Conception et validation de systèmes informatiques à haute sûreté de fonctionnement

Eric Pilaud

THESE

présentée à

l'Institut National Polytechnique de Grenoble

pour obtenir le grade de
DOCTEUR INGENIEUR
«Génie Informatique»

par

Eric PILAUD

CONCEPTION ET VALIDATION DE SYSTEMES INFORMATIQUES A HAUTE SURETE DE FONCTIONNEMENT.

Thèse soutenue le 23 novembre 1982 devant la commission d'examen.

	C. DELOBEL	Président
Messieurs	J.P. AUCLAIR	
	J.P. BANATRE	
	P. CASPI	Examinateurs
	J. GOLDBERG	
	J.C. LAPRIE	
Madame	G. SAUCIER	

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

Année universitaire 1979-1980

Président : M. Philippe TRAYNARD

Vice-Présidents : M. Georges LESPINARD
M. René PAUTHENET

PROFESSEURS DES UNIVERSITES

MM.	ANCEAU François	Informatique fondamentale et appliquée
	BENOIT Jean	Radioélectricité
	BESSON Jean	Chimie Minérale
	BLIMAN Samuel	Electronique
	BLOCH Daniel	Physique du Solide - Cristallographie
	BOIS Philippe	Mécanique
	BONNETAIN Lucien	Génie Chimique
	BONNIER Etienne	Métallurgie
	BOUVARD Maurice	Génie Mécanique
	BRISSONNEAU Pierre	Physique des Matériaux
	BUYLE-BODIN Maurice	Electronique
	CHARTIER Germain	Electronique
	CHERADAME Hervé	Chimie Physique Macromoléculaires
Mme	CHERUY Arlette	Automatique
MM.	CHIAVERINA Jean	Biologie, Biochimie, Agronomie
	COHEN Joseph	Electronique
	COUMES André	Electronique
	DURAND Francis	Métallurgie
	DURAND Jean-Louis	Physique Nucléaire et Corpusculaire
	FELICI Noël	Electrotechnique
	FOULARD Claude	Automatique
	GUYOT Pierre	Métallurgie Physique
	IVANES Marcel	Electrotechnique
	JOUBERT Jean-Claude	Physique du Solide - Cristallographie
	LACOUME Jean-Louis	Géographie - Traitement du Signal
	LANCIA Roland	Electronique - Automatique
	LESIEUR Marcel	Mécanique
	LESPINARD Georges	Mécanique
	LONGEQUEUE Jean-Pierre	Physique Nucléaire Corpusculaire
	MOREAU René	Mécanique
	MORET Roger	Physique Nucléaire Corpusculaire
	PARIAUD Jean-Charles	Chimie - Physique
	PAUTHENET René	Physique du Solide - Cristallographie
	PERRET René	Automatique

.../...

MM.	PERRET Robert	Electrotechnique
	PIAU Jean-Michel	Mécanique
	PIERRARD Jean-Marie	Mécanique
	POLOUJADOFF Michel	Electrotechnique
	POUPOT Christian	Electronique - Automatique
	RAMEAU Jean-Jacques	Chimie
	ROBERT André	Chimie Appliquée et des matériaux
	ROBERT François	Analyse numérique
	SABONNADIERE Jean-Claude	Electrotechnique
Mme	SAUCIER Gabrielle	Informatique fondamentale et appliquée
M.	SOHM Jean-Claude	Chimie - Physique
Mme	SCHLENKER Claire	Physique du Solide - Cristallographie
MM.	TRAYNARD Philippe	Chimie - Physique
	VEILLON Gérard	Informatique fondamentale et appliquée
	ZADWORNY François	Electronique

CHERCHEURS DU C.N.R.S. (Directeur et Maître de Recherche)

M.	FRUCHART Robert	Directeur de Recherche
MM.	ANSARA Ibrahïm	Maître de Recherche
	BRONOEL Guy	Maître de Recherche
	CARRE René	Maître de Recherche
	DAVID René	Maître de Recherche
	DRIOLE Jean	Maître de Recherche
	KAMARINOS Georges	Maître de Recherche
	KLEITZ Michel	Maître de Recherche
	LANDAU Ioan-Doré	Maître de Recherche
	MERMET Jean	Maître de Recherche
	MUNIER Jacques	Maître de Recherche

Personnalités habilitées à diriger des travaux de recherche (décision du Conseil Scientifique)

E.N.S.E.E.G.

MM.	ALLIBERT Michel	
	BERNARD Claude	
	CAILLET Marcel	
Mme	CHATILLON Catherine	
MM.	COULON Michel	
	HAMMOU Abdelkader	
	JOUD Jean-Charles	
	RAVAINE Denis	
	SAINFORT	C.E.N.G.

.../...

MM. SARRAZIN Pierre
SOUQUET Jean-Louis
TOUZAIN Philippe
URBAIN Georges — Laboratoire des Ultra-Réfractaires ODEILLO

E.N.S.M.E.E.

MM. BISCONDI Michel
BOOS Jean-Yves
GUILHOT Bernard
KOBILANSKI André
LALAUZE René
LANCELOT Françis
LE COZE Jean
LESBATS Pierre
SOUSTELLE Michel
THEVENOT François
THOMAS Gérard
TRAN MINH Canh
DRIVER Julian
RIEU Jean

E.N.S.E.R.G.

MM. BOREL Joseph
CHEHIKIAN Alain
VIKTOROVITCH Pierre

E.N.S.I.E.G.

MM. BORNARD Guy
DESCHIZEAUX Pierre
GLANGEAUD François
JAUSSAUD Pierre
Mme JOURDAIN Geneviève
MM. LEJEUNE Gérard
PERARD Jacques

E.N.S.H.G.

M. DELHAYE Jean-Marc

E.N.S.I.M.A.G.

MM. COURTIN Jacques
LATOMBE Jean-Claude
LUCAS Michel
VERDILLON André

A mes Parents...

Je tiens à exprimer toute ma reconnaissance à Madame Gabrièle SAUCIER, Professeur à l'ENSIMAG, pour m'avoir accueilli dans son équipe de recherche et m'avoir dirigé dans mes travaux.

Je tiens à remercier :

Monsieur Claude DELOBEL, Professeur à l'USMG, de me faire l'honneur de présider mon jury de thèse,

Monsieur Jean Claude LAPRIE, Maître de Recherche au LAAS de Toulouse, d'avoir accepté d'être le rapporteur de cette thèse et qui par ses conseils et remarques a contribué à son amélioration,

Monsieur Jean Pierre AUCLAIR, Chef de la division VZA de la SNCF, dont j'ai pu apprécier les remarques au cours d'une collaboration fructueuse avec ses services,

Monsieur Paul CASPI, Chargé de recherche au CNRS, qui a animé ce travail de recherche et dont j'ai particulièrement apprécié l'amitié et la compétence.

Je tiens également à remercier :

Monsieur Jean Pierre BANATRE, Ingénieur de recherche de l'IRISA de Rennes, et Monsieur Jack GOLDBERG, Directeur du laboratoire d'informatique au Stanford Research Institute, de m'avoir fait l'honneur de participer à ce jury.

Je voudrais que soient également remercier ici :

Monsieur Jacques PULOU, dont les travaux sont à l'origine de cette thèse,

Tous mes camarades de l'Equipe Concpetion et Sécurité des Systèmes, dont les remarques et l'amitié m'ont aidé,

Les membres de l'atelier de micro-informatique de l'IMAG où la maquette du calculateur CARL présentée ici, a été réalisée, et parmi eux tout particulièrement Patrick BOREL qui a réalisé la version actuelle,

Geneviève BOULESTEIX qui a assuré la frappe de ce travail et l'équipe de reprographie de l'IMAG, animée par Daniel IGLESIAS qui en a assuré le tirage : cette thèse m'a permis d'apprécier une fois de plus leur gentillesse, leur patience et leur compétence.

P L A N

CHAPITRE I

INTRODUCTION

L'automatisation de nombreux systèmes de commande ou de contrôle de processus et le remplacement de systèmes logiques ou analogiques peu intégrés par des systèmes à haute intégration ont conduit à un élargissement considérable du domaine de la haute sûreté de fonctionnement informatique : ce domaine concerne les applications pour lesquelles un bon fonctionnement continu est impératif, soit en raison de la mise en jeu de vies humaines (transports...), soit à cause de l'investissement élevé qui serait perdu par la défaillance du calculateur (espace...), soit même à cause du coût et de la gêne que pourraient provoquer des défaillances (gestion de certains processus industriels, gestion bancaire...).

Dès les premières phases de l'étude de tels systèmes, les problèmes liés à leur validation se trouvent au centre des préoccupations du concepteur : il conviendra de "prouver" la bonne conception des mécanismes permettant de réagir à l'apparition de pannes, de vérifier cette conception (tests, simulation...) et d'estimer, de manière convaincante, des grandeurs prévues et significatives mesurant les performances de sûreté de fonctionnement. Le travail présenté ici s'intéresse à ces deux aspects de la sûreté de fonctionnement : la conception et la validation. Avant de situer et de présenter ce travail, il convient de préciser les définitions des concepts de base du domaine.

I.1 - CONCEPTS DE BASE

I.1.1. Pannes, défaillances, défauts, fautes, erreurs

La terminologie employée pour designer les fonctionnements anormaux d'un système est très controversée, en dépit d'un effort récent pour tenter de l'unifier [FTC12]. Le problème réside essentiellement dans le fait que les auteurs cherchent à distinguer, par cette terminologie, des critères différents. En effet, on peut chercher à caractériser :

- le niveau d'abstraction d'un fonctionnement anormal : on peut considérer les fonctionnements anormaux d'un niveau très fin (non respect des caractéristiques physiques d'un transistor) à un niveau global (réalisation incorrecte d'une application) [AVI82];
- les causes et les effets d'un fonctionnement anormal : il s'agit de distinguer l'origine d'un mauvais fonctionnement de son effet observable dans le système. Parmi les causes, on pourra chercher à distinguer les causes physiques (occurence d'une altération du matériel) des causes humaines (existence d'une erreur d'analyse ou de conception d'un matériel ou d'un logiciel)[KOP82][LAP82];
- les phénomènes de manifestation des fonctionnements anormaux : il s'agit de distinguer l'existence d'un fonctionnement anormal à l'intérieur d'une entité (circuit, calculateur ou système) de sa manifestation observable à l'extérieur de celle-ci [LAP82].

Dans ce travail, nous nous préoccupons surtout des effets des fonctionnements anormaux et non des causes, et cela au niveau d'unités fonctionnelles entières (processeurs, mémoires, calculateurs, systèmes).

Ainsi, il nous suffira de différencier l'existence ou l'occurence d'un fonctionnement anormal à l'intérieur d'une unité (panne) de sa manifestation à l'extérieur de cette unité (erreur).

Nous ne mentionnerons que très rarement la cause d'une panne matérielle (nous le ferons alors sous le terme de défaillance). Dans le domaine logiciel, la mauvaise conception d'un programme sera appelée faute de conception ou de programmation.

I.1.2. Tolérance aux fautes et évitement de fautes

Ces termes désignent deux approches complémentaires du problème de la sûreté de fonctionnement :

l'évitement de fautes ou de pannes consiste à essayer, à priori, d'éviter ou de limiter les occurences de fonctionnements anormaux : utilisation de composants fiables pour limiter les pannes matérielles, approche structurée et prudente de la conception pour éviter les fautes de conception ; c'est cette approche d'évitement de fautes que nous emploierons dans l'écriture des logiciels présentés dans cette thèse ;

la tolérance aux fautes ou aux pannes consiste à considérer que les fonctionnements anormaux sont des évènements qui peuvent, de toutes façons, arriver ; il s'agit alors de permettre aux systèmes d'y faire face, en les détectant ou/et en y survivant ; la plus grande partie de ce travail s'intéresse aux pannes matérielles pour lesquelles nous appliquerons l'approche de tolérance aux pannes.

I.1.3. Fiabilité et sécurité

Un certain nombre de grandeurs ont été définies pour mesurer la sûreté de fonctionnement d'un système [LAP75][LAP82]. Dans le cadre de ce travail, nous chercherons à évaluer deux types de grandeurs, la fiabilité et la sécurité ; suivant le contexte d'utilisation des systèmes considérés (mission de courte durée ou fonctionnement continu), nous utiliserons des mesures différentes :

- la fiabilité de mission, qui représente la probabilité de bon fonctionnement continu pendant la durée de mission. Plus formellement, on définit l'état E d'un système à l'instant t comme étant correct (c) ou non correct (nc) ; la fiabilité R(T) est donnée par :

$$R(T) = \text{Prob}\ (E(t) = c,\ t \in [0,T]).$$

- la sécurité de mission, qui représente la probabilité de fonctionnement non dangereux pendant la durée de mission : en partitionnant l'ensemble des des états non corrects en états dangereux (d) et non dangereux (nd), la sécurité S(T) s'écrit :

$$S(T) = \text{Prob}\ (E(t) \neq d,\ t \in [0,T]).$$

Notons que la notion de fonctionnement non dangereux est plus large que la notion de bon fonctionnement, de sorte que :

$$S(T) \geq R(T).$$

La notion de fonctionnement non dangereux est définie en général dans le cas ou l'arrêt du calculateur est admissible (alors que la poursuite erronée du calcul ne l'est pas). Remarquons cependant que, même lorsque la sécurité est l'objectif poursuivi, l'obtention d'une fiabilité acceptable est un souci du concepteur : un système toujours

à l'arrêt atteint l'objectif de haute sécurité mais est inutilisable.

- le taux d'insécurité permet de définir dans le cas des systèmes en fonctionnement continu la notion de sécurité :

$$TIS(t) = \lim_{t \to 0} \frac{1}{\Delta} \text{ Prob } (E(t+\Delta) = d \ / \ E(t) \neq d).$$

I.2 - CONCEPTION ET VALIDATION

Le concepteur d'un système tolérant les fautes se préoccupe de réaliser un système qui, malgré les pannes et fonctionnements anormaux internes, ait une faible probabilité de mal accomplir sa mission.

Le problème est donc de concevoir un tel système mais aussi de démontrer sa faible probabilité d'échec, c'est à dire de le valider (prouver et tester (par simulation ou injection de fautes) la bonne réalisation des mécanismes et évaluer les grandeurs significatives de sûreté de fonctionnement).

