentation du taux d'accident est paradoxalement responsable d'une baisse de *n*, donc d'une augmentation de la sécurité. Cela peut être expliqué par le fait qu'un taux d'accident élevé signifie qu'il y a une forte probabilité que le coussin gonflable soit sollicité avant qu'il soit défaillant. Ce résultat est bien entendu d'un intérêt purement académique, le taux d'accident est fixé par la suite à sa valeur nominale ($3*10^{-4}$/h).



$\tau_{reca}=10^{-4}$h ; $\tau_{amr}=500$h ; $\tau_{fvoy}=1$h ; $\tau_{rmu}=0,1$h ; $d=0,99$; $l=0,05$; $\lambda_t=10^{-4}$/h ; $\lambda_p=10^{-5}$/h

Figure 4-4 : Valeurs de n pour le Coussin-Simplex-1 en fonction de λ_{acc}

IV.2.1.5 Le risque supplémentaire

Comme déjà expliqué au paragraphe III.4.1, l'introduction du coussin gonflable peut créer un risque supplémentaire qui est dû au gonflement intempestif. Nous avons formulé la condition que ce risque doit être très petit devant le risque sans

coussin, c'est-à-dire qu'il faut vérifier $n_{gi} \ll n_{acc}$. Le *Tableau 4-4* montre les valeurs de n_{gi}, n_{ng} et $n = n_{gi} + n_{ng}$ pour différentes valeurs de la couverture d'autodétection d.

Tableau 4-4 : Valeurs de n_{gi}, n_{ng} et n pour le Coussin-Simplex-1 en fonction de d et l

$\tau_{reca}=10^{-4}$h ; $\tau_{amr}=500$h ; $\tau_{tvoy}=1$h ; $\lambda_{acc}=3*10^{-4}$/h ; $\tau_{rmu}=0,1$h ; $\lambda_{l}=10^{-4}$/h ; $\lambda_{p}=10^{-5}$/h

d	l=0,05			l=0,5		
	n_{gi} [ppm]	n_{ng} [ppm]	n [ppm]	n_{gi} [ppm]	n_{ng} [ppm]	n [ppm]
0,9	4839	13,3	4852	2548	132,1	2680
0,99	485	13,4	498	255	132,3	387
0,999	49	13,4	62	26	132,3	158

Le taux d'accident nominal $\lambda_{acc} = 3*10^{-4}$/h conduit à un nombre de véhicules accidentés par an pour un million de véhicules, n_{acc}, donné par :

$$n_{acc} = 10^6 * (1 - e^{-3*10^{-4}*500}) \ ppm$$
$$= 139\ 292 \ ppm$$

L'analyse des valeurs numériques du *Tableau 4-4* pour $l = 0,05$ et $l = 0,5$ montre que la condition $n_{gi} \ll n_{acc}$ est satisfaite pour toutes les valeurs de d. On vérifie que l'augmentation de d induit une légère augmentation de n_{ng} (due aux accidents survenant après la détection d'une erreur mais avant la réparation ou le recouvrement automatique). Cependant, l'augmentation de d conduit aussi à une baisse importante de n_{gi} (due à la baisse de la proportion d'erreurs propagées) et une baisse de la somme $n = n_{gi} + n_{ng}$. Le rapport entre n_{gi} et n_{ng} est fortement lié à la proportion d'erreurs latentes l. Lorsque la proportion d'erreurs latentes est à sa valeur nominale, le nombre de défaillances dues au non-gonflement du coussin n_{ng} peut être négligé par rapport au nombre de défaillances dues au gonflement intempestif n_{gi}. Nous vérifions que le rapport entre n_{ng} et n_{gi} est sensiblement plus élevé pour $l = 0,5$. Cela est vrai notamment pour $d = 0,999$ (ce qui signifie seulement (1-0,999)*0,5 = 0,05% d'erreurs propagées), où n_{gi} est plus petit que n_{ng}.

IV.2.1.6 Partage de la responsabilité entre chauffeur et fabricant

Dans le paragraphe III.4.1, nous avons proposé de décomposer n en les deux composantes n_{fabr} et n_{chauf}, afin de pouvoir distinguer les erreurs signalées des erreurs non-signalées. Il est évident que la durée moyenne du respect de la maintenance d'urgence τ_{rmu} (suite à une signalisation d'erreur) influe en premier sur n_{chauf}. Le *Tableau 4-5* montre les valeurs pour n_{fabr}, n_{chauf} et $n = n_{fabr} + n_{chauf}$ en fonction de τ_{rmu}.

Nous vérifions qu'une augmentation de τ_{rmu} induit une forte augmentation de n_{chauf} et une légère baisse de n_{fabr}. On vérifie également que n augmente lorsque la durée du respect de la maintenance d'urgence croît. L'analyse du tableau nous

apprend que n_{chauf} est très faible par rapport à n_{fabr} pour toutes les valeurs de τ_{rmu}, seulement au-delà de τ_{rmu} = 100h, l'influence de τ_{rmu} sur n devient plus perceptible.

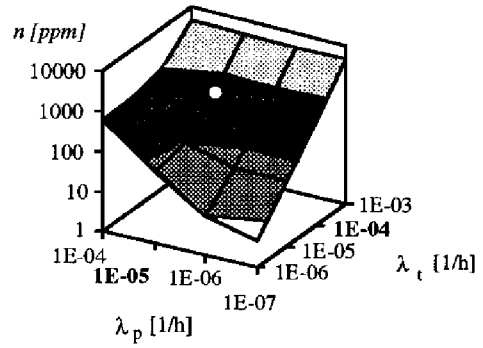
Tableau 4-5 : Valeurs de n_{fabr}, n_{chauf} et n pour le Coussin-Simplex-1 en fonction de τ_{rmu}

τ_{reca}=10^{-4}h ; τ_{amr}=500h ; τ_{tvoy}=1h ; λ_{acc}=3*10^{-4}/h ; d=0,99 ; l=0,05 ; λ_t=10^{-4}/h ; λ_p=10^{-5}/h

τ_{rmu} [h]	n_{fabr} [*ppm*]	n_{chauf} [*ppm*]	n [*ppm*]
0,01	498,3	0	498,3
0,1	498,3	0,1	498,4
1	498,3	1,3	499,6
10	498,2	12,8	511
100	497,9	103,7	601,6

IV.2.1.7 Influence des fautes transitoires et fautes permanentes

Lors de la modélisation du calculateur, nous avons distingué entre fautes transitoires et fautes permanentes. L'influence des taux de faute sur la sécurité est illustrée sur la *Figure 4-5*, où l'on trace — en échelle logarithmique — n en fonction de λ_t et de λ_p (les valeurs nominales sont indiquées en gras et la valeur associée de n est illustrée par un point). Les courbes qui séparent les différents motifs de la surface tracée correspondent aux équipotentielles, c'est-à-dire à n = constante.



τ_{reca}=10^{-4}h ; τ_{amr}=500h ; τ_{tvoy}=1h ; λ_{acc}=3*10^{-4}/h ; τ_{rmu}=0,1h ; d=0,99 ; l=0,05

Figure 4-5 : Valeurs de n pour le Coussin-Simplex-1 en fonction de λ_t et λ_p

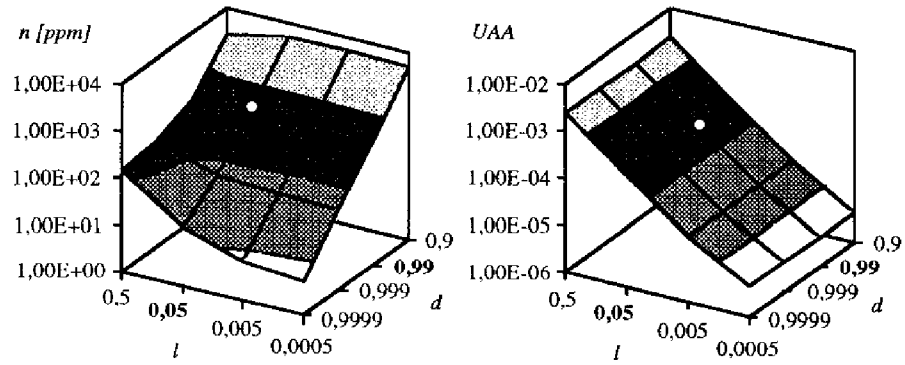
On vérifie non seulement que l'augmentation de λ_t et de λ_p entraîne une augmentation de n, mais aussi que l'influence des fautes transitoires sur n est plus forte que celle des fautes permanentes. L'influence plus forte de λ_t sur la sécurité peut être expliquée par le fait que λ_{pr}, qui est le facteur le plus influant sur la sécurité, suit proportionnellement la variation de λ_t et de λ_p, λ_t étant dix fois plus fort que λ_p dans notre exemple de calcul. La figure illustre que l'objectif quantitatif pour n que

nous avons fixé à $n = 1$ *ppm* n'est pas atteint avec la gamme de valeurs considérées, notamment autour des valeurs nominales de λ_t et de λ_p. Sachant que les fautes transitoires sont essentiellement dues aux contraintes sévères d'environnement, on pourrait doter le système d'un blindage conséquent afin d'obtenir une valeur pour le taux de fautes transitoires λ_t qui est bien plus faible que sa valeur nominale. Néanmoins, compte tenu du fait que la valeur nominale du taux de fautes permanentes λ_p est généralement difficile à améliorer, les résultats de la figure nous indiquent que l'objectif fixé pour n ne peut pas être atteint même pour λ_t très faible.

IV.2.1.8 Influence de la politique de signalisation

Dans ce paragraphe, nous étudions la variation de n ainsi que de l'indisponibilité avant accident *UAA* en fonction des paramètres d et l. L'architecture Simplex permet l'implantation de deux politiques différentes de signalisation d'erreur, qui sont comparées dans ce paragraphe. La différence entre ces deux politiques de signalisation est essentiellement liée à la durée du recouvrement automatique. Afin de pouvoir mieux observer cette différence, nous choisissons $\tau_{reca} = 0{,}01$h, ce qui est une valeur cent fois plus élevée que la valeur nominale (voir paragraphe IV.1).

La *Figure 4-6* illustre pour la politique 1, en échelle logarithmique, l'influence de d et l sur n et sur *UAA*. On observe que la baisse de l induit une baisse de n seulement pour des valeurs très élevées de d. Cela peut être expliqué par le fait que le nombre des erreurs non latentes qui se propagent est globalement très faible lorsque d est élevé. En effet, la baisse de la proportion d'erreurs latentes est dans ce cas significative d'une augmentation sensible du nombre d'erreurs détectées. Par contre, pour des faibles valeurs de d, la proportion d'erreurs propagées devient globalement plus forte et, pour une baisse de l, on peut observer l'effet de l'augmentation des erreurs propagées par une augmentation de n. La diminution de l induit également une baisse de *UAA*, car la proportion d'erreurs détectées augmente sensiblement (procurant ainsi la possibilité de réparation avant une sollicitation éventuelle). Par contre, la variation de d (c'est-à-dire, la variation de la proportion d'erreurs propagées) n'a pratiquement aucune influence sur *UAA*. En effet, dans la plage numérique analysée, le taux de propagation d'erreur peut être négligé par rapport au taux d'accident, ce qui veut dire que les états "coussin non prêt" et "coussin prêt" du modèle markovien de la *Figure 4-3* sont tous les deux quittés vers leurs états absorbants respectifs ("coussin non gonflé" et "coussin gonflé") approximativement avec le taux λ_{acc}. L'influence de d sur *UAA* devrait devenir plus perceptible pour des valeurs de d extrêmement faibles. Nous constatons que l'objectif $n = 1$ *ppm* n'est pas atteint pour la plage des valeurs numériques de d et l, tandis que l'objectif $UAA = 10^{-4}$ est atteint pour $l < 0{,}05$.



τ_{reca}=0,01h ; τ_{amr}=500h ; τ_{fvoy}=1h ; λ_{acc}=3*10^{-4}/h ; τ_{rmu}=0,1h ; λ_t=10^{-4}/h ; λ_p=10^{-5}/h

Figure 4-6 : Valeurs de n et UAA pour le Coussin-Simplex-1 en fonction de d et l

Dans un but de comparaison des deux politiques de signalisation d'erreur, le *Tableau 4-6* montre le rapport entre les *n* et *UAA* de la politique 1 et 2 (n_1/n_2 et UAA_1/UAA_2) pour différentes valeurs de *d* et *l*.