Les études réalisées dans le domaine présentent une caractéristique de grande variété et ne concernent en général qu'un point particulier du problème : étude d'un mécanisme original de survie aux fautes, présentation d'un système ou évaluation d'une stratégie particulière.

Il apparait de toutes façons que la validation d'un système entier est un problème difficile en raison de la complexité des analyses à mener et de l'incertitude existant sur la connaissance de certains phénomènes. Cette incertitude peut difficilement être levée par l'observation de résultats d'expériences, en raison de la durée et de l'importance que celles-ci devraient avoir pour être significatives. En effet, les phénomènes de pannes matérielles étant intrinsèquement rares dans

un circuit intégré, il faudrait observer beaucoup de circuits pendant longtemps pour obtenir un échantillon suffisant de réactions d'un système à une variété suffisante de pannes. Cette incertitude est en général levée par des hypothèses floues ou non crédibles ce qui peut être inadmissible si la validation est amenée à faire foi sur le plan juridique ("certification").

Le travail présenté ici est basé sur ces remarques : nous présentons une approche de la conception permettant de structurer les systèmes afin de réduire la complexité de leur validation et nous proposons des méthodes d'évaluation "crédibles" permettant de limiter le nombre d'hypothèses sous lesquelles les résultats sont obtenus.

I.3 -PLAN DE LA THESE

Nous présenterons dans une première partie notre approche de la conception : cette approche est basée sur un effort de caractérisation des systèmes existants ; cette caractérisation permet d'exhiber les principaux choix de conception sur lesquels est fondée notre approche.

La deuxième partie présente notre méthode d'évaluation appliquée à une large classe de stratégies de tolérance aux pannes.

Enfin, dans une troisième partie, nous nous intéressons à une classe de systèmes dont la validation présente des caractéristiques particulières : les systèmes dont la tolérance aux pannes est assurée par des mécanismes logiciels.

Tous ces aspects sont appliqués à un cas particulier : nous présentons la conception et la validation d'un système tolérant les pannes réalisé pendant cette étude : CARL (Calculateur A Reconfiguration Logicielle).

C H A P I T R E I I

UNE DEMARCHE DE CONCEPTION DES SYSTEMES A HAUTE SURETE DE FONCTIONNEMENT

II.1 - INTRODUCTION

L'objet de ce chapitre est d'apporter au concepteur d'un système à haute sûreté de fonctionnement un guide, lui permettant de dégager clairement les choix qui lui sont offerts, puis de s'orienter vers la conception d'un système simple et bien structuré (c'est-à-dire "facilement validable").
La nécessité d'un tel guide apparaît clairement si l'on consulte les travaux publiés sur la conception de dispositifs tolérant les pannes ; ceux-ci peuvent être grossièrement répartis en deux types :

- de nombreux travaux proposent et étudient des stratégies de tolérance aux pannes pour un calculateur ou pour des parties d'un calculateur. Ces stratégies s'appliquent au niveau de la porte ("quadded logic", "fail safe logic" [TRY] , [JEN63] , [MIN67]), de mémoires ou d'organes de transmission (codes détecteurs ou correcteurs [AVI71], [PET72], [STI77]), de circuits d'horloge ([DAL73], [LEW79]) ou à un niveau quelconque (redondances massives, hybrides, stratégies de détection, localisation, remplacement [MAT71] , [LOS76]),
- deuxièmement, des études présentent des calculateurs à haute sûreté de fonctionnement dont la réalisation est aujourd'hui, sinon achevée, du moins largement entamée ([SIE78], [WAK76], [MER76] , [HOP78], [WEN78], [MOR76]). Ces études décrivent les principes de base sur lesquels sont fondés ces calculateurs ainsi que les stratégies particulières appliquées à chacun de leurs organes. Des évaluations de fiabilité ou de sécurité complètent ces études.

Le concepteur d'un nouveau système tolérant les pannes doit, bien-sûr, s'appuyer sur les résultats précédemment acquis et notamment ceux issus de ce second

type d'études. Il se heurte alors à deux problèmes :

- ces calculateurs sont sensiblement différents les uns des autres, chaque concepteur développant son point de vue pratiquement indépendamment des autres.

- les choix effectués apparaissent souvent mal et ne sont que trop rarement ([WEN78], [BEO78]) situés par rapport aux autres possibilités qui étaient offertes

Afin de faire face à ces problèmes, nous proposons un système de description des calculateurs tolérant les pannes, permettant de situer les diverses machines existantes entre elles : ce système doit permettre, d'une part de dégager la suite des choix liés à la tolérance aux pannes effectués durant la conception de ces machines, d'autre part de mettre en évidence les autres possibilités offertes lors de chacun de ces choix ainsi que leurs conséquences, tant sur les choix ultérieurs que sur les performances de sûreté de fonctionnement qui en résultent. Par rapport aux systèmes déja proposés ([LAP79], [AND81]...), ce système se caractérise par le fait qu'il vise surtout à faire apparaitre une partition claire de l'architecture afin de bien décrire les réactions des calculateurs aux pannes matérielles. En ce sens, il ne prétend pas permettre une caractérisation exhaustive de la philosophie de chaque calculateur

Afin de refléter ces divers choix de conception, nous proposons des descriptions formées par la succession de quatre rubriques, relevant chacune d'un type de choix particulier :

. structure fonctionnelle non redondante du système,
. partition du système en sous-ensembles,
. stratégie de tolérance aux pannes de chaque sous-ensemble,
. méthode d'implantation de ces stratégies.

Nous espérons que ce système de description constituera pour le concepteur un guide à trois usages possibles :

. il peut lui permettre de situer son projet par rapport à des réalisations antérieures,

. il peut lui servir d'outil d'analyse "a posteriori" de son système : cette analyse précède l'évaluation chiffrée et l'oblige à partitionner clairement son système et à décrire soigneusement les réactions de chaque sous-ensemble aux évènements de pannes

. enfin, il peut servir de base à une démarche de conception, chaque rubrique de description constituant une étape de conception (sans pour autant que le processus de conception ne se résume qu'en un seul passage dans chacune des quatre étapes proposées).

Ce chapitre comprend trois paragraphes principaux :

. le premier présente le système de description proposé,

. dans le deuxième paragraphe, nous traitons l'application de ce système à différentes machines existantes

. dans le troisième paragraphe, nous proposons une démarche de conception basée sur ce système de description et nous discutons des manières de suivre cette démarche.

II.2 - SYSTEME DE DESCRIPTION PROPOSE

Ce paragraphe présente les choix (et leurs caractéristiques) correspondant aux quatre rubriques sur lesquelles s'appuie le système de description proposé.

II.2.1. Caractéristiques fonctionnelles de la machine décrite

Il s'agit de décrire les caractéristiques de l'architecture non redondante à partir de laquelle est construite la machine tolérant les pannes.
Ces caractéristiques relèvent en général de choix non techniques (économiques notamment), ou de choix techniques indépendants de la tolérance aux pannes (prise en compte des critères de performances "classiques" : vitesse nécessaire, capacité mémoire...).

Deux grandes classes apparaissent tout d'abord : les mono processeurs (ou mono calculateurs) et les multiprocesseurs (ou multicalculateurs).
En effet, la seconde classe regroupe des machines très différentes qu'il convient de bien caractériser, notamment du point de vue de la sûreté de fonctionnement : il s'agit de décrire les caractéristiques du système de communication entre processeurs.
Cette caractérisation pourra s'appuyer sur la taxinomie proposée par Anderson et Jensen [AND75], fondée sur les critères suivants :

- Communication directe ou indirecte : elle est indirecte si un organe de communication modifie ou aiguille les messages.
- Communication centralisée ou décentralisée : cela est sans objet dans le cas direct ; dans le cas indirect, il s'agit de savoir si l'organe de communication cité ci-dessus est unique ou distribué.
- A chemins spécifiques ou partagés ("dédicated-shared").

Nous reproduisons à la figure 1 l'arbre de classification ainsi obtenu. Cette classification prend néanmoins en compte d'autres facteurs que la tolérance aux pannes ; dans le cadre de celle-ci, il nous semble que l'on peut se restreindre à distinguer trois classes :

a) *Directe-spécifique* : l'architecture apparaît comme un réseau de calculateurs reliés par des lignes de communication. Nous considèrerons aussi dans cette classe les réseaux non complets (exclus par Anderson et Jensen qui, par hypothèse, n'envisagent que les systèmes de communication où tout calculateur peut communiquer avec tout autre) : ces réseaux non complets correspondent au cas où l'on désire construire un système non pas d'usage général mais adapté à une application particulière [CAS81a].

b) *Directe-partagée* : cette classe se distingue par le fait qu'il existe des chemins de communication partagés qui devront être protégés contre les pannes et d'autre part des connexions directes des agents communiquant sur ces chemins avec des possibilités de pannes de chemins induites par des pannes de ces agents.

c) *Indirecte* : dans tous ces systèmes, il existe des organes de communication à protéger contre les pannes. En revanche, ce cas se distingue du précédent par le fait que ces organes isolent entre eux les agents communiquant du point de vue de leurs pannes.

En conclusion, la description de ces caractéristiques devra clarifier le problème de l'isolation/propagation des pannes dans les communications : il s'agit, en quelque sorte, de répondre à la question : qui a le pouvoir d'agir sur les données en cours de transmission ?

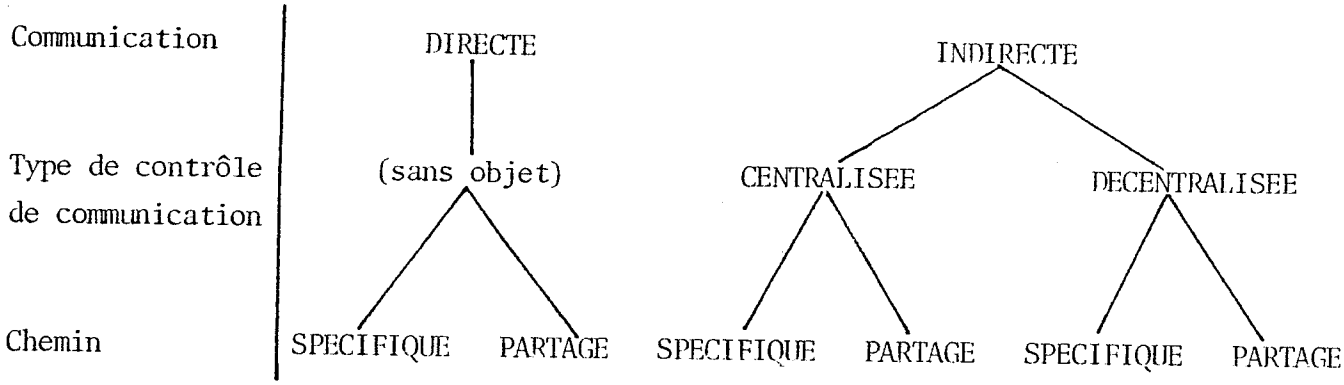


Figure 1 . Critères de classification des multiprocesseurs d'après Anderson et Jensen

II.2.2. Partition du système en sous-ensembles

II.2.2.1. Caractérisation

Ce critère se justifie vis-à-vis des deux objectifs adoptés pour cette classification : du point de vue description, la représentation du système doit mettre en évidence les sous-ensembles d'unités impliqués dans une même stratégie de tolérance aux pannes ; du point de vue conception, le concepteur doit répartir les divers éléments qui apparaissent dans la description fonctionnelle de son système, en sous-ensembles pour lesquels il devra choisir des stratégies de tolérance aux pannes.

Pour caractériser ces partitions, il convient tout d'abord de définir les éléments que l'on peut trouver dans les sous-ensembles.

Nous définirons :

- un calculateur : c'est un ensemble matériel capable d'exécuter des programmes inscrits dans certains de ces éléments, d'acquérir et de délivrer des informations à l'extérieur de lui-même ; il contient en général des mémoires, des processeurs, une horloge, des interfaces d'entrée/sortie ;

- un processeur : c'est un ensemble matériel capable d'exécuter des instructions qu'il acquiert de l'extérieur et d'afficher vers l'extérieur les résultats de ces instructions ; la base de temps (horloge) peut lui être externe ou interne ;
- un système de communication : c'est un ensemble capable de transmettre des informations entre des calculateurs ou entre des processeurs et des mémoires ;
- enfin, les sous-ensembles pourront contenir tous les autres éléments d'un calculateur ayant des activités fonctionnelles ou de tolérance aux pannes : horloge, alimentation, comparateurs, voteurs, codeurs, décodeurs, etc... que l'on désignera sous le nom de circuits.

Il reste maintenant à caractériser les sous-ensembles en fonction de ces éléments. Pour obtenir une caractérisation simple, on remarquera qu'il existe une relation d'ordre partiel entre ces éléments : on peut construire un calculateur à partir de processeurs, mémoires et systèmes de communication. De même, ces derniers peuvent être construits à partir des circuits plus simples. On obtient ainsi le treillis suivant (Figure 2).

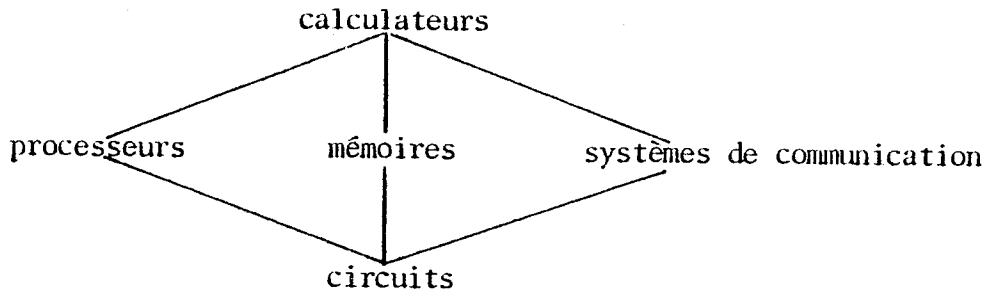


Figure 2 : hiérarchie entre les éléments d'un système.

On caractérisera alors un sous-ensemble par le nom de l'élément le plus grand qu'il contient. Dans le cas de sous-ensembles composés de circuits fonctionnels

et de tolérance aux pannes, seul le nom des circuits fonctionnels sera indiqué. Ainsi, un sous-ensemble composé de calculateurs et de voteurs sera appelé sous-ensemble (calculateur), un sous-ensemble composé d'horloges et de voteurs sera appelé sous-ensemble (horloge). La raison de cette simplification apparaîtra lors de la discussion des implantations des stratégies.

II.2.2.2. Caractérisation "multiniveaux"

Une autre justification de l'introduction d'une relation d'ordre partiel entre les éléments provient de la nécessité, pour certains systèmes, de décrire les stratégies de tolérance aux pannes sur plusieurs niveaux : certains éléments peuvent être regroupés pour suivre une stratégie globale mais suivre individuellement (à leur niveau) une stratégie particulière : ces stratégies implantées au niveau des éléments permettent de limiter la proportion des pannes devant être prises en compte par la stratégie globale.

Ainsi, par exemple, la caractérisation à deux niveaux définie ainsi :

Système : calculateur . organe de sortie

Calculateur : processeur . mémoire RAM . mémoire ROM . organe d'entrée

indique que les éléments du sous-ensemble calculateur (processeur . mémoire RAM, mémoire ROM, organe d'entrée) suivront une même stratégie mais que, à l'intérieur de ce sous-ensemble, chaque élément suivra une stratégie propre, permettant de supprimer la "contribution" de certaines pannes de ces éléments à la stratégie globale.

Cette notion de stratégies sur plusieurs niveaux sera encore développée lors du paragraphe sur la description des stratégies (II.2.3.3.).

II.2.2.3. Influence de la sûreté de fonctionnement sur les partitions

Les progrès des techniques d'intégration ont eu pour effets, outre les gains de volume et de consommation, d'accroître considérablement la fiabilité intrinsèque des matériels. Dans le domaine d'application considéré ici, on

utilise donc essentiellement des composants très intégrés, réalisant donc souvent une unité fonctionnelle entière ; ceci explique que les partitions seront surtout fondées sur les sous-ensembles suivants :

- {calculateur}
- {calculateur} {système de communication}
- {processeur} {mémoire} {système de communication}

ainsi que l'usage limité de sous-ensembles du type {circuits}.

II.2.3. Stratégies de tolérance aux pannes

Pour permettre à une machine de "survivre" à une ou plusieurs pannes, il est nécessaire d'ajouter certaines redondances à la structure fonctionnelle choisie pour réaliser l'application : redondances dans l'architecture (redondances matérielles), dans les programmes (redondances logicielles) et dans les traitements eux-mêmes (redondances temporelles). Ces redondances sont à la base des "stratégies" de tolérance aux pannes.
Dans une première partie, nous définirons ces redondances, puis nous nous intéresserons, dans une deuxième partie, à ces stratégies.

II.2.3.1. Redondances employées

a) Redondance matérielle massive

On définira comme telle la multiplication de modules entiers (processeurs, mémoires, bus, unités d'entrée sortie...). Ces modules redondants serviront aussi bien pour assurer la détection (comparaison des sorties de deux modules identiques) ou le masquage des pannes (vote majoritaire entre les sorties de plusieurs modules identiques), que pour assurer le remplacement d'unités défaillantes.

b) Redondance matérielle sélective (ou partielle)

On définira comme telle l'adjonction dans une unité d'une partie de matériel supplémentaire : ce type de technique comprend l'utilisation de codes détecteurs ou correcteurs qui permettent la détection ou le masquage de pannes dans les organes de mémorisation ou de transmission (adjonction de matériel pour le support des bits supplémentaires du codage [AVI71]) ; il comprend également l'utilisation de circuits rendus "autotests" (par l'adjonction d'une "circuiterie" spéciale en différents points, permettant ainsi la détection de certaines classes de pannes [DIA74], [CRO78]).

c) Redondance logicielle sélective

On définira comme telle l'implantation de programmes de test exécutés lors de périodes d'oisiveté ou en cours de fonctionnement [BEL77] [ROB79]. Ces techniques permettent une détection et un diagnostic de pannes dont l'efficacité est difficile à chiffrer. Notons que l'exécution de tests en cours de fonctionnement peut être également qualifiée de redondance temporelle dans la mesure où elle ralentit l'exécution de la tâche traitée.

d) Redondances utilisées pour la survie aux pannes transitoires

Les pannes transitoires constituent une part importante des cas de pannes d'un système [MAC79]. Il est souvent indispensable de permettre à une unité présentant une telle panne de retrouver un état correct (ou cohérent) afin de reprendre sa tâche normalement (la persistance de l'apparition de cette panne après un nombre fixé d'essais est alors interprétée comme la manifestation d'une panne définitive). Pour ce faire, l'unité doit recharger un "contexte" à partir duquel la reprise de la tâche est possible. Les problèmes liés à la gestion de ce contexte (implantation de "points de reprise" dans les programmes, contenu d'un "contexte"...[ROH73][YOU74][CHA76][MER76], ne seront pas

discutés ici. On se contentera de donner, pour chaque stratégie, une idée de la complexité et de l'utilité de cette gestion. Des mécanismes analogues permettent, dans les stratégies avec remplacement, "d'initialiser" une réserve en lui donnant le même contexte que les unités actives).

II.2.3.2. Description et caractéristiques des stratégies

Nous distinguerons trois types de stratégies :

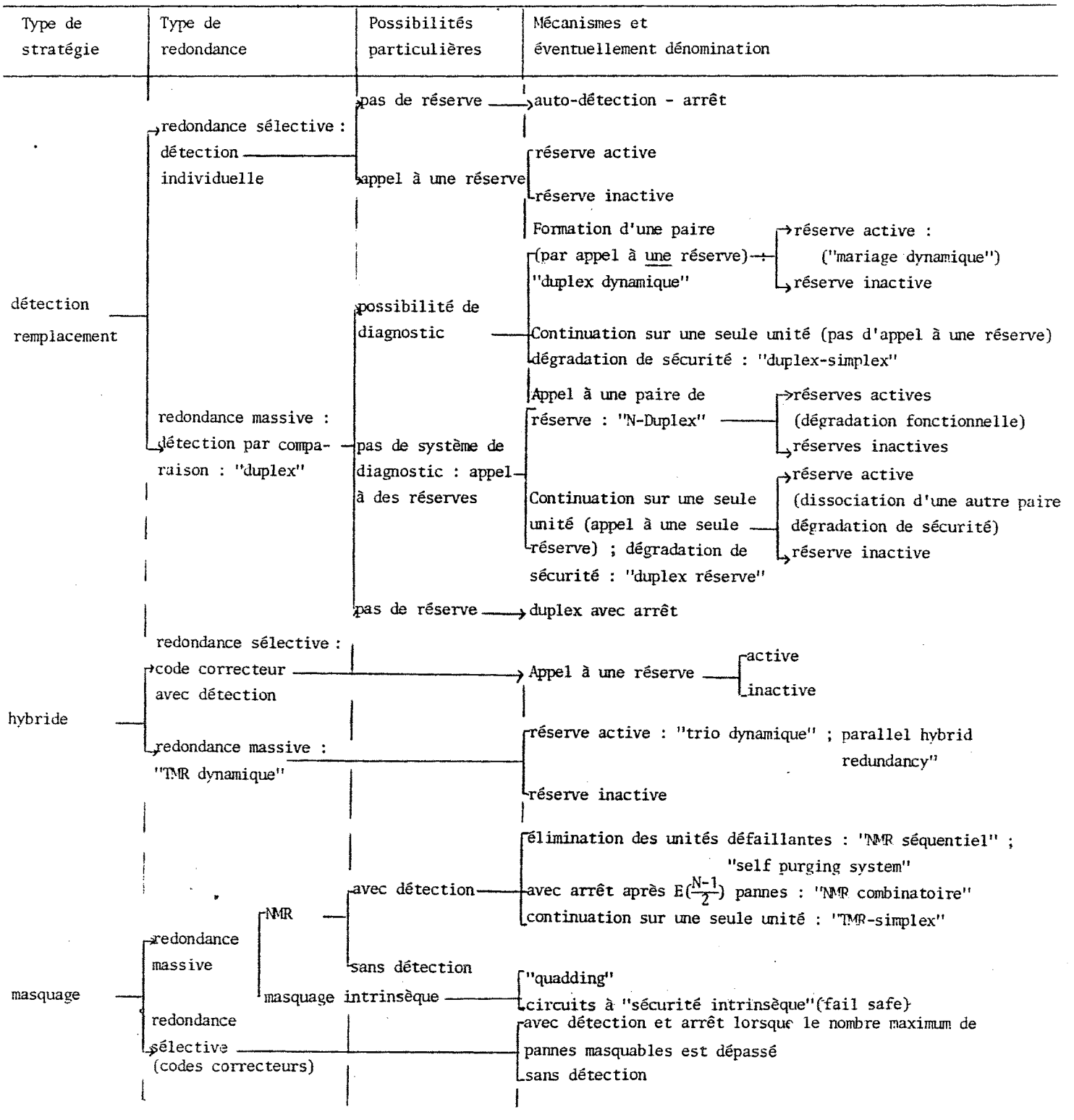
. les stratégies basées sur la détection des pannes et le remplacement de l'unité défaillante,

. les stratégies basées sur le masquage des pannes,

. les stratégies hybrides (masquage des pannes, détection et remplacement de l'unité défaillante).

La figure 3 résume les stratégies que nous détaillerons ci-après.
Nous remarquerons que les stratégies sur lesquelles sont fondées les machines du domaine d'étude, sont essentiellement basées sur les redondances massives : les autres techniques ne sont, dans cette gamme de machine, utilisées que comme techniques d'appoint destinées à améliorer légèrement la détection ou le masquage dans un sous-ensemble de la machine ; ce rôle limité tient à diverses raisons : les codes, appliqués aux mémoires et aux bus, nécessitent souvent un ensemble de boitiers de plus faible intégration et présentent une plus faible fiabilité que des unités fonctionnelles entières, souvent complètement intégrées ; les techniques d'autotest ou de test ont une efficacité parfois difficile à déterminer et sont, de toutes façons, souvent insuffisantes pour permettre d'atteindre à elles seules les objectifs fixés.

Figure 3 : RESUME DES STRATEGIES DE TOLERANCE AUX PANNES



II.2.3.2.1. Stratégies "détection remplacement"

a) Description

Les stratégies diffèreront sensiblement suivant le mode de détection employé : lorsque cette détection est assurée par redondance sélective (logicielle ou matérielle), chaque unité est soumise individuellement à la détection : à l'apparition d'une panne, on peut diagnostiquer quelle unité est en panne et la remplacer immédiatement ; lorsqu'elle est assurée par comparaison entre deux unités complètes ("duplex"), de nombreuses possibilités existent, suivant qu'on dispose ou non d'un moyen de diagnostic permettant de déterminer l'unité en panne de la paire : en l'absence de diagnostic, il faudra remplacer la paire complète, soit par une autre paire en réserve (stratégie "bi-duplex" et plus généralement "N-duplex" [COU76]) soit par une unité de réserve (il y a alors dégradation de la sécurité puisque les résultats délivrés par l'unité ne sont pas soumis à une comparaison : "duplex réserve") ; si un diagnostic est possible, on pourra n'éliminer que l'unité en panne et conserver l'unité valide de la paire, soit pour former une nouvelle paire avec une unité en réserve (on appellera ce type de stratégie "duplex dynamique") soit pour lui faire continuer la tâche en "simplex" (il y a encore dégradation de sécurité : "duplex-simplex").

Notons enfin que les unités dites "en réserve" pourront être inactives ou actives sur une autre tâche : lorsqu'elles sont actives, l'apparition de pannes se traduira par une dégradation fonctionnelle (ou une dégradation de performances) (bi-duplex ou N-duplex avec tâches prioritaires, "duplex dynamique" avec "mariages dynamiques") ou par une dégradation de sécurité répartie sur deux tâches (cas où le "duplex-réserve" est réalisé en dissociant un deuxième duplex pour fournir la réserve : on forme alors deux "simplex").

b) Caractéristiques

Les caractéristiques générales des stratégies "détection remplacement" semblent pouvoir se résumer en deux points :

. Elles sont tout à fait adaptées lorsque l'objectif est d'arrêter le système à l'apparition d'une panne (minimum d'unités mises en jeu, simplicité des mécanismes de détection).

. Elles semblent lourdes à mettre en oeuvre lorsque les objectifs imposent la survie à plusieurs pannes, c'est-à-dire la réalisation automatique de remplacements :

- nombre d'unités important dans le cas où il n'y a pas de diagnostic (N-duplex),
- lourdeur des mécanismes spécifiques pour réaliser le diagnostic de l'unité en panne,
- dans tous les cas, il y a nécessité d'un retour arrière ("rollback") à l'apparition de panne : pour survivre à une panne transitoire, il faut réexécuter la tâche à partir du dernier point de reprise, ou, pour initialiser la ou les réserves, il faut revenir au dernier état correct, c'est-à-dire revenir au dernier point de reprise (dans certains cas, il suffit de revenir à un état cohérent pour permettre une reprise satisfaisante de la tâche) : la gestion des sauvegardes de contexte est un point fondamental (et difficile !) dans l'utilisation de telles stratégies.

II.2.3.2.2. Stratégies "masquage"

a) Description

Ces stratégies sont basées sur une redondance matérielle massive ou sélective. Les stratégies à base de redondance massive sont appelées N.M.R. (N.Modular Redundancy) et sont réalisées par un vote majoritaire sur les

sorties de N modules ; la stratégie N.M.R. minimale en nombre de modules est la stratégie T.M.R. (vote majoritaire sur trois modules).

Les stratégies à base de redondance sélective sont les codes correcteurs, appliqués à des organes de mémorisation ou de transmission de données. Dans tous les cas, ces stratégies pourront être implantées avec ou sans organe de détection (et localisation) de pannes. En l'absence de système de détection, l'arrêt de la mission (inhibition des sorties, génération de signaux d'alarme...) ne pourra pas être décidé lorsque le nombre limite de pannes "masquables" est atteint (au moins $E(\frac{N-1}{2})$ pannes dans un N.M.R.) ; au contraire, un système de détection permet cet arrêt (stratégies avec arrêt).

Dans le cas des stratégies NMR, on peut facilement concevoir un système de détection localisation (le module défaillant est en désaccord avec les autres, c'est-à-dire avec le résultat du vote) : on peut alors concevoir une stratégie qu'on appellera N.M.R. séquentiel (par opposition au N.M.R. simple où il n'y a pas de mémorisation des occurrences de panne : celui-ci sera appelé NMR combinatoire) ou "self Purging Redundancy" [LOS76] : il s'agit d'éliminer, à chaque apparition de panne, le module défaillant et de transformer ainsi la stratégie NMR en stratégie (N-1). M.R. puis (N-2). MR jusqu'à la stratégie TMR avec arrêt (voire jusqu'à la stratégie duplex) ; il est alors possible de survivre à (N-2) pannes (voire (N-1) pannes). Ce système de détection peut également permettre de donner l'alarme à l'apparition d'une panne et de continuer la tâche sur une des unités valides (dégradation de sécurité : "TMR-simplex" [MAT71].