Tableau 4-6 : Rapport entre les valeurs de n et UAA pour le Coussin-Simplex-1 et le Coussin-Simplex-2 en fonction de d et l

n_1=*n* pour la polit. 1 ; n_2=*n* pour la polit. 2 ; UAA_1=*UAA* pour la polit. 1 ; UAA_2=*UAA* pour la polit. 2 ;
τ_{reca}=0,01h ; τ_{amr}=500h ; τ_{fvoy}=1h ; λ_{acc}=3*10^{-4}/h ; τ_{rmu}=0,1h ; λ_t=10^{-4}/h ; λ_p=10^{-5}/h

d		*l* 0,0005	0,005	**0,05**	0,5
0,9	n_1/n_2	1,00000	1,00000	1,00002	1,00000
	UAA_1/UAA_2	1,04357	1,00714	1,00073	1,00004
0,99	n_1/n_2	1,00020	1,00020	1,00000	1,00026
	UAA_1/UAA_2	1,04593	1,00780	1,00080	1,00004
0,999	n_1/n_2	1,00195	1,00000	1,00000	1,00000
	UAA_1/UAA_2	1,04615	1,00786	1,00081	1,00004
0,9999	n_1/n_2	1,00000	1,00000	1,00546	1,00000
	UAA_1/UAA_2	1,04618	1,00787	1,00081	1,00004

On vérifie d'abord que la sécurité et la disponibilité de la politique 2 (signalisation des erreurs provoquées par des fautes à la fois transitoires et permanentes) sont globalement meilleures que celles de la politique 1 (signalisation seulement des erreurs dues à des fautes permanentes). Cette amélioration est cependant très, très faible et elle deviendrait même invisible pour la valeur nominale de τ_{reca}. La faible différence s'explique par le fait que le processus de recouvrement automatique est beaucoup plus rapide que le respect de la maintenance d'urgence par le conducteur (τ_{rmu} = 0,1h). La possibilité de recouvrir une faute transitoire par intervention extéricure est alors négligeable au niveau de la sécurité et disponibilité. On note

également que le rapport UAA_1/UAA_2 augmente lorsque l diminue et d augmente. On vérifie alors que l'augmentation de d et la baisse de l (plus d'erreurs détectées) signifie un avantage plus fort pour la politique 2 au niveau de la disponibilité.Une autre conséquence de la politique 2 est que le voyant s'affiche bien plus souvent que dans la politique 1, et dans la plupart des cas, il s'éteint très vite (après le succès du recouvrement automatique). Un allumage fréquent du voyant n'étant pas souhaité tant qu'il n'améliore que très peu la sécurité des passagers, nous pouvons conclure que l'implantation de la politique 2 ne serait pas justifiée.

IV.2.1.9 Influence du recouvrement automatique

La tentative de recouvrement automatique est un processus à durée courte par rapport aux processus de création de fautes. Nous pouvons supposer alors que la durée du recouvrement automatique puisse être négligée, c'est-à-dire que la transition correspondante du modèle puisse être remplacée par une transition instantanée. Cela nous permettrait de générer une chaîne de Markov simplifiée, de réduire le nombre de paramètres numériques à définir et de faciliter l'analyse des résultats.

Le *Tableau 4-7* compare les résultats obtenus pour n et UAA à partir du modèle Coussin-Simplex-i avec ceux du modèle Coussin-Simplex-1.

Tableau 4-7 : Rapport entre les valeurs de n et UAA pour le Coussin-Simplex-1 et le Coussin-Simplex-i en fonction de d et l

$n_1=n$ pour τ_{reca}=0,01h (polit.1) ; $n_i=n$ pour τ_{reca}=inst. ;
$UAA_1=UAA$ pour τ_{reca}=0,01h (polit.1) ; $UAA_i=UAA$ pour τ_{reca}=inst. ;
τ_{amr}=500h ; τ_{fvoy}=1h ; $\lambda_{acc}=3*10^{-4}$/h ; τ_{rmu}=0,1h ; $\lambda_t=10^{-4}$/h ; $\lambda_p=10^{-5}$/h

d		l 0,0005	0,005	**0,05**	0,5
0,9	n_1/n_i	1,00002	1,00002	1,00004	1,00004
	UAA_1/UAA_i	1,31680	1,04258	1,00421	1,00022
0,99	n_1/n_i	1,00039	1,00039	1,00020	1,00026
	UAA_1/UAA_i	1,33872	1,04666	1,00463	1,00024
0,999	n_1/n_i	1,00390	1,00191	1,00162	1,00063
	UAA_1/UAA_i	1,34085	1,04706	1,00467	1,00025
0,9999	n_1/n_i	1,01852	1,03077	1,01099	1,00000
	UAA_1/UAA_i	1,34106	1,04710	1,00468	1,00025

L'analyse du tableau nous permet de vérifier que la sécurité n'est influencée que très peu par l'introduction d'un recouvrement automatique instantané. Afin de mieux voir la différence numérique entre les deux modèles, nous choisissons τ_{reca} = 0,01h, ce qui est une valeur cent fois plus élevée que la valeur nominale. La différence que l'on peut observer serait invisible si τ_{reca} prenait sa valeur nominale (τ_{reca} =

0,0001h). On confirme alors l'hypothèse du départ, par conséquent nous représentons désormais le recouvrement automatique par une transition instantanée.

IV.2.2 Architecture Duplex : deux stratégies de tolérance aux fautes

L'analyse numérique des modèles Coussin-Duplex-1 et Coussin-Duplex-2 se fait de façon analogue à celle des modèles Coussin-Simplex. On note cependant que la durée de recouvrement automatique est supposée instantanée. Concernant le recouvrement global, nous supposons que sa durée est aussi courte que celle du recouvrement automatique local, elle peut alors être exprimée également par une transition instantanée. Les études de sécurité en fonction du temps permettant d'accroître la confiance dans le modèle ne sont pas présentées ici, car elles n'ont que peu d'intérêt dans le cadre de ce mémoire. Nous retenons de ces études que l'évolution de la sécurité est encore de forme exponentielle négative avec une asymptote non nulle.

Ce paragraphe est destiné à une comparaison entre les deux stratégies de traitement d'erreur du Duplex décrites au paragraphe III.4.6, c'est-à-dire :

- Stratégie 1 : coupure du service suite à une autodétection d'erreur dans un calculateur
- Stratégie 2 : continuité du service suite à une autodétection d'erreur dans un calculateur (par commutation sur le calculateur de secours)

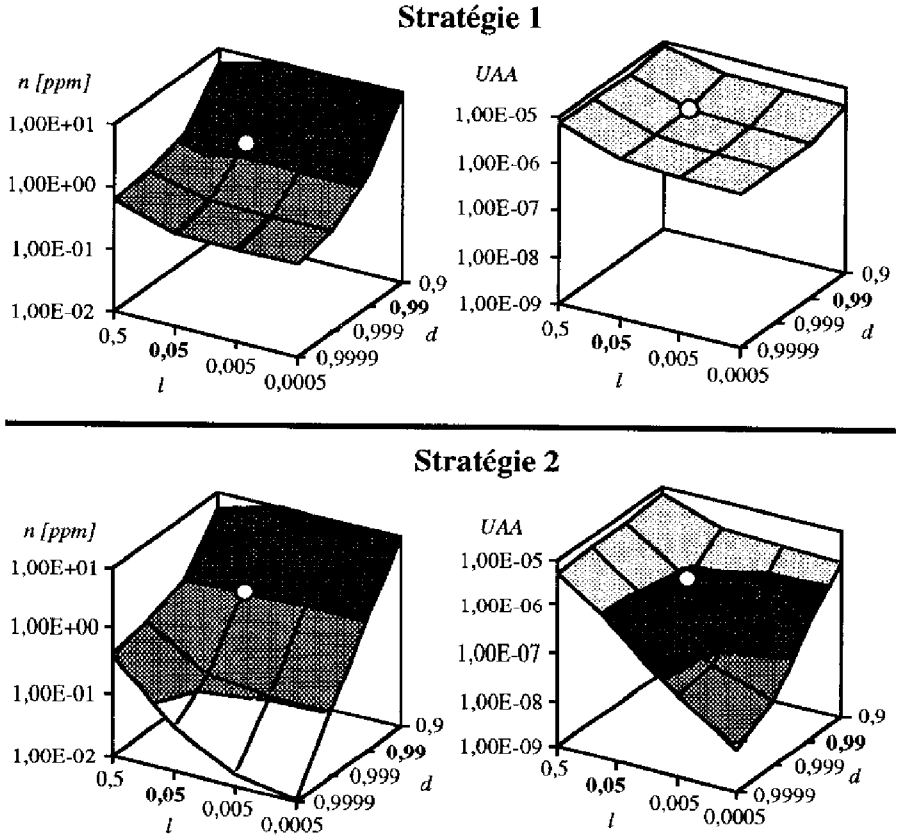
L'architecture Duplex étant composée de deux calculateurs et d'un comparateur, on étudiera l'évolution de n et de UAA en fonction de trois paramètres, à savoir la couverture d'autodétection d, la couverture de détection par comparaison cd et la proportion d'erreurs latentes l. Le premier sous-paragraphe est destiné à l'analyse de n et de UAA en fonction de d et l, le deuxième sous-paragraphe illustre les effets de la variation de d et de cd.

IV.2.2.1 Influence de l'autodétection et de la latence

La *Figure 4-7* illustre, en échelle logarithmique, l'évolution de n et de UAA en fonction de d et l pour les deux stratégies.

Nous vérifions que l'évolution de n correspond dans la tendance à celle de l'architecture Simplex, les valeurs absolues de n étant plus basses d'un facteur d'environ 100. Pour les deux stratégies, l'objectif $n = 1$ *ppm* est atteint lorsque $d \geq 0{,}999$ et $l \leq 0{,}05$. Concernant l'évolution de UAA, on vérifie que la baisse de l et l'augmentation de d induisent une baisse de UAA, l'objectif $UAA = 10^{-4}$ étant largement atteint à l'intérieur des plages définies.

Nous vérifions que l'apport de la compensation d'erreur (stratégie 2) par rapport à la simple détection d'erreur (stratégie 1) au niveau de la disponibilité devient plus grand lorsque l baisse et d augmente, l'apport étant significatif pour des valeurs de $d > 0{,}99$ et $l < 0{,}05$. De plus, nous constatons les mêmes phénomènes en ce qui concerne la sécurité. Cela s'explique par le fait qu'une plus grande disponibilité du coussin gonflable entraîne également une baisse de la probabilité de non-gonflement lors d'une sollicitation éventuelle (la probabilité de gonflement intempestif étant très peu affectée par la stratégie de compensation). Aux alentours des valeurs nominales de d et l, néanmoins, l'apport de la stratégie de compensation d'erreur est relativement faible.

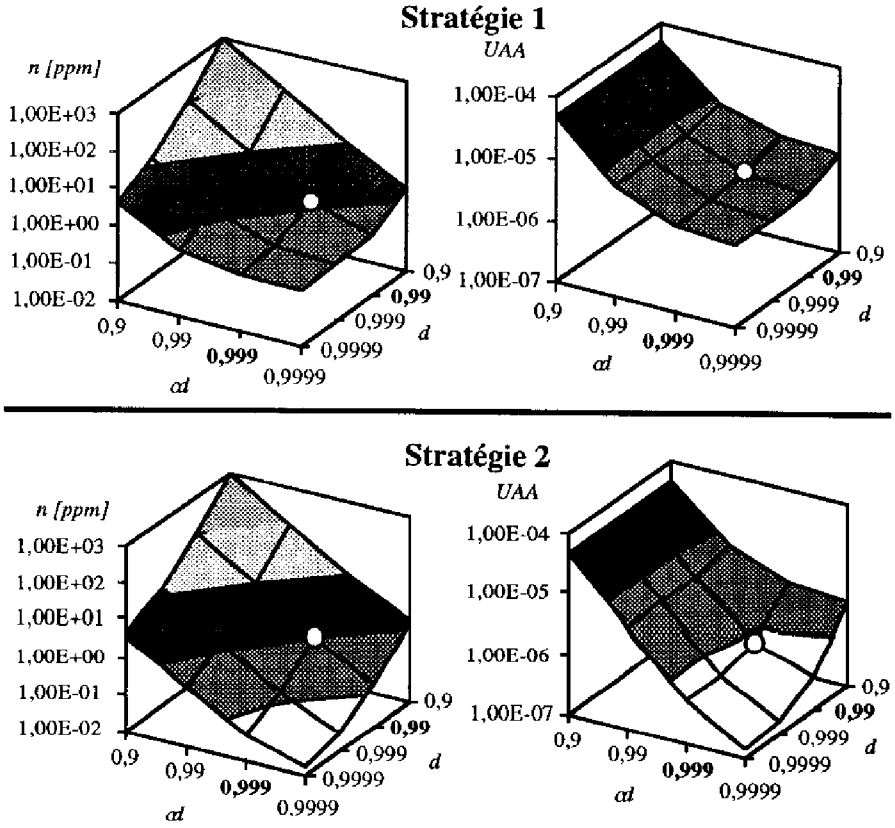


τ_{amr}=500h ; τ_{fvoy}=1h ; λ_{acc}=3*10^{-4}/h ; τ_{rmu}=0,1h ; cd=0,999 ; λ_4=10^{-4}/h ; λ_p=10^{-5}/h

Figure 4-7 : Valeurs de n et UAA pour le Coussin-Duplex-1 et le Coussin-Duplex-2 en fonction de d et l

IV.2.2.2 Influence de l'autodétection et de la détection globale

L'architecture Duplex dispose d'un comparateur dont la couverture de détection *cd* influe également sur la sécurité. La *Figure 4-8* illustre, en échelle logarithmique, le fait que la baisse de *d* et de *cd* entraînent une baisse de *n* et de *UAA*. On vérifie que la stratégie de compensation d'erreur est particulièrement avantageuse au niveau de la disponibilité pour $d \geq 0{,}999$ et $cd \geq 0{,}999$. De façon analogue au paragraphe précédent, nous constatons également une nette amélioration de la sécurité pour ces valeurs. Dans les deux stratégies, l'objectif $n = 1$ *ppm* est atteint pour $d \geq 0{,}99$ et $cd \geq 0{,}99$. En ce qui concerne l'objectif $UAA = 10^{-4}$, nous constatons qu'il est atteint pour toutes les valeurs de la gamme de *d* et de *cd*.



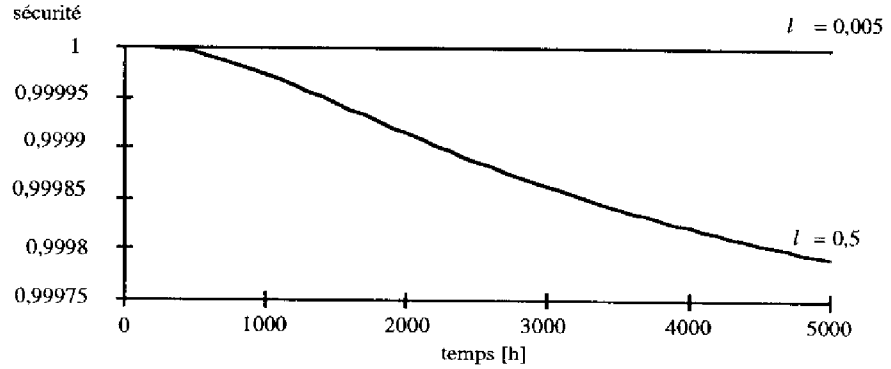
τ_{amr}=500h ; τ_{tvoy}=1h ; λ_{acc}=3*10^{-4}/h ; τ_{rmu}=0,1h ; k=0,05 ; λ_t=10^{-4}/h ; λ_p=10^{-5}/h

Figure 4-8 : Valeurs de n et UAA pour le Coussin-Duplex-1 et le Coussin-Duplex-2 en fonction de d et cd

IV.2.3 Architecture TMR

Dans le cas du modèle Coussin-TMR, on dispose de trois calculateurs et d'un voteur. On suppose que le voteur est parfait, par conséquent on n'introduit pas de facteur de couverture par vote. De plus, il est important à noter que les calculateurs n'ont pas d'autodétection locale (et, par conséquent, pas de procédure de recouvrement local). Cela signifie la suppression des paramètres d et τ_{reca}. La *Figure 4-9* illustre l'évolution temporelle de la sécurité pour deux valeurs différentes du paramètre l.

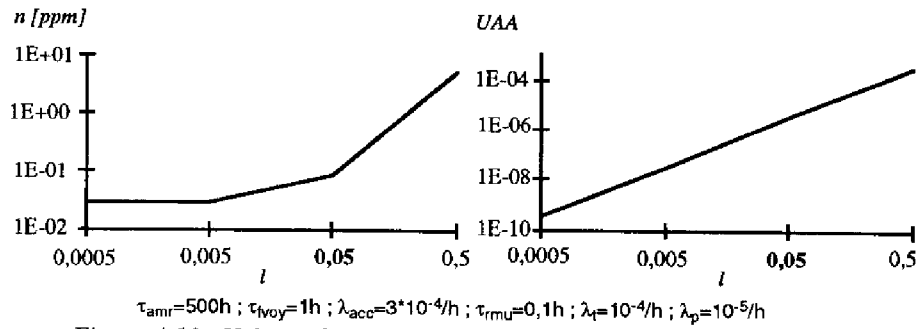
Nous vérifions que la baisse de l (donc, l'augmentation de la proportion d'erreurs propagées) induit l'augmentation du niveau de sécurité. Cela peut être expliqué par le fait que les erreurs propagées sont par définition détectées par le voteur et sont recouvertes soit par masquage (en cas de faute transitoire), soit par réparation (en cas de faute permanente). Le problème principal de l'architecture TMR sont alors les erreurs latentes qui ne peuvent pas être vues par le voteur. Une deuxième caractéristique que l'on peut observer en analysant les courbes, est que l'évolution temporelle de la sécurité ne suit plus une exponentielle négative. Ces courbes confirment les résultats d'autres sources telles que [Siewiorek & Swarz 1992].



τ_{amr}=500h ; τ_{tvoy}=1h ; λ_{acc}=3*10^{-4}/h ; τ_{rmu}=0,1h ; λ_t=10^{-4}/h ; λ_p=10^{-5}/h

Figure 4-9 : Sécurité ponctuelle du Coussin-TMR en fonction de l

La *Figure 4-10* montre l'évolution de n et de *UAA* en fonction de l. Nous vérifions à nouveau que l'augmentation de la proportion d'erreurs latentes est nuisible à la sécurité et à la disponibilité, la sécurité étant particulièrement influencée au-delà de l = 0,005. Nous constatons que l'objectif n = 1 *ppm* et *UAA* = 10^{-4} est atteint pour $l \leq 0,05$.



τ_{amr}=500h ; τ_{lvoy}=1h ; λ_{acc}=3*10^{-4}/h ; τ_{rmu}=0,1h ; λ_t=10^{-4}/h ; λ_p=10^{-5}/h

Figure 4-10 : Valeurs de n et UAA pour le Coussin-TMR en fonction de l

IV.2.4 Comparaison des architectures Simplex, Duplex et TMR

Dans ce paragraphe, nous allons comparer les résultats obtenus pour le Coussin-Simplex simplifié, le Coussin-Duplex (pour les deux stratégies de tolérance) et le Coussin-TMR. Le *Tableau 4-8* montre les résultats obtenus pour la gamme définie des paramètres *d* et *l*, les autres paramètres étant fixés à leurs valeurs nominales. Pour chaque jeu de paramètres, on peut trouver les valeurs de *n* et de *UAA* ; de plus, les valeurs de *n* et de *UAA* pour les architectures Duplex-1, Duplex-2 et TMR sont mises en rapport avec celles du Simplex-i.

Dans un premier temps, nous analysons les différentes valeurs de *n*. On constate que l'avantage du Duplex-1 et du Duplex-2 par rapport au Simplex-i est important pour tous les jeux de paramètres, ce qui est dû à l'amélioration de la couverture de détection d'erreur par la comparaison des sorties des deux calculateurs (induisant surtout une baisse de la proportion d'erreurs propagées). Par contre, le faible avantage du Duplex-2 par rapport au Duplex-1 illustre que le coussin gonflable n'est sollicité que rarement et peut être dans la plupart des cas réparé avant d'être sollicité. La stratégie de compensation d'erreur offerte par le Duplex-2 n'est peut-être pas alors méritée pour le coussin gonflable. On note également que, pour d = 0,99, l'augmentation de la proportion d'erreurs latentes entraîne un avantage décroissant des architectures Duplex par rapport à l'architecture Simplex. Ici se traduit le fait que la baisse de la proportion d'erreurs latentes est avantageuse pour la sécurité, car la forte couverture d'autodétection permet de faire intervenir la réparation (pour le Duplex-1) et la stratégie de compensation d'erreur par commutation (pour le Duplex-2). Par ailleurs, l'avantage du Duplex-2 par rapport au Duplex-1 devient plus perceptible.

Tableau 4-8 : Valeurs de n et UAA pour les différentes architectures du coussin gonflable en fonction de d et l

$n_{si}=n$ pour le Simplex (τ_{reca}=inst.) ; $UAA_{si}=UAA$ pour le Simplex (τ_{reca}=inst.) ;
$n_{d1}=n$ pour le Duplex (strat. 1) ; $UAA_{d1}=UAA$ pour le Duplex (strat. 1) ;
$n_{d2}=n$ pour le Duplex (strat. 2) ; $UAA_{d2}=UAA$ pour le Duplex (strat. 2) ;
$n_t=n$ pour le TMR; $UAA_t=UAA$ pour le TMR;
τ_{amr}=500h ; τ_{fvoy}=1h ; $\lambda_{acc}=3*10^{-4}$/h ; τ_{rmu}=0,1h ; cd=0,999 ; $\lambda_t=10^{-4}$/h ; $\lambda_p=10^{-5}$/h

d		l 0,0005	0,005	**0,05**	0,5
0,9	n_{si} [ppm]	5091,4	5069,7	4852,5	2680,0
	n_{d1} [ppm]	10,8	10,7	10,3	5,9
	n_{d2} [ppm]	10,5	10,5	10,0	5,7
	n_t [ppm]	0,03	0,03	0,08	5,71
	n_{si}/n_{d1}	471,4	473,8	471,1	454,2
	n_{si}/n_{d2}	484,2	484,1	483,6	473,2
	n_{si}/n_t	181835,7	181060,7	58463,9	469,5
0,99	n_{si} [ppm]	510,6	509,5	498,4	387,4
	n_{d1} [ppm]	1,3	1,3	1,3	1,1
	n_{d2} [ppm]	1,1	1,1	1,0	0,9
	n_t [ppm]	0,03	0,03	0,08	5,71
	n_{si}/n_{d1}	392,8	391,9	383,4	352,2
	n_{si}/n_{d2}	485,4	484,8	479,2	416,6
	n_{si}/n_t	18235,7	18196,4	6004,8	67,9
0,9	UAA_{si}	3,12E-06	2,31E-05	2,23E-04	2,22E-03
	UAA_{d1}	4,00E-06	4,04E-06	4,38E-06	7,86E-06
	UAA_{d2}	2,20E-06	2,24E-06	2,63E-06	6,52E-06
	UAA_t	3,07E-10	3,07E-08	3,06E-06	2,94E-04
	UAA_{si}/UAA_{d1}	0,8	5,7	50,9	282,3
	UAA_{si}/UAA_{d2}	1,4	10,3	84,9	340,5
	UAA_{si}/UAA_t	10163,3	753,1	73,0	7,6
0,99	UAA_{si}	3,12E-06	2,31E-05	2,23E-04	2,22E-03
	UAA_{d1}	2,20E-06	2,25E-06	2,68E-06	7,02E-06
	UAA_{d2}	2,25E-07	2,73E-07	7,51E-07	5,54E-06
	UAA_t	3,07E-10	3,07E-08	3,06E-06	2,94E-04
	UAA_{si}/UAA_{d1}	1,5	10,3	83,3	316,3
	UAA_{si}/UAA_{d2}	14,3	85,1	297,2	400,8
	UAA_{si}/UAA_t	10456,0	756,0	73,0	7,6

Concernant l'architecture TMR, nous constatons une grande amélioration de la sécurité pour des faibles proportions d'erreurs latentes. Cela est logique, car

l'architecture TMR est avantageuse lorsque la proportion d'erreurs se propageant du calculateur au voteur est grande. La couverture d'autodétection d n'intervient pas dans le modèle TMR alors que celle-ci est un paramètre très sensible dans les autres architectures. C'est pour cela que l'avantage du TMR augmente lorsque d baisse. Lorsque la proportion d'erreurs latentes et la couverture d'autodétection sont relativement élevées (l = 0,5 et d = 0,99), le TMR devient plus mauvais que les architectures Duplex.

Concernant l'indisponibilité avant accident *UAA*, l'interprétation des résultats est la suivante : pour une valeur de d donnée, l'augmentation de l induit un plus fort avantage pour les architectures Duplex dont le comparateur permet de détecter les erreurs latentes. Lorsque ces erreurs sont dues à des fautes transitoires, le recouvrement global permet la restauration du système ; en cas de faute permanente, la signalisation d'erreur donne la possibilité de réparer le système.

On note que, pour l = 0,0005 et d = 0,9, le *UAA* du Duplex est pire que celui du Simplex. Ici se traduit le fait que deux calculateurs peuvent présenter des erreurs propagées et qu'un nombre trop important de ces erreurs n'est pas détecté ce qui amène à la défaillance catastrophique (c'est-à-dire, le gonflement intempestif). Il est évident que l'augmentation de d induit une amélioration de *UAA* pour les architectures Duplex, notamment pour le Duplex à compensation d'erreur. Concernant le TMR, on vérifie que la baisse de l (c'est-à-dire, l'augmentation de la proportion d'erreurs propagées) procure un plus grand avantage pour le TMR.

IV.3 Analyse de la direction électronique

La direction est un système qui est sollicité en permanence, une défaillance du calculateur électronique aurait alors immédiatement des conséquences graves. De manière analogue au coussin gonflable, la réparation d'une telle défaillance n'est pas considérée. Par contre, contrairement au coussin gonflable, la situation où le calculateur est défaillant mais non sollicité (défaillance potentiellement catastrophique), n'existe pas ici. C'est pour cela que l'on ne peut pas distinguer entre sécurité et fiabilité du système. Cependant, nous continuerons à utiliser le vocable "sécurité" pour souligner le caractère catastrophique des défaillances. La *Figure 4-11* illustre le modèle fonctionnel de la direction électronique.

Dans le cas de la direction électronique, une architecture du type Simplex n'a, a priori, que peu d'intérêt, car une défaillance due à une faute permanente induirait l'interruption immédiate et totale de cette fonction indispensable, les conséquences seraient en général catastrophiques. D'une façon qualitative, on peut dire qu'une éventuelle signalisation d'une telle erreur n'améliore en rien cette situation. C'est pour cela que seulement les résultats de l'architecture Simplex simplifiée ("Direction-Simplex-i") sont présentés dans le paragraphe IV.3.1. Lorsque l'on

dispose d'une architecture Duplex, on peut profiter de la redondance des processeurs, par conséquence nous présentons dans le paragraphe IV.3.2 les résultats obtenus pour la "Direction-Duplex-2". L'architecture TMR pour la direction électronique est traitée dans le paragraphe IV.3.3. Une comparaison plus approfondie des architectures Simplex, Duplex et TMR est présentée dans le paragraphe IV.3.4. Nous notons que l'accident du véhicule n'a pas d'influence sur le modèle fonctionnel de la direction et, par conséquent, il n'est pas pris en compte dans les différents modèles globaux de la direction.