Enfin, il convient de citer une classe de stratégies que nous regrouperons sous le terme de "stratégies à masquage intrinsèque" qui s'appliquent plus fréquemment au niveau de la porte ou de petites unités, mais qui peuvent être classées comme redondances massives puisque des unités entières sont ajoutées : il s'agit de rendre impossibles certains effets de pannes par l'usage d'une technologie adaptée ; on se prémunira ainsi contre certains types de pannes sur des signaux par la mise en parallèle et en série

Dans certains cas, ces techniques sont employées pour ne se prémunir que contre un mode de pannes, catastrophique ; nous parlerons alors de "circuits à sécurité intrinsèque" ("fail safe logic" [MIN67]) ou de "quadding partiel" .

b) Caractéristiques

Les stratégies NMR sont caractérisées par une relative simplicité de gestion :

- pas d'appel à des réserves (pas de problème de "switch", d'initialisation de réserves),
- gestion aisée de contexte : pour survivre aux pannes transitoires, une unité peut recopier l'état correct sur la majorité des unités (sans retour arrière),
- localisation facile de la panne : c'est la détection d'un désaccord entre l'unité et la sortie du vote.

En contrepartie, ces stratégies sont coûteuses en matériel (nombre important d'unités, unités inutilisables pour d'autres tâches comme dans les stratégies avec réserve).

II.2.3.2.3. Stratégies "hybrides"

a) Description

Le but de telles stratégies est de concilier les avantages des stratégies de détection remplacement (économie de matériel) et les stratégies de masquage (souplesse de gestion). Ces stratégies utiliseront les dispositifs de base des stratégies de masquage : codes correcteurs (redondance sélective) ou TMR (redondance massive). Un dispositif de détection permet d'éliminer l'unité en panne et de faire appel à une unité de réserve. Cette unité de réserve pourra être active ou inactive ; dans le cas où les unités de réserves

sont actives, ces stratégies sont nommées TMR dynamiques ou "trio dynamiques".

b) Caractéristiques

Les stratégies hybrides de redondance massive (TMR dynamique) permettent une utilisation plus économique du nombre de modules que les stratégies NMR, mais sont moins simples à gérer (réalisation de la fonction "switch", initialisation des réserves). Elles restent cependant plus simples que les stratégies de détection remplacement (pas de retour-arrière, localisation plus aisée de l'unité en panne).

II.2.3.3. Stratégies "multiniveaux"

La caractérisation multiniveaux proposée précédemment permet de simplifier la description des stratégies choisies dans un calculateur : il arrive que des stratégies annexes soient implantées pour certains éléments d'un sous-ensemble pour améliorer l'efficacité de la stratégie appliquée sur ce sous-ensemble. La caractérisation proposée permet une description claire de ces stratégies en les faisant apparaître chacune à leur niveau : ainsi, dans l'exemple déjà étudié, on pourra obtenir la caractérisation suivante non ambigüe :

PARTITION : Système : calculateur - organe de sortie
Calculateur : processeur - mémoire RAM - mémoire ROM - organe de sortie

STRATEGIE : Calculateur : duplex
organe de sortie : simplex
processeur : simplex
mémoire RAM : duplex
mémoire ROM : détection par "check-sum" périodique
organe sortie : duplex

Cette caractérisation indique notamment que le système est constitué de deux calculateurs et d'un organe de sortie et que, à l'intérieur de chaque calculateur un certain nombre de dispositifs de détection de pannes sont implantés : le processeur ne suit pas de stratégie particulière, la mémoire RAM et l'organe de sortie sont dupliqués alors que certaines pannes de la mémoire ROM sont détectées par check sum périodique. Un exemple d'implantation d'un tel système est donné dans la figure 4 .

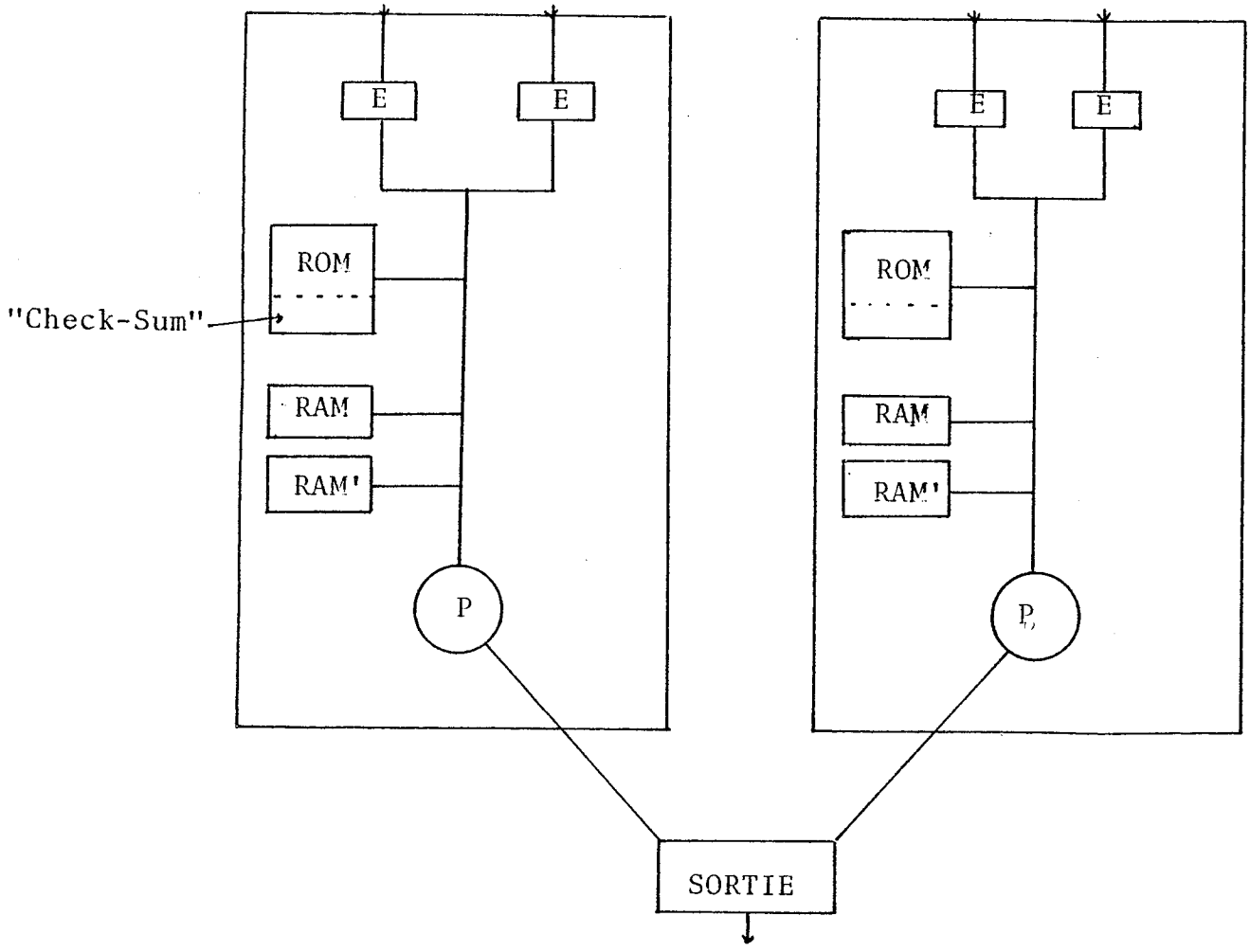


Figure 4 : Exemple de système nécessitant une description à deux niveaux.

II.2.4. Implantation

Les implantations possibles des stratégies décrites précédemment peuvent être caractérisées et différenciées suivant deux critères principaux, que nous décrirons dans ce paragraphe :

. Implantation logicielle ou matérielle des mécanismes de survie aux pannes,

. Localisation de ces mécanismes dans la partition du système.

II.2.4.1. Choix d'une implantation logicielle ou matérielle

Les mécanismes à implanter sont les fonctions de détection, de masquage et de reconfiguration (c'est-à-dire localisation de la panne et "switch").

a) Détection et masquage

Les mécanismes réalisant ces fonctions relèvent du même principe et peuvent être réalisés de manière similaire.

* Solutions matérielles

Elles consistent en la réalisation de circuits générateurs ou détecteurs de codes, de dispositifs "chien de garde" ("watch-dog" : ces dispositifs permettent de tester la bonne exécution temporelle d'une tâche) dans le cas des redondances matérielles sélectives.

Dans le cas des redondances matérielles massives, elles consistent en la réalisation de circuits de vote (fonction "majorité" entre les valeurs à voter) ou de comparaison (la fonction "désaccord" est la "disjonction" entre les valeurs à comparer).

* Solutions logicielles

Les fonctions décrites précédemment peuvent être assurées par programme : dans le cas des codes, l'acquisition (ou la production) d'une donnée est accompagnée par l'exécution d'un programme de décodage (ou de codage) ; dans le cas d'un vote ou d'une comparaison logicielle, chaque unité du système redondant exécutera, après synchronisation avec les autres unités et

lecture de leurs résultats, un programme de vote ou de comparaison ; le problème important est de déterminer le temps séparant deux exécutions du programme de vote (ou de détection) en fonction des objectifs de performance (un temps trop court pénalise la vitesse d'exécution du programme d'application) et de sûreté (une trop grande "latence" de détection favorise l'occurrence de pannes multiples qui mettent en échec la plupart des stratégies par détection-remplacement.

Enfin, il convient de citer, parmi ces solutions logicielles, les mécanismes qui relèvent du domaine du test : détection en ligne (exécution d'instructions de vérification d'invariants de l'application) ou détection périodique en oisiveté (exécution, pendant des périodes où le programme d'application est "stoppé", de programmes de test destinés à détecter les pannes latentes).

b) Reconfiguration (localisation-"Switch")

Ces mécanismes doivent assurer la localisation de la partie en panne, son élimination (ou son isolation) et l'appel à une réserve.

Leur réalisation est très liée à la réalisation des mécanismes de masquage ou détection (la localisation de la panne est un problème trivial dans le cas des stratégies de masquage mais difficile dans les cas de simple détection) ; au niveau du système entier, la localisation est liée aux possibilités de confinements des pannes, c'est-à-dire à la capacité de partitionner clairement le système.

Les solutions matérielles consistent en la réalisation de fonctions booléennes simples (organes de disjonction, "switch") ; elles sont souvent reliées à un "noyau" logiciel : mémorisation de l'apparition antérieure de phénomènes supposés transitoires, initialisation des unités de réserve...

Les solutions logicielles reviennent à faire assurer par chaque unité la mise à jour d'une table d'allocation des tâches, éventuellement à partir des résultats de programmes de localisation exécutés par certaines unités. Le problème fondamental est la protection du système contre les unités défaillantes : il faut prévoir des frontières logiques lors de la définition de l'architecture (protection des mémoires...) et physiques (par l'utilisation d'un noyau matériel) à la propagation d'erreur.

c) Caractéristiques

Les solutions matérielles sont caractérisées par la simplicité de la conception logique des mécanismes, mais sont limitées par leur manque de souplesse. De plus, leur utilisation se heurte souvent à deux problèmes :

- elle nécessite l'emploi de composants peu intégrés : en effet, les quelques circuits intégrés spécialisés disponibles prennent rarement en compte toutes les spécificités de l'application (cas limites particuliers, caractéristiques temporelles spéciales...). Outre les conséquences que cela entraîne sur les coûts d'étude et de mise au point, nous noterons que le concepteur est alors conduit à introduire dans le système des boitiers de faible intégration, c'est-à-dire de fiabilité intrinsèque peu satisfaisante. Cette conséquence est inacceptable lorsque ces organes sont uniques et constituent un "point dur" de la machine (c'est-à-dire qu'une panne d'un de ces organes entraîne l'échec de la mission de la machine). Le concepteur devra alors veiller à réaliser ces organes dans une technologie réputée sûre, mais en général coûteuse : circuits électroniques à sécurité intrinsèque (fail-safe-logic), technologie à relais, voteurs mécaniques...
- elle pose un problème de synchronisation :

si le comparateur ou le voteur sont de simples circuits combinatoires, les unités en redondance doivent être synchronisées bit à bit afin que les

entrées de ces circuits combinatoires soient toujours cohérentes, c'est à dire que les valeurs à comparer soient présentes en même temps (sous peine de détecter à tort une erreur) ; la conséquence est double : d'une part, une horloge commune est nécessaire et peut constituer un point dur ; d'autre part, en cas d'occurence de phénomènes parasites, on peut craindre que deux unités identiques,synchronisées, donc dans le même état, réagissent de la même manière à ce phénomène et produisent une erreur multiple ; une autre solution consiste à utiliser des comparateurs ou voteurs séquentiels "intelligents" qui acceptent un décalage maximum entre les arrivées des valeurs à comparer ; la compléxité de tels circuits constitue alors un inconvénient important.

Les solutions logicielles, souples d'utilisation, se heurtent à deux objections :

- l'importance de la charge supplémentaire de traitement ("overhead") qu'elles apportent peut se révéler insupportable pour respecter les contraintes de l'application ;
- lorsqu'on les oppose aux solutions matérielles et que l'on met en avant les éléments matériels qu'elles permettent d'économiser, c'est à dire la baisse de taux de pannes qui en résulte, la question de la validité de ces mécanismes logiciels est immédiatement posée (certains parlent de "taux de pannes" de ce logiciel). La discussion de cette question est abordée dans le chapitre VIII de ce mémoire.