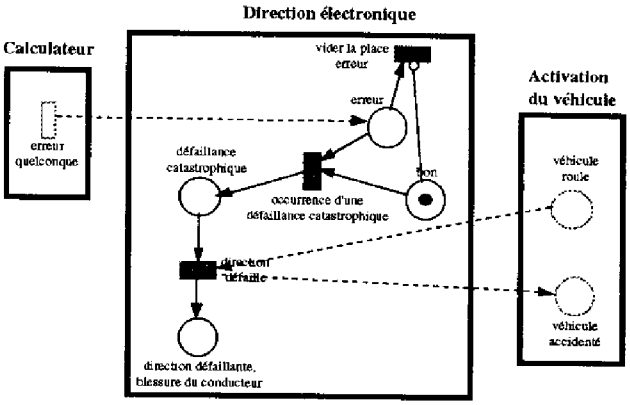


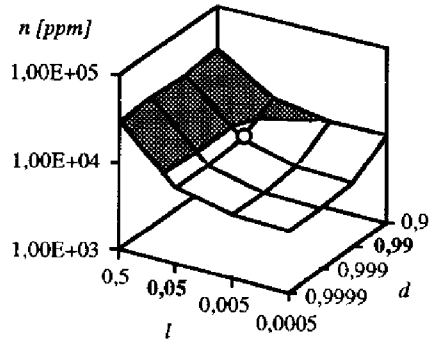
Figure 4-11 : Modèle fonctionnel de la direction électronique

IV.3.1 Architecture Simplex

L'architecture Simplex ne procure pas de compensation d'erreur, c'est pour cela qu'elle doit posséder une autodétection très puissante permettant ainsi de détecter et recouvrir la plus grande part des fautes transitoires. Par conséquent, nous analysons dans ce sous-paragraphe n en fonction de la couverture d'autodétection d et de la proportion d'erreurs latentes l. Lors des études de la sécurité ponctuelle de l'architecture Direction-Simplex, nous vérifions l'évolution exponentielle de la sécurité. La *Figure 4-12* trace les valeurs obtenues pour n en fonction de d et l. Nous vérifions que n baisse lors d'une augmentation de d et d'une baisse de l (on note cependant, que n est très élevé par rapport à l'objectif $n = 1$ *ppm*).

Les résultats obtenus peuvent être comparés avec ceux du Coussin-Simplex-i (cf. *Figure 4-6*). Nous constatons que la sécurité d'une architecture est liée à la fonctionnalité qui est réalisée. Dans le cas du coussin gonflable, la baisse de d et de l induit une plus forte baisse de n que dans le cas de la direction. Cela est logique dans la mesure où la détection d'erreur dans le calculateur permet d'éviter la défaillance catastrophique de la direction électronique en cas de faute transitoire uniquement. Au cas où la faute associée est permanente, le coussin gonflable se distingue de la

direction électronique par un mode de défaillance qui devient catastrophique seulement en cas de sollicitation du coussin.



τ_{amr}=500h ; τ_{fvoy}=1h ; τ_{rmu}=0,1h ; λ_l=10^{-4}/h ; λ_p=10^{-5}/h

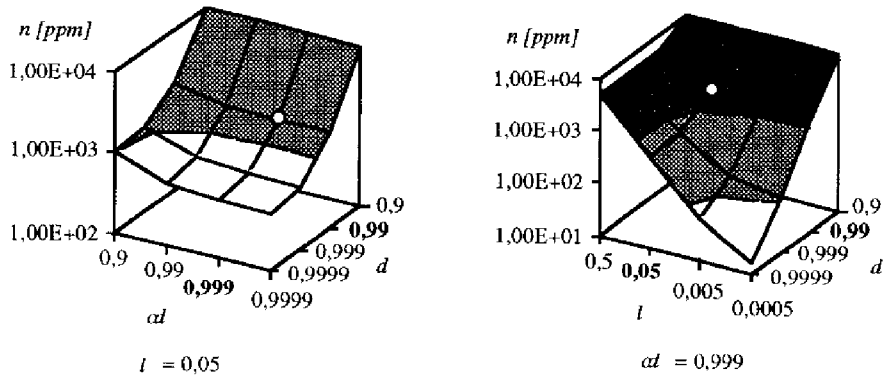
Figure 4-12 : Valeurs de n pour la Direction-Simplex-i en fonction de d et l

IV.3.2 Architecture Duplex

Nous rappelons que l'architecture Duplex permet de procurer soit une simple détection d'erreur soit une compensation d'erreur (dynamique). Dans le cas de la direction électronique, la stratégie de simple détection d'erreur ne revêt aucun intérêt, car cela consisterait à mettre deux calculateurs Simplex en parallèle, la défaillance de chaque calculateur pouvant induire une défaillance catastrophique de la direction. Par conséquent, on étudiera seulement l'architecture Duplex permettant une compensation des erreurs qui sont dues à des fautes détectées localement par l'autodétection de chaque calculateur. La *Figure 4-13* illustre l'évolution de n en fonction des paramètres cd (couverture de détection par comparateur), d (couverture d'autodétection) et l (proportion d'erreurs latentes).

La surface tracée dans la partie gauche de la figure nous permet de vérifier que l'augmentation de d et de cd induisent une baisse de n. La partie droite de la figure illustre l'influence de l et de d sur la sécurité. Nous vérifions que l'augmentation de d et la baisse de l induisent une baisse de n. On note cependant que l'objectif $n = 1$ *ppm* n'est jamais atteint pour toute la gamme de cd, d et de l.

De façon analogue au paragraphe précédent, nous pouvons comparer les résultats obtenus pour la Direction-Duplex-2 avec ceux du Coussin-Duplex-2 (cf. *Figure 4-7*). On peut tirer des conclusions tout à fait analogues à celles pour l'architecture Simplex.

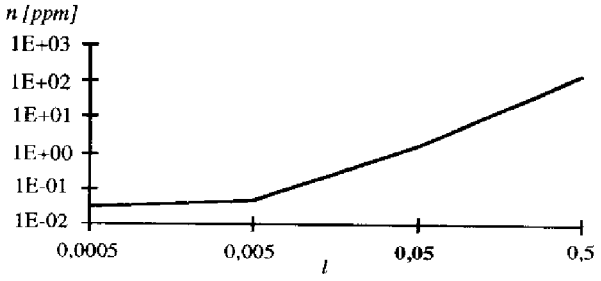


τ_{amr}=500h ; τ_{tvoy}=1h ; τ_{rmu}=0,1h ; λ_t=10^{-4}/h ; λ_p=10^{-5}/h

Figure 4-13 : Valeurs de n pour la Direction-Duplex-2 en fonction de cd, d et l

IV.3.3 Architecture TMR

L'architecture TMR ne possède pas d'autodétection au niveau de chaque calculateur et le voteur est supposé parfait. Par conséquent, nous étudions dans ce paragraphe l'évolution de n en fonction du paramètre l seulement. La *Figure 4-14* illustre l'évolution de n en fonction de l. Nous vérifions que l'augmentation de l entraîne une augmentation de n, l'objectif n = 1 *ppm* étant atteint pour $l < 0{,}05$.



τ_{amr}=500h ; τ_{tvoy}=1h; τ_{rmu}=0,1h ; λ_t=10^{-4}/h ; λ_p=10^{-5}/h

Figure 4-14 : Valeurs de n pour la Direction-TMR en fonction de l

De façon analogue aux deux paragraphes précédents, nous comparons la fonction de direction électronique avec la fonction de coussin gonflable sur l'architecture TMR (cf. *Figure 4-10*). On remarque dans ce cas que la sécurité procurée par l'architecture TMR est très similaire pour les deux fonctions. Cependant, on observe que l'augmentation de la proportion des erreurs latentes est plus nuisible à la sécurité de la direction électronique. Dans ce cas, la probabilité de l'occurrence de deux erreurs

latentes dues à des fautes permanentes et, par conséquent, d'une défaillance catastrophique de la direction électronique devient plus forte.

IV.3.4 Comparaison des architectures Simplex, Duplex et TMR

Dans ce paragraphe, nous allons comparer les résultats obtenus pour les architectures Direction-Simplex-i, Direction-Duplex-2 et Direction-TMR. Le *Tableau 4-9* montre les résultats obtenus pour différentes valeurs de d et de l, les autres paramètres étant fixés sur leurs valeurs nominales.

On constate tout d'abord le faible apport en sécurité de l'architecture Duplex par rapport à l'architecture Simplex. Cet apport devient plus significatif pour une augmentation de la couverture d'autodétection. Par contre, l'architecture TMR apporte beaucoup en matière de sécurité lorsque la proportion d'erreurs latentes est inférieure à 0,05. On vérifie également que la baisse de la couverture d'autodétection d induit un avantage plus fort pour le TMR (cela étant dû simplement à la dégradation des architectures Simplex et Duplex).

Tableau 4-9 : Valeurs de n pour les différentes architectures de la direction électronique en fonction de d et l

$n_{si}=n$ pour le Simplex (τ_{reca}=inst.) ; $n_{d2}=n$ pour le Duplex (strat. 2) ; $n_t=n$ pour le TMR ; τ_{amr}=500h ; τ_{fvoy}=1h ; τ_{rmu}=0,1h ; cd=0,999 ; $\lambda_t=10^{-4}$/h ; $\lambda_p=10^{-5}$/h

d		l 0,0005	0,005	**0,05**	0,5
0,9	n_{si}	9972,50	10173,00	12175,40	31977,60
	n_{d2}	10939,30	10935,30	10895,20	10494,50
	n_t	0,03	0,05	1,62	157,10
	n_{si}/n_{d2}	0,91	0,93	1,12	3,05
	n_{si}/n_t	329125,41	223582,42	7515,22	203,55
0,99	n_{si}	5509,60	5731,10	7943,40	29797,10
	n_{d2}	1103,90	1144,40	1548,80	5584,30
	n_t	0,03	0,05	1,62	157,10
	n_{si}/n_{d2}	4,99	5,01	5,13	5,34
	n_{si}/n_t	181834,98	125958,24	4903,03	189,67

Conclusion

Dans ce chapitre, nous avons montré l'utilité des modèles markoviens à l'évaluation quantitative de la sûreté de fonctionnement. Notre choix a été de mesurer la sécurité par le biais du nombre de systèmes catastrophiquement défaillants au bout d'un certain temps (n) ainsi que l'indisponibilité avant accident (UAA). L'utilisation de la mesure n nous a permis d'exprimer la sécurité des systèmes étudiés d'une

manière compréhensible et facilement interprétable. Concernant la mesure *UAA*, elle nous a été utile seulement pour l'étude des systèmes à plusieurs degrés de défaillance, tels que le coussin gonflable, et a permis dans ce cas de compléter l'interprétation des résultats dérivés de la mesure de n.

Les études effectuées dans ce chapitre ont permis non seulement de vérifier des phénomènes auxquels on pouvait s'attendre, mais aussi de révéler d'autres effets intéressants et de dégager des conclusions pour la réalisation des systèmes modélisés. Une conclusion importante de nos études concerne la faible influence de la durée de l'intervalle de maintenance sur la sécurité des systèmes électroniques embarqués sur l'automobile. Cela pourrait permettre aux constructeurs de rallonger les intervalles de maintenance, dans le but de réduire le coût de maintenance pour le client. Nous avons également montré qu'une politique de signalisation des erreurs dues à des fautes transitoires (dans le cas du Simplex) ne revêt aucun intérêt particulier du point de vue de la sécurité du conducteur.

Concernant la réalisation d'un système de coussin gonflable avec une architecture Simplex, nous constatons que les objectifs de sûreté de fonctionnement ne peuvent pas être atteints avec les valeurs nominales des paramètres. Il faudrait soit augmenter sensiblement la couverture d'autodétection d'erreurs (d = 0,99999, par exemple, ce qui semble peu réaliste avec une architecture Simplex), soit diminuer fortement le taux de défaillances transitoires par un blindage conséquent (λ_t de l'ordre de 10^{-7}/h). Par ailleurs, le gonflement intempestif pouvant être important par rapport au non-gonflement lors d'une sollicitation (n_{gi}/n_{ng} est de l'ordre de 300 pour les valeurs nominales des paramètres), il faudrait prendre des précautions afin de diminuer la probabilité de propagation d'erreurs. Par exemple, la société Siemens Automotive S.A. vend un système où la commande électronique de l'actionneur pyrotechnique est autorisée seulement au-delà d'une certaine décélération du véhicule. La décélération est mesurée par le biais du déplacement horizontal d'une bille sur un rail renfermé. Au-delà d'un déplacement minimal, le calculateur est autorisé à commander l'actionneur pyrotechnique. La sécurité de l'architecture Coussin-Simplex passerait donc par un blindage conséquent et une autodétection très puissante pour diminuer l'impact des fautes transitoires en vie opérationnelle ainsi que par l'évitement des fautes permanentes lors de la phase de conception (et production) du système.

L'architecture Duplex permet de détecter un nombre plus important d'erreurs et ainsi d'atteindre l'objectif de sécurité fixé pour le coussin gonflable. Parmi les deux stratégies du Duplex, l'apport de la stratégie de compensation d'erreur en matière de sécurité est relativement faible, ce qui signifie qu'une architecture tolérant une faute est sans grand intérêt dans le cas du coussin gonflable. Ce résultat est confirmé par le fait que l'architecture TMR ne procure pas un avantage significatif par rapport à l'architecture Duplex, elle peut même — lorsque la proportion d'erreurs latentes est

très élevée — être désavantageuse par rapport à celle-ci, non seulement au niveau de la complexité, mais également au niveau de la sûreté de fonctionnement. Une conclusion importante de nos études est alors le grand intérêt d'une architecture du type Duplex pour le système du coussin gonflable, celle-ci permettant d'atteindre le niveau de sécurité exigé à un prix plus faible que l'architecture TMR.

En ce qui concerne la direction électronique, par contre, nos résultats confirment la nécessité d'une architecture disposant d'une compensation d'erreur. L'intérêt d'une architecture du type TMR est évident, sous condition que la plus grande part des erreurs dans les calculateurs se propagent à l'entrée du voteur au lieu de rester latentes. Une comparaison de la sécurité des deux fonctions (le coussin gonflable et la direction électronique) sur la même architecture nous a montré notamment que les architectures Simplex et Duplex procurent un niveau de sécurité plus élevé pour le coussin gonflable, tandis que l'architecture TMR offre à peu près le même niveau de sécurité pour les deux fonctions.

Conclusion générale

Dans ce mémoire, nous avons traité de la sûreté de fonctionnement des systèmes embarqués sur l'automobile. L'évolution de l'automobile et, plus particulièrement, de ses systèmes embarqués est caractérisée par l'introduction de calculateurs d'une complexité matérielle et logicielle de plus en plus grande. Ces calculateurs remplacent partiellement les parties mécaniques, procurant ainsi un découplage mécanique entre la commande du conducteur et l'action effectuée. Les démarches de conception visent notamment à s'assurer que la sûreté de fonctionnement des systèmes embarqués sur l'automobile ne soit pas dégradée par l'introduction de la nouvelle technologie, d'autant plus qu'il s'agit de maîtriser leur complexité sans cesse croissante ainsi que les contraintes d'environnement sévères. Les systèmes embarqués étant à l'heure actuelle conçus d'une façon fédérée (c'est-à-dire, spécifique par rapport à leur fonctionnalité), la tendance est de procéder à une intégration des systèmes afin de réduire leur coût total tout en améliorant leur fonctionnement global et, en particulier, leur sûreté de fonctionnement. L'échange d'informations entre les systèmes jouant un rôle primordial lors de l'intégration fonctionnelle, nous avons présenté et analysé différents réseaux multiplexés du domaine automobile dont le but est d'assurer la communication entre plusieurs systèmes embarqués. Nous avons conclu que les réseaux actuellement offerts sur le marché de l'automobile ne remplissent qu'en partie les objectifs de sûreté de fonctionnement souhaités et que d'autres réseaux devraient être étudiés.

Il est évident que l'intégration nécessite une nouvelle approche globale en ce qui concerne la conception de l'architecture matérielle-logicielle des systèmes embarqués ainsi que les procédures de développement associées. Cela signifie également la nécessité d'une démarche globale à la sûreté de fonctionnement. L'approche globale à la conception des systèmes embarqués est illustrée à travers un spectre d'architectures décrivant plusieurs solutions entre une architecture fédérée à un bout du spectre et une architecture intégrée à l'autre. En se basant sur ce spectre, nous avons pu comparer différentes architectures qualitativement par rapport à plusieurs critères, avec un effet de loupe sur l'aspect coût. D'une manière générale, nous avons conclu que l'intégration peut être intéressante au niveau coût et sûreté de fonctionnement seulement au-delà d'une certaine richesse fonctionnelle.

La comparaison qualitative des architectures au niveau de leur sûreté de fonctionnement est approfondie par une méthode (quantitative) de prévision des fautes. Cette méthode pourrait se placer au sein d'une procédure de spécification et de développement des systèmes embarqués. Elle a été développée dans le but d'effectuer

une évaluation probabiliste de la sûreté de fonctionnement des systèmes embarqués et de pouvoir guider le choix de l'architecture "optimale" pour une fonctionnalité donnée. La méthode de modélisation suivie permet de décrire l'alternance entre service correct et service incorrect en tenant compte des différentes possibilités d'architecture. Elle consiste à modéliser le comportement par des processus stochastiques (markoviens) en passant par une modélisation modulaire en Réseaux de Petri Stochastiques Généralisés. L'avantage principal de la méthode modulaire proposée est de faciliter la maîtrise de la complexité et la compréhension des modèles markoviens générés. Son originalité réside dans le fait de décrire les aspects fonctionnels indépendamment de l'architecture ainsi que de l'activation du véhicule. Nous pouvons définir les mesures de sûreté de fonctionnement seulement à l'aide d'un modèle fonctionnel dont les changements d'état sont dictés par un modèle du calculateur sous-jacent. Cela signifie que plusieurs architectures du calculateur peuvent être modélisées sans changer le modèle fonctionnel et le modèle de l'activation du véhicule. Dans ce mémoire, cette méthode est appliquée à deux fonctions — à savoir le coussin gonflable et la direction électronique — qui sont implantées sur certaines architectures du spectre. Lors de la quantification des mesures de sûreté de fonctionnement à l'aide des modèles markoviens générés, nous avons pu vérifier que les chiffres absolus sont à interpréter avec prudence, car les modèles sont basés sur des hypothèses simplificatrices. Les résultats issus de la comparaison, par contre, nous semblent aptes à guider le choix d'une architecture pour une fonction donnée. Nous avons ainsi illustré l'utilité des modèles markoviens à l'évaluation quantitative de la sûreté de fonctionnement et au choix d'architecture.

Il est évident que les différents aspects traités dans ce mémoire peuvent être étendus par d'autres travaux de recherche. Par exemple, l'étude des réseaux multiplexés devrait être complétée par l'analyse des réseaux embarqués notamment du domaine avionique ou militaire. D'un point de vue industriel, il serait intéressant d'approfondir l'étude du coût en améliorant le modèle du coût proposé et en développant des prototypes associés aux architectures analysées.

Quant au travail de modélisation effectué, nous pensons qu'il peut être aisément étendu grâce à la modularité de l'approche. Afin de couvrir entièrement le spectre d'architectures développé, il faudrait modéliser l'aspect communication inter-processeurs (surtout multiplexée). De plus, hormis les deux fonctions considérées dans ce mémoire, d'autres fonctions devraient être modélisées. Sachant que plusieurs fonctions peuvent cohabiter sur la même architecture, il serait nécessaire d'étudier exactement les façons de lier plusieurs modèles fonctionnels simultanément avec les modèles d'architecture et d'activation du véhicule. Notons que la méthode de modélisation modulaire, notamment la séparation en trois points de vue, est un choix qui a demandé de nombreuses itérations et mises au point. Ce choix est "personnel"

dans la mesure où l'on aurait pu séparer les modèles différemment et modéliser le comportement des systèmes d'une autre manière. Une proposition serait d'étudier la généralité de la méthode, c'est-à-dire son applicabilité à des domaines différents de l'automobile. En ce qui concerne l'utilisation de l'outil SURF-2, nous avons constaté que l'édition graphique offerte est avantageuse pour la saisie de réseaux relativement peu complexes, mais qu'elle devrait être complétée par une possibilité de description modulaire afin de pouvoir maîtriser des réseaux complexes et de mieux soutenir notre démarche.

Lors de l'analyse des modèles, nous avons observé que la mesure de la sécurité des systèmes basée sur les modèles développés permet d'évaluer la sûreté de fonctionnement et de guider le choix d'architecture, tandis que la mesure de la disponibilité avant accident est difficile à interpréter. Il aurait pu être intéressant de modéliser les systèmes de façon à pouvoir mesurer la disponibilité des systèmes embarqués, en tenant compte de différentes politiques de maintenance. Afin d'élargir le travail d'analyse, on pourrait supposer que --- dû à la grande complexité des systèmes — la maintenance et la réparation des systèmes électroniques embarqués ne sont pas parfaites (induisant ainsi une plus longue présence des fautes permanentes). Les résultats présentés pourraient être également enrichis par la prise en compte d'autres paramètres tels que le temps de détection (qui dans nos modèles est supposé égal à zéro) ou bien le temps de propagation d'erreur.

L'automobile du futur contiendra de plus en plus de systèmes électroniques procurant un découplage mécanique entre la commande du conducteur et l'action effectuée. La fonctionnalité de ces systèmes étant souvent liée à la sécurité des passagers et de l'environnement (par exemple, les systèmes d'anti-collision ou de contrôle de la dynamique du véhicule), il faudra notamment garantir leur sûreté de fonctionnement. Un aspect très important et nouveau est que ces fonctions concernent non pas une partie spécifique de l'automobile, mais affectent en général son ensemble. Par exemple, le système de contrôle de la dynamique du véhicule fait intervenir à la fois le freinage, le contrôle moteur et le contrôle de la transmission. Il sera alors essentiel de concevoir l'ensemble des systèmes embarqués d'une façon homogène et, par conséquent, de faire une démarche globale à leur sûreté de fonctionnement. En proposant une méthode d'évaluation de la sûreté de fonctionnement des systèmes embarqués, nous espérons avoir contribué à la mise en place d'une telle procédure de spécification et de développement des systèmes embarqués.

Annexe

Définition des mesures

Lorsque l'on évalue la sûreté de fonctionnement d'un système en se basant sur un modèle markovien, on peut choisir entre un grand nombre de mesures. Dans cette annexe, nous approfondissons les définitions générales des mesures de sûreté de fonctionnement (voir chapitre I) d'un point de vue mathématique. Nous allons focaliser sur les mesures qui sont utilisées le plus couramment dans la littérature et qui sont aussi implantées dans l'outil SURF-2 [Bachmann 1996]. Les paragraphes A.1 à A.4 donnent les définitions mathématiques des mesures de fiabilité, (in-)disponibilité, maintenabilité et sécurité. Dans le paragraphe A.5, nous présentons les mesures des différents temps moyens, tels que le temps moyen entre deux défaillances successives. L'outil SURF-2 permet également de calculer des mesures de type revenu (voir aussi chapitre III), lesquelles sont traitées dans le paragraphe A.6. Ces mesures permettent par exemple d'évaluer le temps moyen passé dans un certain état du système ou bien de compter le nombre de tirs d'une certaine transition. La définition des revenus permet à son tour de mesurer la disponibilité avant défaillance catastrophique, en se basant sur le temps moyen passé dans les états sûrs ainsi que sur le temps moyen passé dans les états propres (voir également chapitre III). La définition exacte de la disponibilité avant défaillance catastrophique est donnée au paragraphe A.7.

A.1 Mesure de la fiabilité

Au paragraphe I.3.3, nous avons défini la fiabilité comme l'aptitude d'un système à délivrer le service attendu pendant une durée donnée. La mesure de la fiabilité, notée $R(t)$ (anglais : reliability), est alors la probabilité qu'un système accomplisse sa fonction pendant l'intervalle de temps $[0,t]$.

- $R(t)$ = P (système non défaillant sur $[0,t]$)
- $R(t) = P_p(0) * e^{\Lambda_{pp} t} * (1_p)^T$
- p est le sous-ensemble d'états propres (np = nombre d'états propres)
- $P_p(0)$ le vecteur de probabilités initiales associées à p
- Λ_{pp} est la matrice des taux de transition associés à p

- (1_p) est un vecteur de dimension *np*, tous ses éléments étant égaux à 1

A.2 Mesure de la disponibilité et de l'indisponibilité

Au paragraphe I.3.3, nous avons défini la disponibilité comme l'aptitude d'un système à être en état de délivrer le service attendu à un instant donné, par rapport à l'alternance "service correct — service incorrect". La mesure de la disponibilité, notée $A(t)$ (anglais : availability), est alors la probabilité qu'un système accomplisse sa fonction à un instant donné.

- $A(t)$ = P (système non défaillant à l'instant t)
- $A(t) = P(0) * e^{\Lambda t} * U^{T}$
- $P(0)$ est le vecteur de probabilités initiales
- Λ est la matrice des taux de transition
- U est un vecteur dont les éléments correspondants aux états *propres* sont égaux à 1, les autres éléments étant égaux à 0.

L'indisponibilité ponctuelle, notée $\bar{A}(t)$, est simplement définie par l'équation :

$$\bar{A}(t) = 1 - A(t)$$

L'outil SURF-2 permet également de calculer directement les valeurs asymptotiques de la disponibilité et de l'indisponibilité, sans devoir passer par une analyse ponctuelle.