II.2.4.2. Localisation et action des mécanismes dans la partition

Dans le cas de stratégies implantées par matériel, il s'agit de situer les organes de gestion de ces stratégies dans la partition du système : ces organes peuvent constituer eux-mêmes des sous-ensembles suivant une stratégie

propre, ou être associés à d'autres sous-ensembles (c'est-à-dire qu'ils suivent la stratégie du sous-ensemble auquel ils sont associés).
Dans le cas des stratégies implantées par logiciel, il s'agit de préciser le sous-ensemble dans lequel sont exécutés les programmes de gestion de ces mécanismes.
Deux grandes classes de localisation d'un mécanisme apparaissent alors : les mécanismes internes (le mécanisme de gestion de la stratégie d'un sous ensemble est lui-même dans ce sous-ensemble), et les mécanismes externes (la gestion de la stratégie d'un sous-ensemble est effectuée dans un (ou plusieurs) autre(s) sous-ensemble(s)). Remarquons alors que certains choix de conception se révèlent incompatibles au niveau de cette localisation :

a) "Impossibilité" des implantations matérielles internes

Soit un sous-ensemble suivant une stratégie donnée, gérée par un mécanisme matériel. Pour que ce mécanisme soit interne, il faut qu'il agisse sur les sorties du sous-ensemble pour rendre celles-ci conformes aux spécifications de la stratégie (sinon le dernier élément en sortie qui échapperait à son action ne pourrait pas être considéré comme appartenant au sous-ensemble) ; les seules exceptions qu'il convient de mentionner pour cette propriété sont les cas limites suivants :

- les mécanismes employés au niveau fin ("quadding", circuits autotests) : ils ne comportent pas à proprement parler d'organe de décision,
- les sous-ensembles ne délivrant pas de sortie : ce peut être le cas d'un sous-ensemble {calculateur} (cas particulier car la mission n'est alors pas définie par les sorties, voir II.3.1.2.).

En dehors de ces exceptions, les organes gérant le mécanisme considéré contribuent donc à l'élaboration des entrées d'un sous-ensemble situé en aval (éventuellement le monde extérieur), soit en les générant (voteur sur

les données), soit en les validant (organe de détection) ; il existe nécessairement des pannes de ces organes provoquant la sortie de résultats erronés et donc assimilables à un mauvais fonctionnement du sous-ensemble situé en aval, ou de tout le système : une panne de voteur a le même effet qu'une panne de l'élément situé en aval alors qu'une panne d'un organe de détection (panne rendant l'organe non détectant) entraîne l'utilisation par le système d'un élément défectueux, donc la possibilité de délivrer des erreurs ; dans tous les cas, ces organes doivent être rattachés au sous-ensemble aval ou être isolés dans un sous-ensemble.

b) "Impossibilité" des implantations logicielles dans les sous-ensembles "non-intelligents"

On appellera sous-ensemble "intelligent" un sous-ensemble capable d'exécuter un programme, c'est-à-dire un sous-ensemble {calculateur}. Il est alors évident que tout autre sous-ensemble sera incapable d'assurer par programme les mécanismes gérant la tolérance aux pannes.

Remarque : la nécessité de localiser les matériels de tolérance aux pannes dans la partition justifie "a posteriori" la simplification apportée en II.2.2.1. concernant la dénomination des sous-ensembles : sauf lorsqu'un sous-ensemble ne contient que des organes de tolérance aux pannes, il est inutile de préciser ceux-ci au stade de définition de la partition puisque ceci sera précisé lors de la définition de l'implantation. Par exemple, on pourra avoir la définition suivante :

Partition : (S_1, S_2)

S_1 : Stratégie NMR, implantation matérielle externe (S_2)

S_2 : ...

Ceci indique sans ambiguité que S_2 contient des voteurs NMR qui suivent la stratégie définie pour les autres éléments de S_2.

II.3 - APPLICATION A LA DESCRIPTION D'ARCHITECTURES

II.3.1. Structures fonctionnelles "monoprocesseurs"

II.3.1.1. C.V.M.P. (Computer Voted Multi-Processor) [SIE78]

La figure 5 présente l'architecture de ce calculateur développé à l'Université de Carnegie Mellon ; il est constitué essentiellement de trois processeurs, trois mémoires et un organe de vote ; il peut fonctionner en trois modes :

- mode "indépendant" : trois couples processeur-mémoire travaillant sur des tâches indépendantes,
- mode "diffusion" : ce mode permet la diffusion d'une information à partir d'une source unique (processeur ou mémoire) vers toutes les unités réceptrices (mémoires ou processeurs),
- mode "tolérance aux pannes" : c'est à ce mode que nous nous intéresserons ici : toutes les unités travaillent sur une tâche unique, un vote étant effectué à chaque transfert entre mémoire et processeur.

Classification de ce calculateur

Fonctionnellement, dans le mode considéré, ce calculateur est un "monoprocesseur". Les choix de partition du système, de stratégies et d'implémentation sont résumés dans la figure 6 .

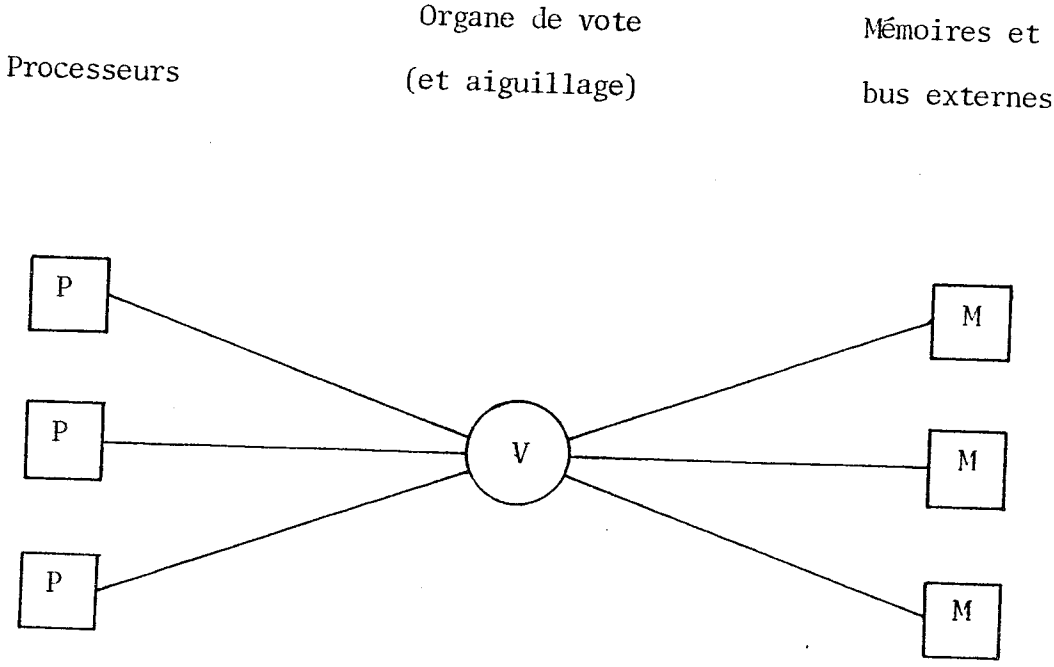


Figure 5 Architecture générale de CVMP

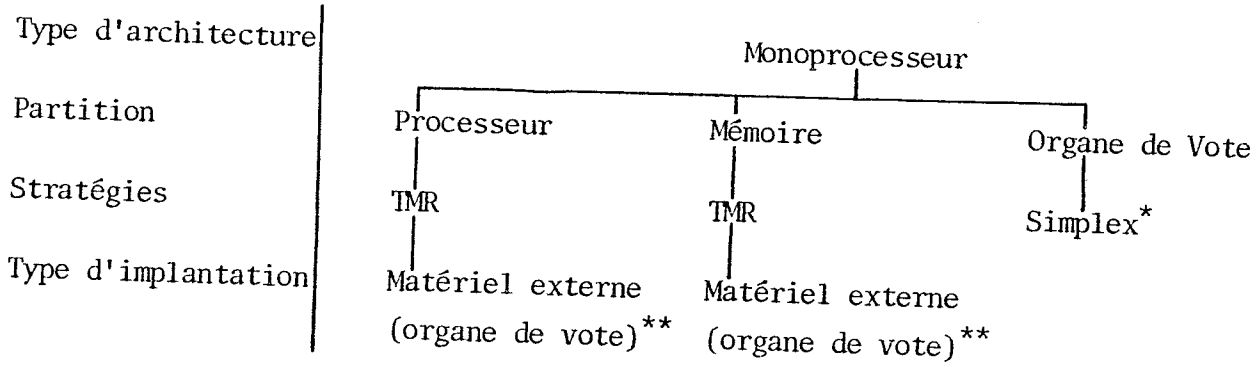


Figure 6 Description et classification de CVMP

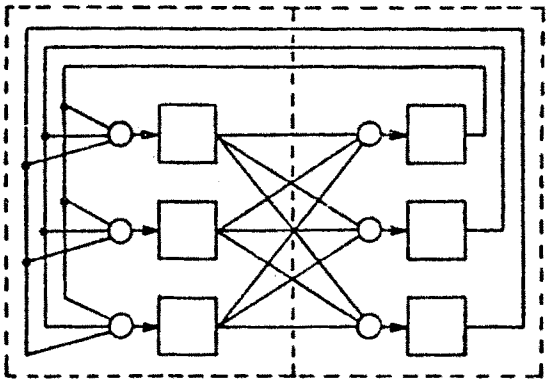
Note : * il est prévu de pouvoir tripliquer le voteur : la partition ne comprendrait alors plus qu'un sous-ensemble {calculateur} : cette architecture s'approcherait de celle proposée par Wakerly , décrite au paragraphe suivant.

** Le voteur est bi-directionnel : fonctionnellement, il y a deux votes, l'un en aval des processeurs, l'autre en aval des mémoires.

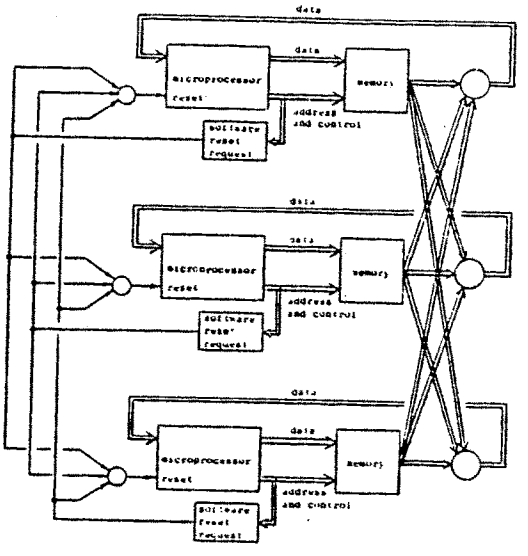
II.3.1.2. Microcalculateur T.M.R. de J.F. Wakerly [WAK76]

Dans une étude sur l'utilisation de la stratégie TMR dans les microcalculateurs, Wakerly s'intéresse à deux types d'architecture décrits dans les figures 7a et 7b et choisit la deuxième architecture qui minimise le nombre de voteurs - qui provoquent une perte de fiabilité sensible en raison de leur manque d'intégration -. Les caractéristiques de ces deux architectures sont résumées dans les tableaux a et b de la figure 8 .

On remarquera que l'option du calculateur CVMP avec trois voteurs bi-directionnels évoquée précédemment, se rapproche fonctionnellement du premier type (vote matériel en aval des processeurs et en aval des mémoires) mais se rapproche, du point de vue de l'influence des pannes (c'est-à-dire de la définition de la partition), du deuxième type (la panne d'un voteur bi-directionnel induit le mauvais fonctionnement d'une mémoire et d'un processeur, c'est-à-dire d'un calculateur. On notera enfin que le deuxième type d'architecture est un exemple de cas limite rendant possible une implantation matérielle interne (l'ensemble {calculateur} ne délivre pas de sortie).



(a)



(b)

Figure 7 : 2 types d'architecture utilisant la stratégie TMR [WAK76]

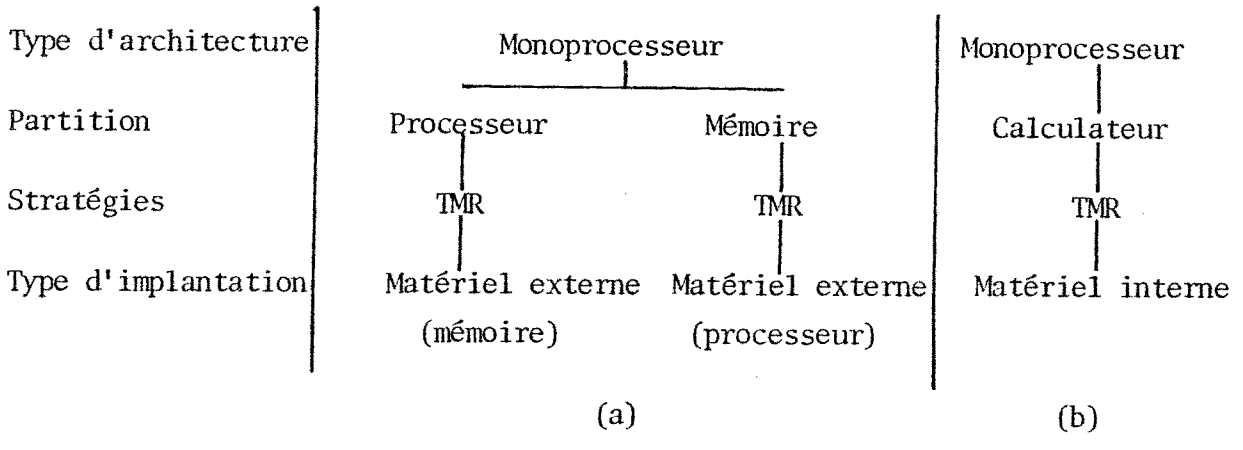


Figure 8 : Classification des architectures proposées par J.F. Wakerly

II.3.2. Structures fonctionnelles "multiprocesseurs"

II.3.2.1. COPRA (Calculateur à Organisation Parallèle Reconfigurable Automatiquement) [MER76]

L'étude de ce calculateur a été réalisée conjointement par la SAGEM (Société d'Applications Générales d'Electronique et de Mécanique) et EMD (Electronique Marcel Dassault). Fonctionnellement, COPRA est un multiprocesseur (dont le nombre d'unités peut, a priori, varier) à communication directe par mémoire partagée. Nous nous intéresserons ici à la version de base, qui apparaît fonctionnellement comme un bi-processeur.

La tolérance aux pannes de COPRA est fondée sur les principes suivants :

- Implantation matérielle de stratégies détection-remplacement (processeurs en duplex, application de code de parité au niveau des mémoires),
- Utilisation de tests périodiques pour assurer la validité des organes critiques (notamment les organes matériels de tolérance aux pannes).