A.3 Mesure de la maintenabilité

Au paragraphe I.3.3, nous avons défini la maintenabilité comme la facilité de préserver ou améliorer l'aptitude du système à délivrer le service attendu. La mesure de la maintenabilité, notée $M(t)$, est alors la probabilité que la maintenance du système soit achevée au temps t après avoir défailli au temps 0 :

- $M(t)$ = P (système est réparé sur $[0,t]$) après avoir défailli à l'instant 0
- $M(t) = 1 - P_i(0) * e^{\Lambda_{ii} t} * (1_i)^{T}$
- i est le sous-ensemble d'états *impropres* (*ni* = nombre d'états impropres)
- $P_i(0)$ est le vecteur de probabilités initiales associées à i
- Λ_{ii} est la matrice des taux de transition associés à i
- (1_i) est un vecteur de longueur *ni*, tous ses éléments étant égaux à 1

A.4 Mesure de la sécurité

Au paragraphe I.3.3, nous avons défini la sécurité(-innocuité) comme l'aptitude d'un système à éviter de faire apparaître des défaillances catastrophiques. La mesure de la sécurité, notée $S(t)$, est alors la probabilité que le système évite de faire apparaître des événements catastrophiques sur l'intervalle de temps $[0,t]$:

- $S(t)$ = P (système ne produit pas de défaillance catastrophique sur $[0,t]$)
- $S(t) = P_s(0) * e^{\Lambda_{ss}t} * (1_s)^T$
- s est les sous-ensemble d'états *sûrs* (ns = nombre d'états sûrs)
- $P_s(0)$ est le vecteur de probabilités initiales associées à s
- Λ_{ss} est la matrice des taux de transition associés à s
- (1_i) est un vecteur de longueur ns, tous ses éléments étant égaux à 1

A.5 Mesure des temps moyens

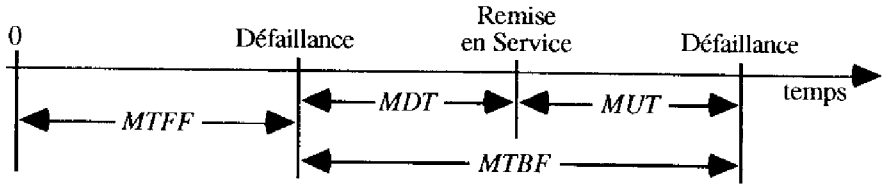
L'outil SURF-2 permet de calculer différents temps moyens :

- *MTFF* (Mean Time to First Failure) :
  Durée moyenne de fonctionnement du système avant la première défaillance
  $MTFF = -P_p(0) * (\Lambda_{pp})^{-1} * (1_p)^T$
- *MTTF* (Mean Time To Failure) :
  Durée moyenne de fonctionnement du système avant la prochaine défaillance en régime stationnaire
  $$MTTF = \int_0^\infty R(t)\ dt = \frac{P_p(\infty)}{P_p(\infty) * 1_p} * (\Lambda_{pp})^{-1} * (1_p)^T$$
- *MUT* (Mean Up Time) :
  Durée moyenne de fonctionnement après réparation en régime stationnaire
  $$MUT = \frac{P_p(\infty) * (1_p)^T}{P_p(\infty) * \Lambda_{pi} * (1_i)^T}$$
  avec Λ_{pi} = matrice des taux de transition des états propres vers les états impropres
- *MDT* (Mean Down Time) :
  Durée moyenne d'indisponibilité consécutive à la dernière défaillance en régime stationnaire
  $$MDT = \frac{P_i(\infty) * (1_i)^T}{P_i(\infty) * \Lambda_{ip} * (1_p)^T}$$

avec Λ_{ip} = matrice des taux de transition des états impropres vers les états propres

- *MTBF* (Mean Time Between Failure) :
  Durée moyenne entre deux défaillances consécutives d'un système réparé en régime stationnaire
  $MTBF = MUT + MDT$

Le schéma suivant illustre les relations entre les différents temps moyens :



A.6 Mesure des revenus

La structure de revenu qui est superposée sur la chaîne de Markov permet de définir deux types de *coût* : **a)** Le coût associé à la présence dans certains états i, en affectant des *taux de récompense* (par exemple, en Deutschmark/heure) à ces états. On note $Y=[y_i]$ le vecteur des taux de récompense. **b)** Le coût associé au franchissement de certaines transitions ij (entre états i et j), en affectant des *bonifications* (par exemple, en Deutschmark) à ces transitions. On note $B=[b_{ij}]$ le vecteur des bonifications. A partir de ces définitions de base, on introduit la notion de récompense cumulée sur l'intervalle $[0,t]$:

$$W(t,\gamma) = \sum_i y_i \int_0^t e^{-\gamma * u} * V_i(u)du + \sum_i \sum_{j \neq i} b_{ij} \int_0^t e^{-\gamma * u} * S_{ij}(u)du$$

- $V_i(u) = 1$ si l'on se trouve dans les états i à l'instant u (sinon $V_i(u) = 0$)
- $S_{ij}(u) = 1$ si l'on franchit les transitions ij dans l'intervalle $[u,u+\delta u]$ (sinon $S_{ij}(u) = 0$)
- γ = taux d'escompte

On peut ainsi obtenir plusieurs sortes de *mesures économiques* :

❑ L'espérance de la récompense cumulée sur l'intervalle $[0,t]$:
$EW(t,\gamma) = E[W(t,\gamma)]$

$$= \sum_i [y_i \int_0^t p_i(u) * e^{-\gamma * u} * V_i(u)du + \sum_{j \neq i} b_{ij} \int_o^t p_i(u) * \lambda_{ij} * e^{-\gamma * u} * S_{ij}(u)du]$$

- $p_i(u)$ = probabilité d'être dans l'état i à l'instant u

• $p_i(u)*\lambda_{ij}$ = densité de probabilité de la transition ij dans l'intervalle $[u,u+\delta u]$

- L'espérance de la récompense asymptotique :
  $EW(\gamma) = \lim_{t->\infty} EW(\gamma,t) = \lim_{t->\infty} E[W(\gamma,t)]$
- Le taux de récompense asymptotique :
  $r = \lim_{t->\infty} \frac{\delta EW(0,t)}{\delta t} = \lim_{t->\infty} \frac{\delta E[W(0,t)]}{\delta t}$
- L'espérance de la récompense moyenne :
  $EW_p(\gamma = 0) = -P_p(0)*(\Lambda_{pp})^{-1}*(R_p)^{T}$ avec $r_i = y_i + \sum_{j=1, j\neq i}^{n} b_{ij}$

A.7 Mesure de la disponibilité avant défaillance catastrophique

L'outil SURF-2 permet également de mesurer la disponibilité avant qu'une défaillance catastrophique ne survienne. Cette mesure est notée AC (anglais : Availability before Catastrophic failure), sa définition précise est donnée dans [Arlat & Laprie 1985].

L'ensemble des états d'accomplissement du service correct est noté A et l'ensemble des états de défaillance bénigne est noté B. Ensuite, on regroupe A et B sous l'appellation S (états sûrs). A partir de ces notations, on peut définir les temps moyens de séjour suivants :

- MAT (Mean Available Time) = temps moyen de séjour dans A avant la première défaillance catastrophique :
  $MAT = -P_p(0)*(\Lambda_{pp})^{-1}*(1_A \quad 0_B)^{T}$
- MBT (Mean Benign Time) = temps moyen de séjour dans B avant la première défaillance catastrophique :
  $MBT = -P_p(0)*(\Lambda_{pp})^{-1}*(0_A \quad 1_B)^{T}$
- MST (Mean Safe Time) = temps moyen de séjour dans S avant la première défaillance catastrophique :
  $MST = MAT + MBT$

Le calcul de AC est défini comme :

$$AC = \frac{MAT}{MAT + MBT}$$

Le calcul de l'indisponibilité avant défaillance catastrophique, notée UAC (anglais : UnAvailability before Catastrophic failure), se calcule simplement par l'équation suivante :

$$UAC = 1 - AC = \frac{MBT}{MAT + MBT}$$

Bibliographie

[AFNOR 1990] AFNOR : "Véhicules routiers — Transmission de données", R 13-708, Association Française de Normalisation (AFNOR), Décembre 1990

[Ammar *et al.* 1987] H.H. Ammar, Y.F. Huang et R.-W. Liu : "Hierarchical Models for Systems Reliability, Maintainability, and Availability", *Transactions on Circuits and Systems*, Vol.CAS-34, N°6, pp. 629-638, IEEE, June 1987

[Aono 1988] S. Aono : "The next step in automotive electronic control", *International Congress on Transportation Electronics (Convergence)*, pp. 83-89, IEEE, October 1988

[Arai 1990] H. Arai : "New roles of automotive electronics", *22. International Symposium on Automotive Technology & Automation (ISATA)*, Vol.1, pp. 61-77, Automotive Automation Limited, England, May 1990

[Arlat 1988] J. Arlat : "Méthodes et outils pour l'évaluation de la sûreté de fonctionnement des systèmes informatiques tolérant les fautes", *Technique et Science Informatiques (T.S.I.)*, Vol.7, N°4, pp. 345-357, AFCET-Bordas, 1988

[Arlat & Laprie 1985] J. Arlat et J.-C. Laprie : "On the Dependability Evaluation of High Safety Systems", *15th International Conference on Fault-Tolerant Computing (FTCS-15)*, pp. 318-323, IEEE, June 1985

[Atamna 1994] Y. Atamna : "Réseaux de Petri Temporisés Stochastiques Classiques et Bien Formés : Définition, Analyse et Applications aux Systèmes Distribués Temps Réel", Rapport N°94418 (Thèse de Doctorat), LAAS-CNRS, Toulouse, France, Octobre 1994

[Bachmann 1996] S. Bachmann : "SURF-2 : Manuel d'utilisateur", LAAS-CNRS, Toulouse, France, Mars 1996

[Baranowski 1990] F. Baranowski : "Définition des objectifs de sécurité dans les transports terrestres", Rapport INRETS N°133, Institut National de Recherche sur les Transports et leur Sécurité (INRETS), Décembre 1990

[Behnke *et al.* 1993] H. Behnke, A. Peters et F. Thoma : "Die neuen Vierventil-Dieselmotoren von Mercedes-Benz", *Motortechnische Zeitschrift (MTZ)*, N°54, pp. 490-503, Oktober 1993

[Bell & Reinert 1993] R. Bell et D. Reinert : "Risk and system integrity concepts for safety-related control systems", *Microprocessors and Microsystems*, Vol.17, N°1, pp. 3-15, January 1993

[Béounes *et al.* 1993] C. Béounes, M. Aguéra, J. Arlat, S. Bachmann, C. Bourdeau, J.-E. Doucet, K. Kanoun, J.-C. Laprie, S. Metge, J. Moreira de Souza, D. Powell et P. Spiesser : "SURF-2 : A Program for Dependability Evaluation of Complex Hardware and Software Systems", *23. International Symposium on Fault-Tolerant Computing (FTCS-23)*, pp. 668-673, IEEE, June 1993

[Bergfried *et al.* 1990] D. Bergfried, B. Mattes et M. Rutz : "Electronic crash sensors for restraint systems", *International Congress on Transportation Electronics (Convergence)*, pp. 169-177, Society of Automotive Engineers (SAE), October 1990

[Bergfried *et al.* 1992] D. Bergfried, W. Nitschke et M. Rutz : "Airbag control modules - performance and reliability", *International Congress on Transportation Electronics (Convergence)*, pp. 155-162, Society of Automotive Engineers (SAE), October 1992

[Borrel 1996] M. Borrel : "Interactions entre composants matériel et logiciel de systèmes tolérants aux fautes — caractérisation, formalisation, modélisation, application à la sûreté de fonctionnement du CAUTRA", Rapport N°96.001 (Thèse de Doctorat), LAAS-CNRS, Toulouse, France, Janvier 1996

[Bosch 1990] Bosch : "CAN Specification, Version 1.2", Robert Bosch GmbH, February 1990

[Breitwieser *et al.* 1994] K.-H. Breitwieser, S. Krebs, U. Schmalohr et K. Steffens : "Motorsteuerung für die neuen Vierzylinder-Vierventilmotoren von Opel", *Motortechnische Zeitschrift (MTZ)*, N°55, pp. 272-279, Mai 1994

[Caumont *et al.* 1982] G. Caumont, J.-C. Laprie et D. Powell : "RHEA: a fault and damage-tolerant hierarchical communication support system for local area computing in aggressive environments", *3. International Conference on Distributed Computing Systems*, pp. 77-83, IEEE, October 1982

[Chen & Ervin 1992] K. Chen et R.D. Ervin : "Worldwide IVHS activities : a comparative overview", *International Congress on Transportation Electronics (Convergence)*, pp. 339-349, Society of Automotive Engineers (SAE), October 1992

[Ciardo & Trivedi 1993] G. Ciardo et K.S. Trivedi : "A decomposition approach for stochastic reward net models", *Performance Evaluation 18*, pp. 37-59, Elsevier Science Publishers B.V., North-Holland, 1993

[Dais & Unruh 1992a] S. Dais et J. Unruh : "Technisches Konzept des seriellen Bussystems CAN — Teil 1", *Automobiltechnische Zeitschrift (ATZ)*, N°94, pp. 66-77, Februar 1992

[Dais & Unruh 1992b] S. Dais et J. Unruh : "Technisches Konzept des seriellen Bussystems CAN — Teil 2", *Automobiltechnische Zeitschrift (ATZ)*, N°94, pp. 208-215, April 1992

[DEKRA 1992] DEKRA : "Technische Mängel an Kraftfahrzeugen 1991", ISSN 0721-7307, DEKRA-Fachschriftenreihe 45/92, DEKRA AG, Schulze-Delitzsch-Str., 70565 Stuttgart, Deutschland, 1992

[DO178 1992] DO178 : "Software considerations in airborne systems and equipment certification", D0-178A/ED-12A, Draft 7, Requirements and technical concepts for aviation, Washington, July 1992

[Dugan & Trivedi 1989] J.B. Dugan et K.S. Trivedi : "Coverage Modeling for Dependability Analysis of Fault-Tolerant Systems", *Transactions on Computers*, Vol.38, N°6, pp. 775-787, IEEE, June 1989

[Dugan *et al.* 1984] J.B. Dugan, K.S. Trivedi, R.M. Geist et V.F. Nicola : "Extended Stochastic Petri Nets : Applications and Analysis", *10th International Symposium on Computer Performance*, pp. 507-519, Elsevier Science Publishers B.V., North-Holland, December 1984