Ainsi, chaque tâche est assurée par deux processeurs en duplex sur un banc mémoire où la détection est assurée par parité (une autre copie de la mémoire est située sur l'autre banc pour faire face aux problèmes de reprise) ; à l'apparition d'une panne permanente, ces unités sont éliminées (un duplex ou un banc mémoire) et, les tâches les moins critiques étant abandonnées, la mission est reprise sur les unités valides (le deuxième duplex ou le deuxième banc mémoire). La figure 9 présente l'architecture générale de COPRA.

Au niveau des communications, on dispose, pour faire face aux pannes d'allocateur, des dispositifs suivants : chaque processeur possède un dispositif de

réveil ("chien de garde") pour éviter d'être bloqué abusivement par un allocateur au cours d'une opération mémoire ; pour permettre le diagnostic en cas d'incident dans une opération mémoire, ces "chiens de garde" et les allocateurs sont testés périodiquement.

Pour constituer les sous-ensembles, on remarque que :

- les données élaborées par les processeurs sont comparées par un comparateur associé à la paire alors que les commandes mémoire élaborées par ces processeurs sont comparées par des comparateurs associés à chaque mémoire.

- les comparateurs de données et les "chiens de garde" suivent la même stratégie de détection (test périodique) remplacement (ils sont associés fonctionnellement à une paire de processeurs : toute panne de l'un de ces organes entraîne le remplacement de la paire donc des autres organes).

- de même, les allocateurs de mémoire, détecteurs de parité et comparateurs de commande, suivent la même stratégie de détection (test périodique et chien de garde : le chien de garde réveille le processeur aussi bien en cas de défaillance de l'allocateur provoquant son blocage, qu'en cas de blocage provoqué par l'autre paire, cet incident étant imputable à un mauvais fonctionnement du comparateur de commande), remplacement (ils sont associés fonctionnellement à un banc mémoire : toute panne de l'un de ces organes entraîne le remplacement du banc donc des autres organes).

Ces deux remarques introduisent la nécessité d'une partition à deux niveaux : un premier niveau groupe dans les mêmes sous-ensembles les éléments suivant la même stratégie de reconfiguration et un deuxième niveau permet de différencier à l'intérieur de ces sous-ensembles les parties suivant des stratégies de détection différentes.

- l'alimentation est fiabilisée par l'utilisation de deux alimentations suivant une stratégie de type "Quadding", une chute de tension dans l'alimentation active rendant passante une diode située en sortie de l'alimentation réserve.
- l'implantation matérielle des dispositifs rend nécessaire une synchronisation étroite entre les unités : le système commun d'horloge sera constitué par un vote majoritaire entre trois voies d'horloge (la fiabilité du voteur d'horloges est un point critique pour la fiabilité du système).

La figure 10 présente les caractéristiques de COPRA dans le système de classification retenu. On remarquera enfin que, comme nous l'avons vu précédemment, les problèmes de reprise constituent un point capital pour la validité des stratégies de détection/remplacement : le développement de COPRA a donc été lié à d'importants travaux dans ce domaine [MER76].

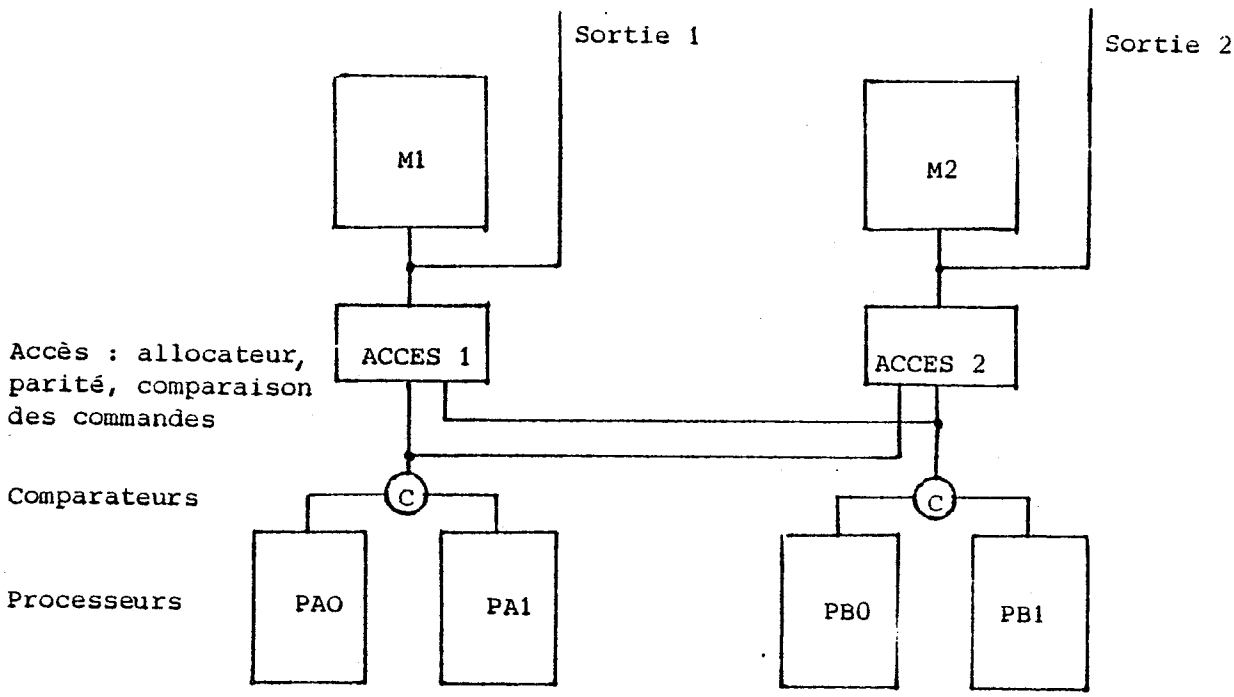
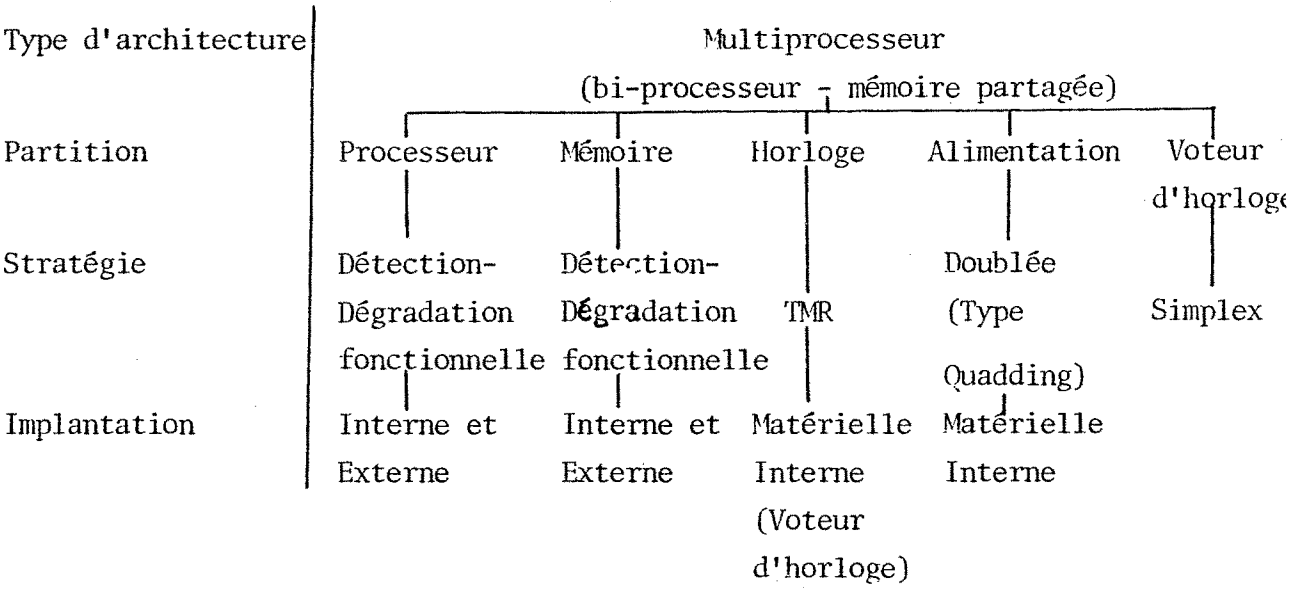


Figure 9. Architecture générale de COPRA

1er NIVEAU



2ème NIVEAU

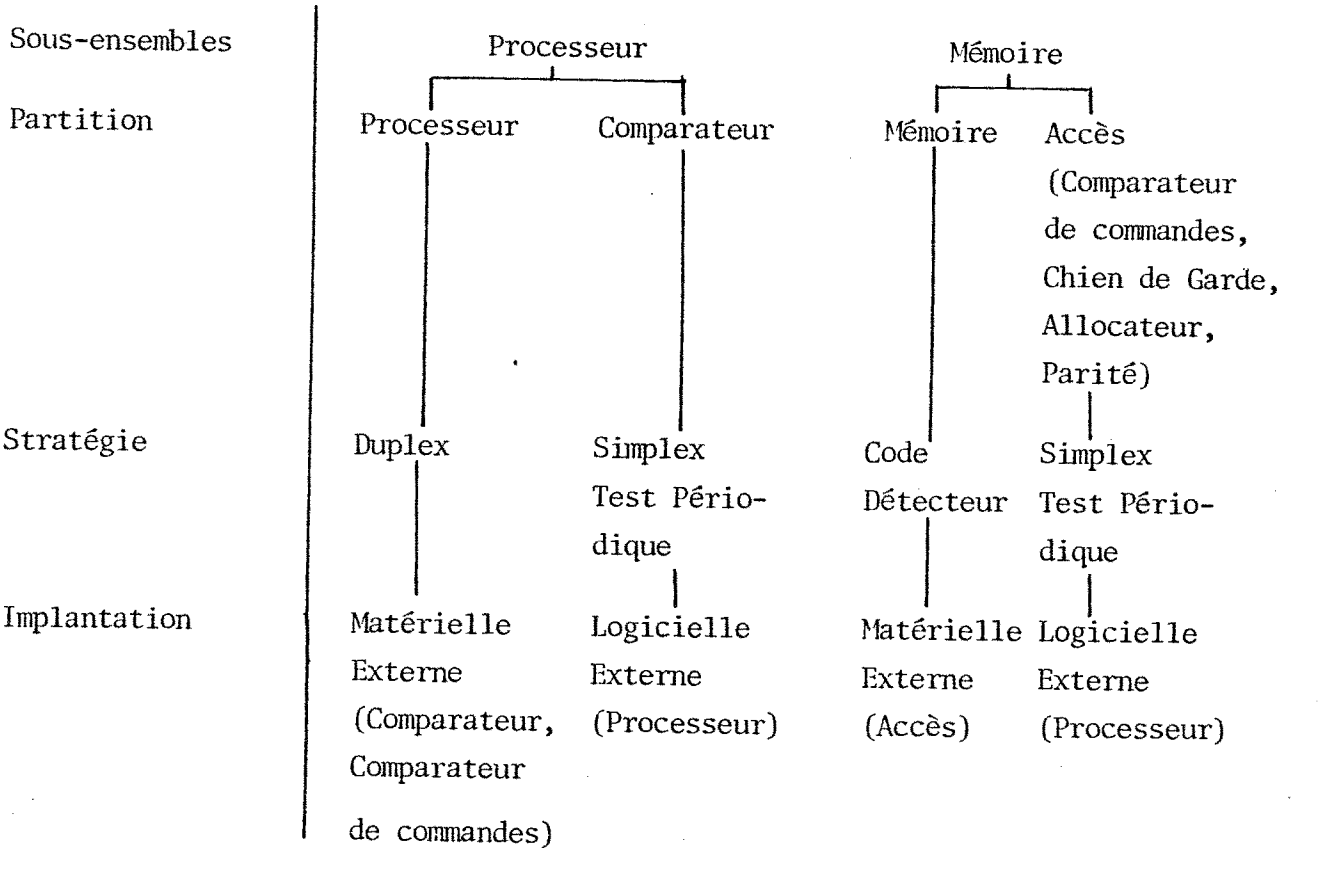


Figure 10 : Classification de COPRA, à 2 niveaux

II.3.2.3. FTMP (Fault Tolerant Multiprocessor) [HOP78]

- Caractéristiques fonctionnelles et de tolérance aux pannes

FTMP est un calculateur en cours d'étude au Charles Strak Draper Laboratory. Il est destiné aux applications avioniques et spatiales à très haute fiabilité pendant une période de 10 heures où aucune maintenance n'est possible, son taux de panne doit être inférieur à 10^{-9} heures^{-1}.

Sur le plan fonctionnel, FTMP apparaît comme étant intermédiaire entre un multicalculateur à communication par mémoire commune, et un multi-processeur à mémoire partagée. En effet, chaque processeur dispose d'une mémoire locale (ou cache) privée dont la taille peut varier au gré des applications. L'autonomie du processeur peut donc varier de zéro (pas de mémoire locale) à une autonomie totale lorsque tous les programmes et une zone de travail suffisante sont implantées en mémoire locale, la mémoire commune ne servant qu'aux communications entre calculateurs (Figure 11).