[Estève *et al.* 1995] D. Estève, A. Coustre et M. Garajedagui : "L'intégration des systèmes électroniques dans l'automobile du XXIe siècle", ISBN 2-85428-386-4, Cépaduès-Editions, 1995

[Fassnacht & Madden 1988] R.E. Fassnacht et W.H. Madden : "Chrysler Collision Detection Bus Interface, Integrated Circuit User Manual", Society of Automotive Engineers (SAE), N°880586, pp. 4.620-4.637, 1988

[Florin & Natkin 1978] G. Florin et S. Natkin : "Evaluation des performances d'un protocole de communication à l'aide des Réseaux de Petri et des Processus Stochastiques", *Journées AFCET Informatique : Multiprocesseurs et Multiordinateurs en Temps Réel*, pp. E1-E24, Centre National de la Recherche Scientifique (CNRS), Paris, Mai 1978

[Fohler 1993] G. Fohler : "Realizing Changes of Operational Modes with a Pre Run-Time Scheduled Hard Real-Time System", dans *Dependable Computing and Fault-Tolerant Systems*, pp. 287-300, ISBN 3-211-82458-8, Springer-Verlag, 1993

[Fontaine *et al.* 1989] H. Fontaine, G. Malaterre et P. Van Elslande : "Evaluation de l'efficacité potentielle des aides à la conduite", Société des Ingénieurs de l'Automobile (SIA), N°89046, pp. 227-233, Octobre 1989

[Fujiki *et al.* 1991] N. Fujiki, N. Hiwa, T. Isobe, T. Inoue et K. Akashi : "An evaluation of multiplexing system for automotive distributed control", Society of Automotive Engineers (SAE), N°910718, pp. 189-194, February 1991

[Geist & Trivedi 1990] R.M. Geist et K.S. Trivedi : "Reliability Estimation of Fault-Tolerant Systems : Tools and Techniques", *Computer*, Vol.23, N°7, pp. 52-61, IEEE, July 1990

[Giralt & Vernières 1995] A. Giralt et F. Vernières : "Technologies vidéo pour la détection de baisse de vigilance des conducteurs routiers", *Mondial du*

Transport Routier, Paris, pp. 1-5, Société des Ingénieurs de l'Automobile (SIA), Septembre 1995

[Hansen 1994a] Hansen : "Chrysler leads in Class-B Mux", *The Hansen Report on Automotive Electronics*, Vol.7, N°1, pp. 1 and 7, P. Hansen Associates, February 1994

[Hansen 1994b] Hansen : "Multiplexing Update", *The Hansen Report on Automotive Electronics*, Vol.7, N°1, pp. 1,6,7 and 8, P. Hansen Associates, February 1994

[Hansen 1995] Hansen : "ABS Effectiveness Questioned", *The Hansen Report on Automotive Electronics*, Vol.8, N°3, pp. 1-2, P. Hansen Associates, April 1995

[Heinrich 1989] B.F. Heinrich : "Electrical/electronic systems integration design strategy", Société des Ingénieurs de l'Automobile (SIA), N°89055, pp. 138-141, October 1989

[Herbault 1991] P. Herbault : "Contribution to Integrated Vehicle Networking", *International Symposium, Vehicle Electronics Integration*, pp. 5-16, Associazone Tecnica dell'Automobile (ATA), April 1991

[Hoefflinger 1993] B. Hoefflinger : "An electronic copilot in your car ?", *Micro*, pp. 7-10, IEEE, February 1993

[Holliday & Vernon 1987] M.A. Holliday et M.K. Vernon : "A Generalized Timed Petri Net Model for Performance Analysis", *Transactions on Software Engineering*, Vol.SE-13, N°12, pp. 1297-1310, IEEE, December 1987

[Howard 1971] R.A. Howard : "Dynamic Probabilistic Systems, Vol.2 : Semi-Markov and Decision Processes", ISBN 0-471-41666-5, J. Wiley and Sons, 1971

[IEC_122 1995] IEC_122 : "Software for computers in the application of industrial safety-related systems (software standard)", IEC reference 65A (Secretariat) 122, International Electrotechnical Commission (IEC), 1995

[IEC_123 1995] IEC_123 : "Functional safety of electrical/electronic/programmable electronic systems; generic aspects : Part 1, General requirements (systems standard)", IEC reference 65A (Secretariat) 123, International Electrotechnical Commission (IEC), 1995

[ISO_10605 1995] ISO_10605 : "Road vehicles -- Electrical disturbances from electrostatic discharge", ISO/TR 10605:1994, International Organization for Standardization (ISO), September 1995

[ISO_11451 1995] ISO_11451 : "Road Vehicles -- Electrical disturbances by narrowband radiated electromagnetic energy -- Vehicle test method (Part 1 -

Part 4)", ISO/DIS 11451, International Organization for Standardization (ISO), September 1995

[ISO_11452 1995] ISO_11452 : "Road vehicles -- Electrical disturbances by narrowband radiated electromagnetic energy -- Component test methods (Part 5 - Part 7)", ISO/DIS 11452, International Organization for Standardization (ISO), September 1995

[ISO_11519 1995] ISO_11519 : "Road vehicles -- Low-speed serial data communication (Part 1 - Part 4)", ISO 11519, International Organization for Standardization (ISO), September 1995

[ISO_11898 1995] ISO_11898 : "Road vehicles -- Interchange of digital information -- Controller area network (CAN) for high-speed communication", ISO 11898:1993 + Amendment 1:1995, International Organization for Standardization (ISO), September 1995

[ISO_6527 1995] ISO_6527 : "Nuclear power plants -- Reliability data exchange -- General guidelines", ISO 6527:1982, International Organization for Standardization (ISO), September 1995

[ISO_7385 1995] ISO_7385 : "Nuclear power plants -- Guidelines to ensure quality of collected data on reliability", ISO 7385:1983, International Organization for Standardization (ISO), September 1995

[ISO_7498 1995] ISO_7498 : "Information processing systems -- Open Systems Interconnection -- Basic Reference Model (Part 1 - Part 4)", ISO/IEC 7498, International Organization for Standardization (ISO), September 1995

[ISO_7637 1995] ISO_7637 : "Road vehicles -- Electrical disturbance by conduction and coupling (Part 0 - Part 3)", ISO 7637, International Organization for Standardization (ISO), September 1995

[Kanoun *et al.* 1993] K. Kanoun, M. Kaâniche et J.-C. Laprie : "Experience in Software Reliability : From Data Collection to Quantitative Evaluation", *4th International Symposium on Software Reliability Engineering (ISSRE '93)*, pp. 234-245, IEEE, November 1993

[Kopetz 1994] H. Kopetz : "A Solution to an Automotive Control System Benchmark", Research Report Nr. 4/94, Institut für Technische Informatik, Technische Universität Wien, April 1994

[Kopetz 1995] H. Kopetz : "Automotive Electronics--Present State and Future Prospects", *25. International Symposium on Fault-Tolerant Computing (FTCS-25), Special Issue*, pp. 66-75, IEEE, June 1995

[Kopetz *et al.* 1989] H. Kopetz, A. Damm, C. Koza, M. Mulazzani, W. Schwabl, C. Senft et R. Zainlinger : "Distributed Fault-Tolerant Real-Time Systems : The MARS Approach", *Micro*, Vol.9, N°1, pp. 25-40, IEEE, February 1989

[Kopetz & Grünsteidl 1993] H. Kopetz et G. Grünsteidl : "TTP — A Time-Triggered Protocol for Fault-Tolerant Real-Time Systems", *23. International Symposium on Fault-Tolerant Computing (FTCS-23)*, pp. 524-533, IEEE, June 1993

[Kopetz & Grünsteidl 1994] H. Kopetz et G. Grünsteidl : "TTP — A Protocol for Fault-Tolerant Real-Time Systems", *Computer*, Vol.27, N°1, pp. 14-23, IEEE, January 1994

[Kopetz & Ochsenreiter 1987] H. Kopetz et W. Ochsenreiter : "Clock Synchronization in Distributed Real-Time Systems", *Transactions on Computers*, Vol.C-36, N°8, pp. 933-940, IEEE, August 1987

[Kopetz & Schwabl 1989] H. Kopetz et W. Schwabl : "Global Time in Distributed Real-Time Systems", Research Report Nr. 15/89, Institut für Technische Informatik
Technische Universität Wien, October 1989

[Lamport *et al.* 1982] L. Lamport, R. Shostak et M. Pease : "The Byzantine Generals Problem", *Transactions on Programming Languages and Systems*, Vol.4, N°3, pp. 382-401, Association for Computing Machinery (ACM), July 1982

[Landwehr *et al.* 1994] C.E. Landwehr, A.R. Bull, J.P. McDermott et W.S. Choi : "A Taxonomy of Computer Program Security Flaws", *Computing Survey*, Vol.26, N°3, pp. 211-254, Association for Computing Machinery (ACM), September 1994

[Laprie 1975] J.-C. Laprie : "Prévision de la Sûreté de Fonctionnement et Architecture de Structures Numériques Temps Réel Réparables", Thèse de Doctorat d'Etat (N° 669), LAAS-CNRS, Toulouse, France, Juin 1975

[Laprie 1993] J.-C. Laprie : "Informatique sûre de fonctionnement : des concepts aux limites", *Actes du Séminaire LAAS (25. anniversaire du LAAS-CNRS)*, pp. 43-54, Cépaduès-Editions, Mai 1993

[Laprie *et al.* 1996] J.-C. Laprie, J. Arlat, J.-P. Blanquart, A. Costes, Y. Crouzet, Y. Deswarte, J.-C. Fabre, H. Guillermain, M. Kaâniche, K. Kanoun, C. Mazet, D. Powell, C. Rabéjac et P. Thévenod : "Guide de la sûreté de fonctionnement", ISBN 2-85428-382-1 (2. édition), Cépaduès-Editions, Janvier 1996

[Laprie *et al.* 1990] J.-C. Laprie, C. Béounes, M. Kaâniche et K. Kanoun : "The Transformation Approach to the Modeling and Evaluation of the Reliability and Availability Growth", *20th International Symposium on Fault-Tolerant Computing (FTCS-20)*, pp. 364-371, IEEE, Juin 1990

[Lawson *et al.* 1992] H.W. Lawson, M. Lindgren, M. Strömberg, T. Lundqvist, K.-L. Lundbäck, L.-A. Johannson, J. Torin, P. Gunningberg et H. Hansson : "Guidelines for Basement : A Real-Time Architecture for Automotive

Systems", Mecel, Inc., Göteborg, and Lawson Publishing and Consulting, Inc., Lidingö, Sweden, November 1992

[Lorenz *et al.* 1988] K. Lorenz, R. Hofmann et K. Hönnebeck : "Interactive engine and transmission control", *International Congress on Transportation Electronics (Convergence)*, pp. 25-31, IEEE, October 1988

[Mahalek 1992] J. Mahalek : "Multiplex Systems in the BMW 850i", Society of Automotive Engineers (SAE), N°920225, pp. 13-22, February 1992

[Marsan *et al.* 1984] M.A. Marsan, G. Balbo et G. Conte : "A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocessor Systems", *Transactions on Computer Systems*, Vol.2, N°2, pp. 93-122, Association for Computing Machinery (ACM), May 1984

[Marsan & Chiola 1986] M.A. Marsan et G. Chiola : "On Petri Nets with Deterministic and Exponentially Distributed Firing Times", *7th European Workshop on Applications and Theory of Petri Nets*, pp. 132-145, Springer-Verlag, Lecture Notes in Computer Science 266, "Advances in Petri Nets, 1987", June 1986

[Martin 1993] P. Martin : "Autonomous intelligent cruise control incorporating automatic braking", Society of Automotive Engineers (SAE), N°930510, pp. 53-59, March 1993

[Mathony *et al.* 1990] H.-J. Mathony, J. Unruh et K.-H. Kaiser : "Zur Sicherheit der Datenübertragung in KFZ-Netzwerken", VDI-Berichte Nr.819, pp. 515-539, Verein Deutscher Ingenieure (VDI), 1990

[Mauro 1993] V. Mauro : "Programmes de recherche en télématique avancée pour les transports", *Recherche Transports Sécurité (RTS)*, N°38-39, pp. 147-154, Juin 1993

[Meyer 1982] J.F. Meyer : "Closed-Form Solutions of Performability", *Transactions on Computers*, Vol.C-31, N°7, pp. 648-657, IEEE, July 1982

[Meyna & Gerstenmeier 1992] A. Meyna et J. Gerstenmeier : "Probalistische Zuverlässigkeits- und Sicherheitsanalyse für die redundante Steuereinrichtung eines Antiblockiersystems", VDI-Berichte Nr.1009, "Elektronik im Kraftfahrzeug", pp. 413-446, Verein Deutscher Ingenieure (VDI), September 1992

[Molloy 1982] M.K. Molloy : "Performance Analysis Using Stochastic Petri Nets", *Transactions on Computers*, Vol.C-31, N°9, pp. 913-917, IEEE, September 1982

[Moreira de Souza & Landrault 1981] J. Moreira de Souza et C. Landrault : "Benefit Analysis of Concurrent Redundancy Techniques", *Transactions on Reliability*, Vol.R-30, N°1, pp. 67-70, IEEE, April 1981