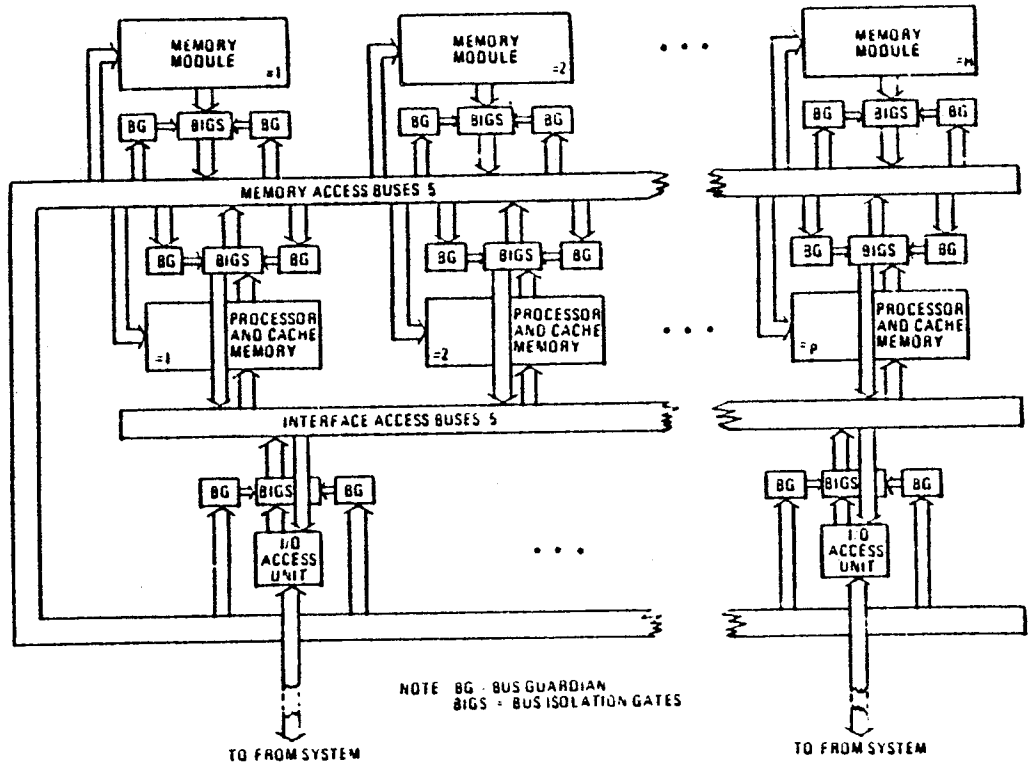


Figure 11 Architecture Générale de FTMP [HOP78]

- Principes de tolérance aux pannes

Du point de vue tolérance aux pannes, FTMP s'appuie sur deux principes fondamentaux :

1) La stratégie TMR dynamique :

A tout instant une tâche est exécutée simultanément par trois processeurs et trois mémoires connectés par trois bus (triades). A tout instant, un processeur ou une mémoire, soit appartiennent à une seule triade, soit sont en réserve. En revanche, le même bus peut appartenir à plusieurs triades : chaque bus est donc muni d'un allocateur. De plus, ces triades peuvent être modifiées dans le temps. La technique TMR a été adoptée parce qu'elle permet le masquage de pannes simples et évite donc le problème de reprise.

2) La synchronisation stricte et l'utilisation de voteurs matériels :

En fait, le choix d'une synchronisation stricte découle de celui de voteurs matériels bit à bit. Ces voteurs sont placés en entrée de chaque processeur. Ils comprennent aussi des circuits d'aiguillage programmables qui sélectionnent la triade de bus active avec l'unité (processeur ou mémoire) à laquelle ils appartiennent, en vue du vote. Ces voteurs (INMUX) (Figure 12) résolvent le problème de la transition des informations des bus vers les unités (processeurs ou mémoires). Il reste à résoudre le problème de la transition des informations des unités vers les bus.

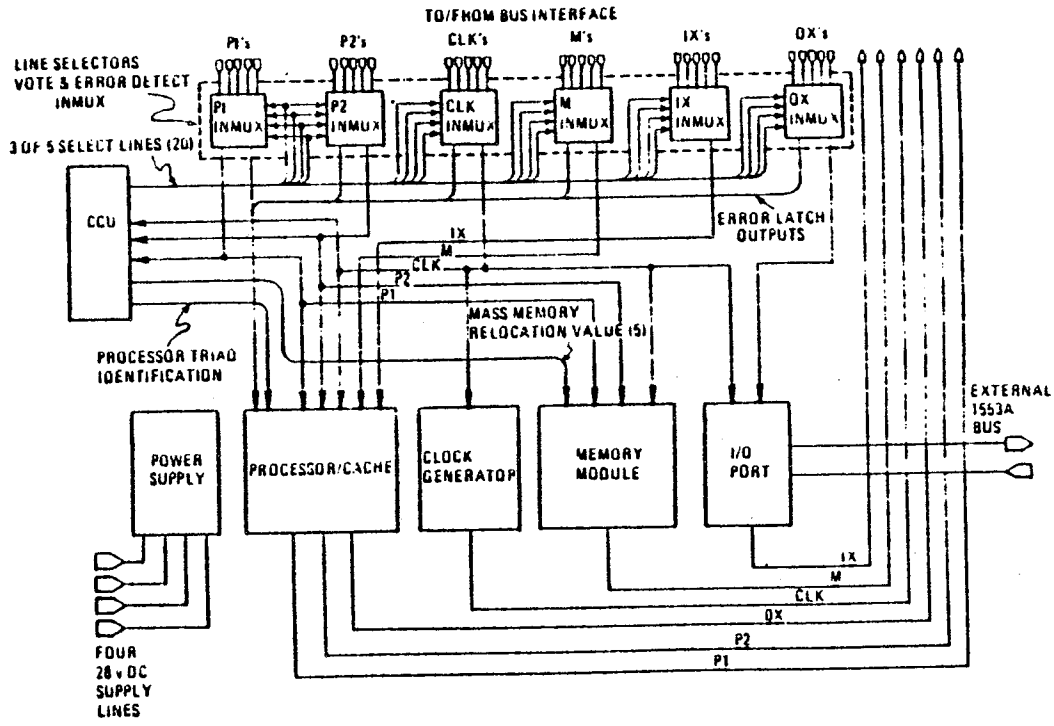


Figure 12 . Structure des votes [HOP78]

Les trois unités d'une triade émettrice émettent simultanément chacune sur un bus, chaque unité de la triade réceptrice votant sur les trois bus. Il se pose cependant à ce niveau un problème de protection : si l'unité émettrice a le pouvoir de sélectionner son bus, il est concevable que par suite d'une panne, elle sélectionne un autre bus que celui qui lui est alloué, voire tous les bus, ce qui entraînerait la perte de la machine. Le pouvoir de sélection du bus en vue d'une émission a donc été retiré à l'unité et affecté à un organe programmable, le B.G.U. (Bus Guardian Unit) : le BGU est vu par le processeur comme un élément mémoire dans lequel il inscrit le numéro de bus correspondant à son unité ; il connaît donc ce bus et peut tuer cette unité (si une majorité de processeurs est d'accord) en coupant son alimentation. Il commande les portes d'entrée des bus (BIGS) qui appartiennent fonctionnellement aux bus.

La protection contre les pannes du BGU a été résolue en utilisant une stratégie de "quadding partiel fail safe" : on remarque qu'il y a deux modes de pannes :

- le BGU ne sélectionne aucun bus : cette panne non dangereuse sera vue par les voteurs des unités réceptrices et conduira à l'élimination de l'unité émettrice.

- le BGU sélectionne un ou plusieurs bus de façon erronée : les communications au sein de la machine sont perturbées et il est très difficile d'en connaître l'origine, cette panne est donc dangereuse. C'est pourquoi il a été décidé de doubler les BGU, l'émission d'une unité sur un bus ne se faisant que si ses deux BGU sont d'accord. Une panne non dangereuse d'un BGU est masquée. En revanche, si les deux BGU d'une unité sont affectés d'une panne dangereuse, cela peut conduire à la perte de un ou plusieurs bus (figure 13).

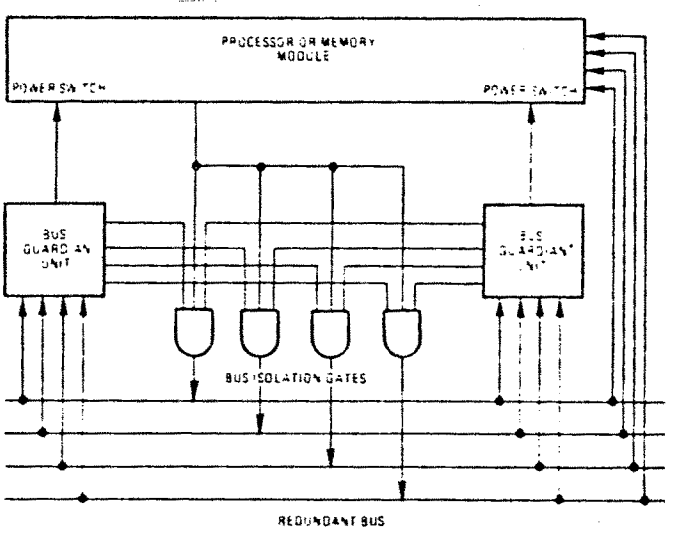


Figure 13 . Connection des BGU [HOP78]

Cela nous conduit à classer les couples de BGU comme autant de sous-ensembles suivant une stratégie de tolérance aux pannes particulière, le quadding, étant entendu que cette stratégie ne s'applique qu'à leur mode de panne dangereux.

Enfin, le choix d'une synchronisation stricte pose le problème d'une horloge centrale tolérant les pannes. La figure 14 donne le schéma adopté pour une telle horloge qui constitue un sous-ensemble de la machine.

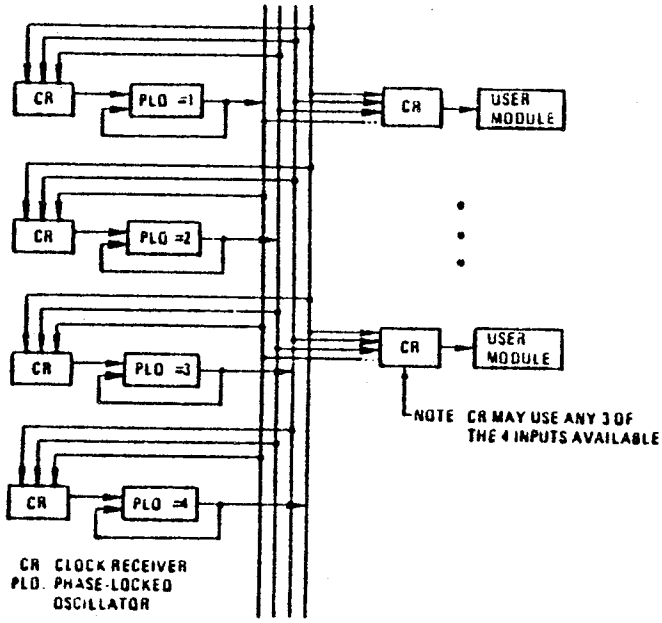


Figure 14 . Système d'horloge [HOP78]

Fonctionnement d'ensemble

Le fonctionnement d'ensemble de la machine ne peut plus être décrit en terme de ses sous-ensembles, mais d'ensembles plus vastes appelés calculateurs constitués de triades de processeurs, bus et mémoires.

La configuration de la machine est déterminée par le contenu des mémoires des voteurs (qui détermine les bus sur lesquels voter) et des BGU (qui détermine

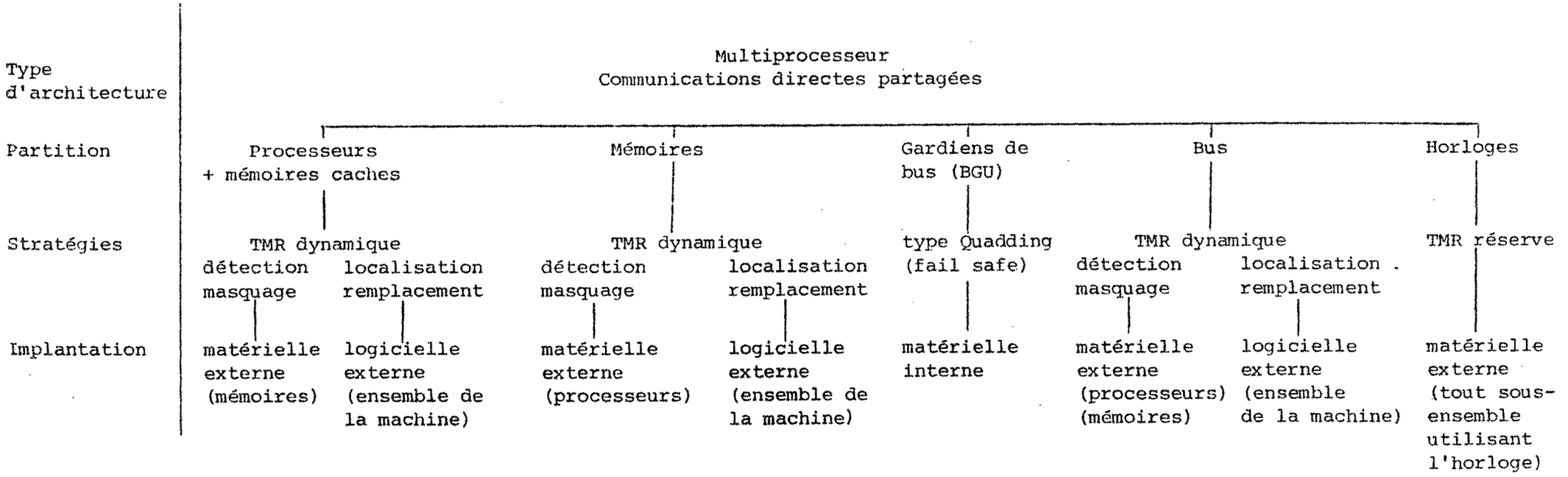


Figure 15 : Caractéristiques de FTMP

les bus sur lesquels émettre). Les calculateurs peuvent modifier le contenu de ces mémoires. De plus, les désaccords au cours des votes sont mémorisés par les voteurs et peuvent être lus par les calculateurs. L'apparition de tels désaccords conduit à l'exécution de programmes de test en vue de l'identification de l'élément en panne et la reconfiguration de la machine. Les caractéristiques de FTMP sont présentées dans le tableau de la figure 15 .

II.3.2.3. S.I.F.T. (Software Implemented Fault Tolerance) [WEN78][WEI80]

SIFT, en cours d'étude au Stanford Research Institute, vise une classe d'applications et de performances de sûreté de fonctionnement en tous points analogues à celles de FTMP. Deux versions, de même philosophie mais différentes au niveau de la réalisation des communications entre calculateurs, ont été successivement étudiées.

Du point de vue fonctionnel, SIFT apparaît comme un multicalculateur : chaque calculateur possède en propre son horloge, sa mémoire de travail et l'ensemble des programmes qu'il est susceptible d'exécuter.

Dans la première version ([WEN78]), l'interconnexion entre les calculateurs apparaît comme un système à communications indirectes partagées (figure 16) : lorsqu'un calculateur désire lire une donnée contenue dans la mémoire d'un autre calculateur, il commence par demander le système de communication. Lorsque celui-ci est libre et accepte de le servir (par arbitrage entre les demandes), le calculateur lui transmet l'adresse demandée. Le système de communication demande alors la mémoire concernée, et dès qu'il l'obtient, transmet la donnée au calculateur demandant.