[Morin 1994] F. Morin : "Architecture électrique et électronique des véhicules", *Congrès SIA-Electronique, Toulouse*, Société des Ingénieurs de l'Automobile (SIA), Avril 1994

[Muzet & Clot 1993] A. Muzet et J. Clot : "Le projet PAVCAS : poste d'analyse de la vigilance en conduite automobile simulée. Une approche pluridisciplinaire de la vigilance au volant", Rapport N°93508, LAAS-CNRS, Toulouse, France, 1993

[Nakano 1988] Y. Nakano : "Engine-transmission integrated control", *International Congress on Transportation Electronics (Convergence)*, pp. 77-82, IEEE, October 1988

[Noble 1992] I.E. Noble : "EMC and the automotive industry", *Electronics & Communication Engineering Journal*, pp. 263-271, IEE, October 1992

[Numazawa 1988] A. Numazawa : "Automotive electronics in passenger cars", *International Congress on Transportation Electronics (Convergence)*, pp. 11-24, IEEE, October 1988

[Ortis *et al.* 1992] M. Ortis, I. Lacrouts-Cazenave et P. Herbault : "Van — The Optimized Industrial Solution for Vehicle Multiplexing (Vehicle Area Network)", Society of Automotive Engineers (SAE), N°920223, pp. 7-11, February 1992

[Palmquist 1993] U. Palmquist : "Intelligent cruise control and roadside information", *Micro*, pp. 20-28, IEEE, February 1993

[Panik 1988] F. Panik : "Electronics in European trucks", *International Congress on Transportation Electronics (Convergence)*, pp. 251-257, IEEE, October 1988

[Panizza 1989] E. Panizza : "Automotive electronics integration — how far should we go ?", *20. International Symposium on Automotive Technology & Automation (ISATA)*, Vol.1, pp. 23-32, Automotive Automation Limited, England, May 1989

[Paret 1992a] D. Paret : "Applications du bus CAN (3)", *Electronique Radio Plans*, N°541, pp. 63-69, Décembre 1992

[Paret 1992b] D. Paret : "Le bus CAN — les composants existants (2)", *Electronique Radio Plans*, N°540, pp. 29-36, Novembre 1992

[Paret 1992c] D. Paret : "Le bus CAN — son protocole et ses particularités (1)", *Electronique Radio Plans*, N°539, pp. 55-59, Octobre 1992

[Paret 1993] D. Paret : "Carte unité centrale "BUS CAN" à l'aide du 87C592 (4)", *Electronique Radio Plans*, N°542, pp. 67-72, Janvier 1993

[Paulsen 1988] J.J. Paulsen : "Powertrain sensors and actuators : driving toward optimized vehicle performance", *International Congress on Transportation Electronics (Convergence)*, pp. 43-63, IEEE, October 1988

[Poledna 1995a] S. Poledna : "Fault Tolerance in Safety Critical Automotive Applications: Cost of Agreement as a Limiting Factor", *25. International Conference on Fault-Tolerant Computing (FTCS-25)*, pp. 73-82, IEEE, June 1995

[Poledna 1995b] S. Poledna : "Tolerating Sensor Timing Faults in Highly Responsive Hard Real-Time Systems", *Transactions on Computers*, Vol.44, N°2, pp. 181-191, IEEE, February 1995

[Poulard 1996] H. Poulard : "Statistiques et réseaux de neurones pour un système de diagnostic : application au diagnostic de pannes automobiles", Rapport N°96.177 (Thèse de Doctorat), LAAS-CNRS, Toulouse, France, Mai 1996

[Rathkey 1992] P.W.K. Rathkey : "Automotive Electronics — a profile of international markets and suppliers to 1997", ISBN 1-85617-147-7, Elsevier Advanced Technology, October 1992

[Rittmannsberger 1988] N. Rittmannsberger : "Antilock braking system and traction control", *International Congress on Transportation Electronics (Convergence)*, pp. 195-202, IEEE, October 1988

[Rivard 1989] J.G. Rivard : "Global automotive electronics — future technical and business challenges", *20. International Symposium on Automotive Technology & Automation (ISATA)*, Vol.1, pp. 3-20, Automotive Automation Limited, England, May 1989

[Rodda 1990] W.J. Rodda : "Vehicular electronics and its impact on software development", *22. International Symposium on Automotive Technology & Automation (ISATA)*, Vol.1, pp. 55-59, Automotive Automation Limited, England, May 1990

[Saad 1988] F. Saad : "Prise de risque ou non-perception du danger", *Recherche Transports Sécurité (RTS)*, N°18-19, pp. 55-62, Septembre 1988

[SAE_J1213/1 1994] SAE_J1213/1 : "SAE J1213/1 JUN91 — Glossary of Vehicle Networks for Multiplexing and Data Communications", dans *1994 SAE Handbook*, pp. 23.19-23.22, ISBN 1-56091-461-0, Society of Automotive Engineers (SAE), 1994

[SAE_J1567 1994] SAE_J1567 : "SAE J1567 AUG87 — Collision Detection Serial Data Communications Multiplex Bus", dans *1994 SAE Handbook*, pp. 23.481-23.486, ISBN 1-56091-461-0, Society of Automotive Engineers (SAE), 1994

[SAE_J1583 1994] SAE_J1583 : "SAE J1583 MAR90 — Controller Area Network (CAN), an In-Vehicle Serial Communication Protocol", dans *1994 SAE Handbook*, pp. 23.486-23.499, ISBN 1-56091-461-0, Society of Automotive Engineers (SAE), 1994

[SAE_J1587 1994] SAE_J1587 : "SAE J1587 AUG92 — Joint SAE/TMC Electronic Data Interchange between Microcomputer Systems in Heavy-duty Vehicle Applications", dans *1994 SAE Handbook*, pp. 23.531-23.553, ISBN 1-56091-461-0, Society of Automotive Engineers (SAE), 1994

[SAE_J1850 1994] SAE_J1850 : "SAE J1850 AUG91 — Class B Data Communication Network Interface", dans *1994 SAE Handbook*, pp. 23.268-23.284, ISBN 1-56091-461-0, Society of Automotive Engineers (SAE), 1994

[SAE_J2056/1 1994] SAE_J2056/1 : "SAE J2056/1 JUN93 — Class C Application Requirement Considerations", dans *1994 SAE Handbook*, pp. 23.366-23.372, ISBN 1-56091-461-0, Society of Automotive Engineers (SAE), 1994

[SAE_J2056/2 1994] SAE_J2056/2 : "SAE J2056/2 APR93 — Survey of Known Protocols", dans *1994 SAE Handbook*, pp. 23.372-23.390, ISBN 1-56091-461-0, Society of Automotive Engineers (SAE), 1994

[SAE_J2057/1 1994] SAE_J2057/1 : "SAE J2057/1 JUN91 — Class A Application/Definition", dans *1994 SAE Handbook*, pp. 23.407-23.413, ISBN 1-56091-461-0, Society of Automotive Engineers (SAE), 1994

[SAE_J2106 1994] SAE_J2106 : "SAE J2106 APR91 — Token Slot Network for Automotive Control", dans *1994 SAE Handbook*, pp. 23.431-23.446, ISBN 1-56091-461-0, Society of Automotive Engineers (SAE), 1994

[Savary 1993] J. Savary : "Tout ce que doivent savoir ceux qui se croient invulnérables au volant", *Auto—Moto*, N°128, pp. 36-40, Août 1993

[Schilke *et al.* 1988] N.A. Schilke, R.D. Fruechte, N.M. Boustany, A.M. Karmel, B.S. Repa et J.H. Rillings : "Integrated vehicle control", *International Congress on Transportation Electronics (Convergence)*, pp. 97-106, IEEE, October 1988

[Sellner & Merker 1993] H.-J. Sellner et W. Merker : "Elektronisches Motormanagement-System für den Audi V6-2,6-l-Motor", *Motortechnische Zeitschrift (MTZ)*, N°54, pp. 340-344, Juli 1993

[SIEMENS 1993] SIEMENS : "TOMSPIN Handbuch", SIEMENS AG, Zentralabteilung Forschung und Entwicklung ZFE ST SN 13, München, 1993

[Siewiorek & Swarz 1992] D.P. Siewiorek et R.S. Swarz : "Reliable Computer Systems — Design and Evaluation", ISBN 1-55558-075-0, Digital Press / Prentice-Hall International Edition, 1992

[Specker 1993] B. Specker : "Using Hierarchical GSPN Models for the Evaluation of Large Fault-Tolerant Systems", Siemens Corporate Research and Development, ZFE ST SN 13, München, 1993

[Szczerbicka 1992] H. Szczerbicka : "A combined queueing network and stochastic Petri-net approach for evaluating the performability of fault-tolerant computer

systems", *Performance Evaluation 14*, pp. 217-226, Elsevier Science Publishers B.V., North-Holland, 1992

[Tindell & Burns 1994] K. Tindell et A. Burns : "Guaranteed Message Latencies for Distributed Safety-Critical Hard Real-Time Control Networks", Real-Time Systems Research Group, Department of Computer Science, University of York, YO1 5DD, England, FTP : minster.york.ac.uk, directory /pub/realtime/papers, May 1994

[Torin 1992] J. Torin : "Dependability in complex automotive systems — Requirements directions and drivers", Rapport PROMETHEUS, Chalmers University of Technology Göteborg, September 1992

[Villemeur 1988] A. Villemeur : "Sûreté de fonctionnement des systèmes industriels", ISBN (ISSN) 0399-4198, Editions Eyrolles, 1988

[Watson 1988] P.K. Watson : "Integration of electronic control units for automotive control applications", *18. International Symposium on Automotive Technology & Automation (ISATA)*, Vol.1, pp. 1-8, Automotive Automation Limited, England, May 1988

[Zanoni & Pavan 1993] E. Zanoni et P. Pavan : "Improving the reliability and safety of automotive electronics", *Micro*, pp. 30-48, IEEE, February 1993

[Zeltwanger 1994] H. Zeltwanger : "Le Bus CAN arrive dans l'Industrie", *Electronique*, N°44, pp. 62-66, Janvier 1994

[Ziegler *et al.* 1994a] C. Ziegler, P. Desroches et D. Powell : "Automobile", dans *Informatique Tolérante aux Fautes, Arago 15*, pp. 167-177, ISBN 2-225-84522-0, Editions Masson, 1994

[Ziegler *et al.* 1994b] C. Ziegler, D. Powell et P. Desroches : "Dependability of On-board Automotive Computer Systems", *Intelligent Vehicles '94 Symposium*, pp. 568-575, IEEE, October 1994

Table des matières

Thèse de **Monsieur Christian Ziegler**

"Sûreté de fonctionnement d'architectures informatiques embarquées sur automobile"

Résumé

Les travaux présentés dans ce mémoire visent à s'assurer que la complexité des parties informatiques et électroniques de systèmes embarqués sur l'automobile ne conduise pas globalement à une dégradation de la sûreté de fonctionnement du véhicule par rapport à un véhicule équipé de commandes mécaniques. La tendance actuelle vers l'intégration des différents systèmes électroniques embarqués nous a conduit à développer un spectre d'architectures qui nous permet d'illustrer, classifier et comparer différentes possibilités existantes entre une architecture entièrement fédérée à un bout du spectre et une architecture entièrement intégrée à l'autre. Après une comparaison qualitative des architectures présentées (avec un effet de loupe sur l'aspect coût), nous focalisons sur l'évaluation quantitative de leur sûreté de fonctionnement. Parmi les différentes techniques d'évaluation généralement employées nous choisissons la technique d'évaluation probabiliste par Réseaux de Petri Stochastiques Généralisés. L'originalité de la méthode réside dans le fait de modéliser les aspects fonctionnels indépendamment de l'architecture ainsi que de l'activation du véhicule. Nous définissons les mesures de sûreté de fonctionnement à l'aide d'un modèle fonctionnel dont les changements d'état sont dictés par un modèle du calculateur sous-jacent. Plusieurs architectures du calculateur sont modélisées sans changer le modèle fonctionnel. Les résultats obtenus pour deux fonctions, à savoir le coussin gonflable et la direction électronique, permettent en particulier de tirer des conclusions concernant le choix d'une architecture pour une fonction donnée.

Mots Clés : Informatique embarquée, Electronique automobile, Sûreté de fontionnement, Tolérance aux fautes, Modélisation markovienne, Réseaux de Petri stochastiques généralisés

"Dependability of automotive embedded computer architectures"

Abstract

The aim of the work presented in this dissertation is to guarantee that the increasing complexity of automotive embedded computer architectures does not degrade the dependability of the vehicle with respect to that of the vehicle containing classical mechanical equipment. The current tendency towards integration of the different embedded computer systems leads us to develop a spectrum of architectures allowing us to illustrate, classify and compare several possibilities between an entirely-federated architecture at one end of the spectrum and an entirely-integrated architecture at the other. After a qualitative comparison of the presented architectures (focussing on the cost criteria), we concentrate on quantitative evaluation of dependability. Among the different evaluation techniques generally used we choose a probabilistic evaluation technique using Generalised Stochastic Petri Nets. The originality of the method is that it models functional aspects independently from the architecture. We define the dependability measures on the base of the functional model whose changes are dictated by an underlying architecture model. Several architectures are modeled without changing the functional model. The results obtained for two functions, an airbag and a steer-by-wire system, allow conclusions to be drawn concerning the choice of an architecture for a given function.

Key Words : Embedded software, Automotive electronics, Dependability, Fault tolerance, Markov modelling, Generalised stochastic Petri nets