Dans la deuxième version ([WEI80]), elle consiste en un système complet de communications directes-spécifiques (unidirectionnelles) : un calculateur émet, par un émetteur série qui lui est propre, vers tous les autres calculateurs ("broadcast") ; chaque unité reçoit, par un organe qui lui est propre,

toutes les émissions des autres calculateurs (figure 17).

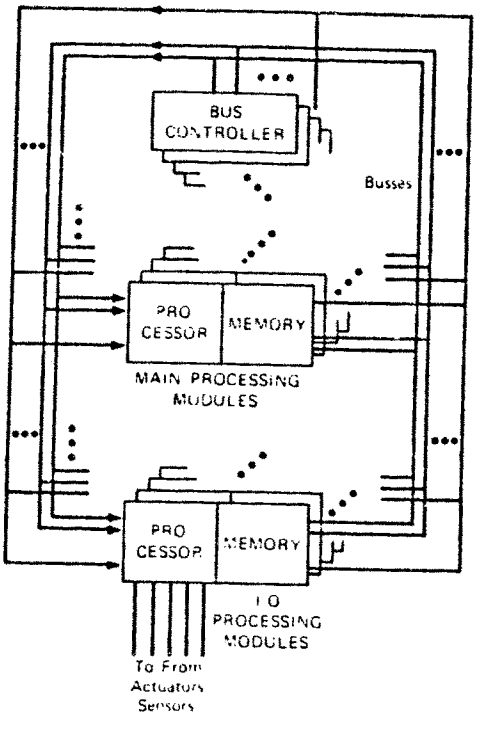


Figure 16 : Structure générale de la première version de SIFT [WEN78]

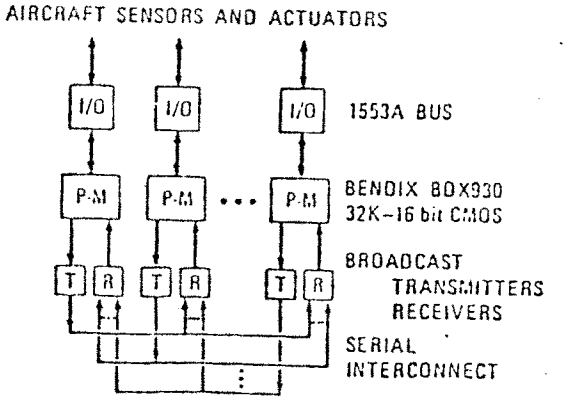


Figure 17 : Structure générale de la deuxième version de SIFT [WEI80]

Principes de tolérance aux pannes

Le principe adopté dans SIFT est celui du NMR dynamique assuré par voie logicielle : à un moment donné SIFT exécute plusieurs tâches, chaque tâche ayant un degré de redondance fixé en fonction de ses impératifs de sûreté de fonctionnement. Une tâche de degré de redondance n est exécutée par n calculateurs quasi simultanément : en effet, les calculateurs ne sont pas synchrones, ayant leur propre horloge : un calculateur peut participer à plusieurs n-uplets correspondant à plusieurs tâches.

Lorsqu'un calculateur doit acquérir une donnée d'une autre tâche, il acquiert les données de tous les calculateurs exécutant cette tâche et vote sur cette donnée. Lorsqu'un désaccord apparaît sur ce vote, dans la deuxième version, tous les calculateurs acquérant la donnée sont capables de diagnostiquer quelle est l'unité émettrice en panne. Dans la première version, un problème de diagnostic se pose, par rapport au système de communication : il s'agit de déterminer si l'erreur provient d'un calculateur ou d'un élément du système de communication ; ce système étant redondant, les calculateurs recevant la donnée pourront effectuer le diagnostic en recommençant l'acquisition de cette donnée en utilisant d'autres systèmes de communications. Un vote majoritaire sur les diverses versions du résultat ainsi obtenu permet de mettre ou non en cause le système de communication. Parmi les tâches de SIFT, une tâche particulière appelée exécutif est exécutée avec un degré de redondance élevé. Elle consiste à lire les diagnostics élaborés par les calculateurs, à établir un diagnostic global et à modifier la table d'allocation des tâches aux calculateurs. Chaque calculateur va périodiquement consulter cette table chez les calculateurs en charge de l'exécutif. Il détermine ainsi par vote majoritaire quelle tâche il doit exécuter, quels sont ses partenaires, qui exécute les tâches couplées à la sienne et qui sont les calculateurs chargés de l'exécutif.

Les caractéristiques des deux versions de SIFT sont présentées dans les figures 18 et 19.

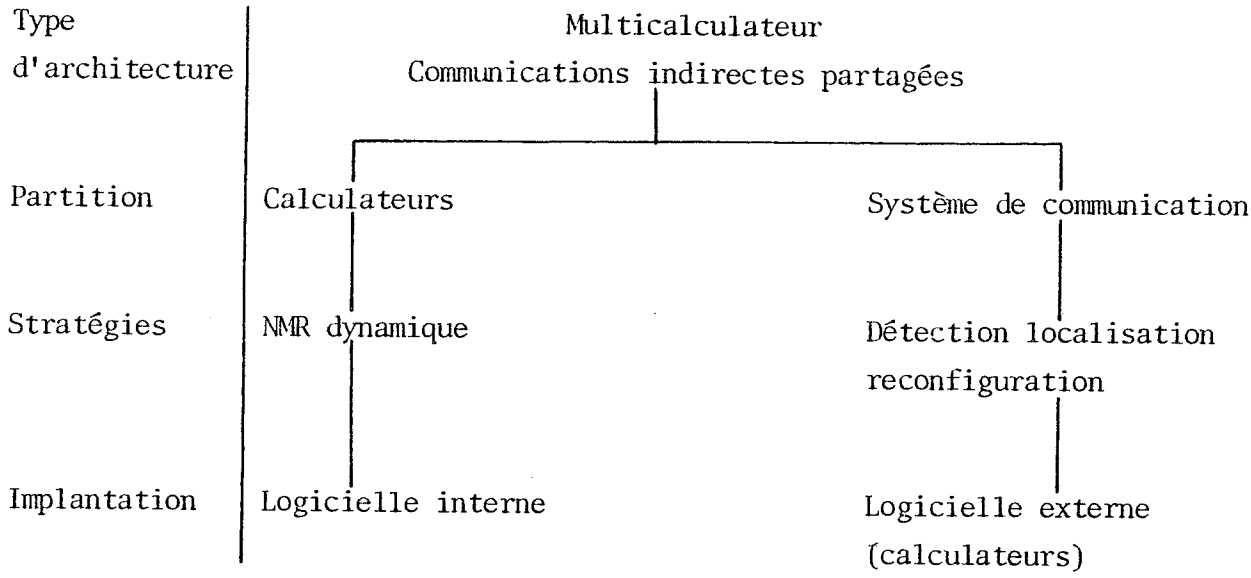


Figure 18 . Caractéristiques de SIFT (première version)

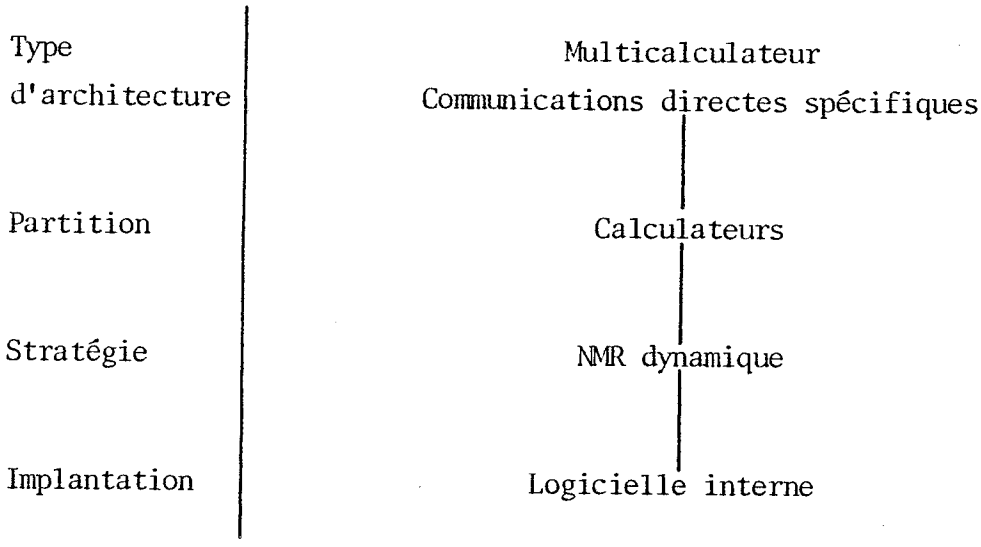


Figure 19 . Caractéristiques de SIFT' (deuxième version)

En conclusion, SIFT apparaît comme un aboutissement en ce qui concerne la tolérance aux pannes assurée par logiciel. En ce qui concerne l'architecture matérielle, l'aspect tolérance aux pannes n'apparaît que de deux façons :

- la redondance des éléments de chaque sous-ensemble
- l'isolation complète du point de vue des pannes entre chaque élément aussi bien dans le même sous-ensemble qu'entre sous-ensembles distincts.

II.3.3. Conclusion

Mis à l'épreuve de la description de machines réelles, le système de description proposé ici nous apparaît comme relativement satisfaisant. Il nous semble qu'il permet d'aller très vite à l'essentiel en posant des questions importantes : quels mécanismes sont mis en oeuvre, à quel matériel s'appliquent-t-ils, où et comment sont-ils réalisés. Ce système permet de plus une évaluation grossière des performances de sûreté de fonctionnement. Par exemple, un système étant partitionné à un niveau donné en n sous-ensembles S_i, chaque sous-ensemble ayant une fiabilité R_i, une évaluation grossière est obtenue par l'expression $\prod_i R_i$ (la fiabilité d'un sous-ensemble peut elle-même être obtenue de façon similaire dans le cas où le système est partitionné sur plusieurs niveaux).

Ces deux caractéristiques (poser les questions importantes, faciliter l'évaluation), semblent constituer une bonne base pour guider le concepteur de systèmes tolérant les pannes.

Nous nous proposons d'étudier dans le paragraphe suivant comment ce système de description peut être utilisé dans le cadre d'une démarche de conception.

II.4 - Un outil pour une conception structurée

II.4.1. Introduction

La nécessité de trouver une démarche pour la conception de systèmes à haute sûreté de fonctionnement a déjà été largement évoquée dans le début de ce chapitre. Plus précisément, il nous semble que les principaux écueils auxquels se heurte le concepteur, au moins pendant les premières phases de la conception, sont les suivants :

- il tente souvent de prendre en compte en même temps les spécifications de sûreté de fonctionnement et les spécifications de l'application elle-même : la variété des critères de choix rend cette phase souvent confuse.
- devant la grande variété de mécanismes de tolérance aux pannes existants, il semble difficile de ne choisir que le strict nécessaire permettant d'atteindre les résultats requis : on cherche souvent à implanter un grand nombre de mécanismes variés destinés à "traquer" la panne, sans être certain que l'augmentation de complexité qui en résulte ne va pas à l'encontre du résultat recherché.
- le processus de conception et celui de validation sont quelquefois trop disjoints, la validation n'étant effectuée qu'a posteriori et ne permettant pas de guider le concepteur.

La démarche que nous proposons dans ce paragraphe est destinée à lever ces difficultés (dont la description rapide est sans doute caricaturale). Pour ce faire, elle s'appuie largement sur la méthode de conception par raffinements successifs proposée dans [PUL79] et sur les rubriques du système de description proposé, transformées alors en étapes de la conception.

II.4.2. Présentation de la démarche

La démarche consiste en la succession des quatre "étapes-rubriques" déjà évoquées :

a) choix d'une architecture non redondante, capable de réaliser l'application : cette étape représente en elle-même un processus complet de conception, pour lequel des démarches structurées devront être appliquées [MOA81], [PIL82].

Cette étape conduisant à la définition d'une architecture non redondante est cependant aussi guidée par les objectifs de sûreté de fonctionnement : certains dispositifs sont réputés comme posant des problèmes lorsqu'ils doivent être fiabilisés (certains dispositifs de communication indirecte pouvant bloquer tous les éléments communiquants...) ou necessitant une implantation particulière pour permettre l'adjonction de mécanismes de tolérance aux pannes (découpage en tranches de la mémoire pour l'implantation de codes) : les choix du concepteur seront donc guidés par la recherche d'une implantation facile de la tolérance aux pannes, sans que cette implantation soit effectuée à ce stade : il s'agit d'éviter l'implantation de mécanismes non justifiée par les objectifs de sûreté de fonctionnement. La prise en compte des objectifs de sûreté de fonctionnement dès cette phase est d'autant plus justifiée que le système à concevoir est important (une remise en cause ultérieure de l'architecture non redondante serait alors trop coûteuse) et sans doute moins primordiale dans le cas de petits systèmes spécifiques à une application, tels que le calculateur CARL, objet de l'étude du prochain chapitre.

b) choix d'une partition du système

cette étape est, nous le verrons par la suite, capitale lors de chaque raffinement : il faut situer les organes spécifiques de tolérance aux pannes, définir des sous-ensembles plus gros ou plus fins selon les résultats d'évaluation...

c) choix des stratégies de tolérance aux pannes

leur nature est fixée par les objectifs de sûreté de fonctionnement et le degré de redondance par les résultats d'évaluation.

d) choix d'une implantation des stratégies

les critères de choix entre les solutions matérielles et logicielles ont déjà été évoqués. Le choix dépend largement des possibilités offertes par les étapes précédentes et du savoir-faire du concepteur.

Cette étape sera souvent suivie d'un retour à la deuxième étape (partition du système) pour placer les éléments introduits pour l'implantation des stratégies dans des sous-ensembles du système.

Les quatre étapes sont systématiquement suivies d'une évaluation quantitative de la grandeur significative de sûreté de fonctionnement. Cette évaluation pourra être grossière dans les premières phases, mais devra s'affiner lorsqu'on se rapprochera du résultat recherché. L'évaluation peut conduire à trois résultats : le résultat est conforme à la spécification (la conception est terminée), le résultat est trop bon, c'est-à-dire meilleur que le chiffre spécifié (on peut alors diminuer le coût et la complexité du système), ou enfin, le résultat est trop mauvais, c'est-à-dire moins bon que le chiffre spécifié (on doit alors remonter à l'une des étapes pour partitionner plus finement le système, renforcer les stratégies ou modifier leur implantation). La démarche de conception est résumée dans la figure 20 .

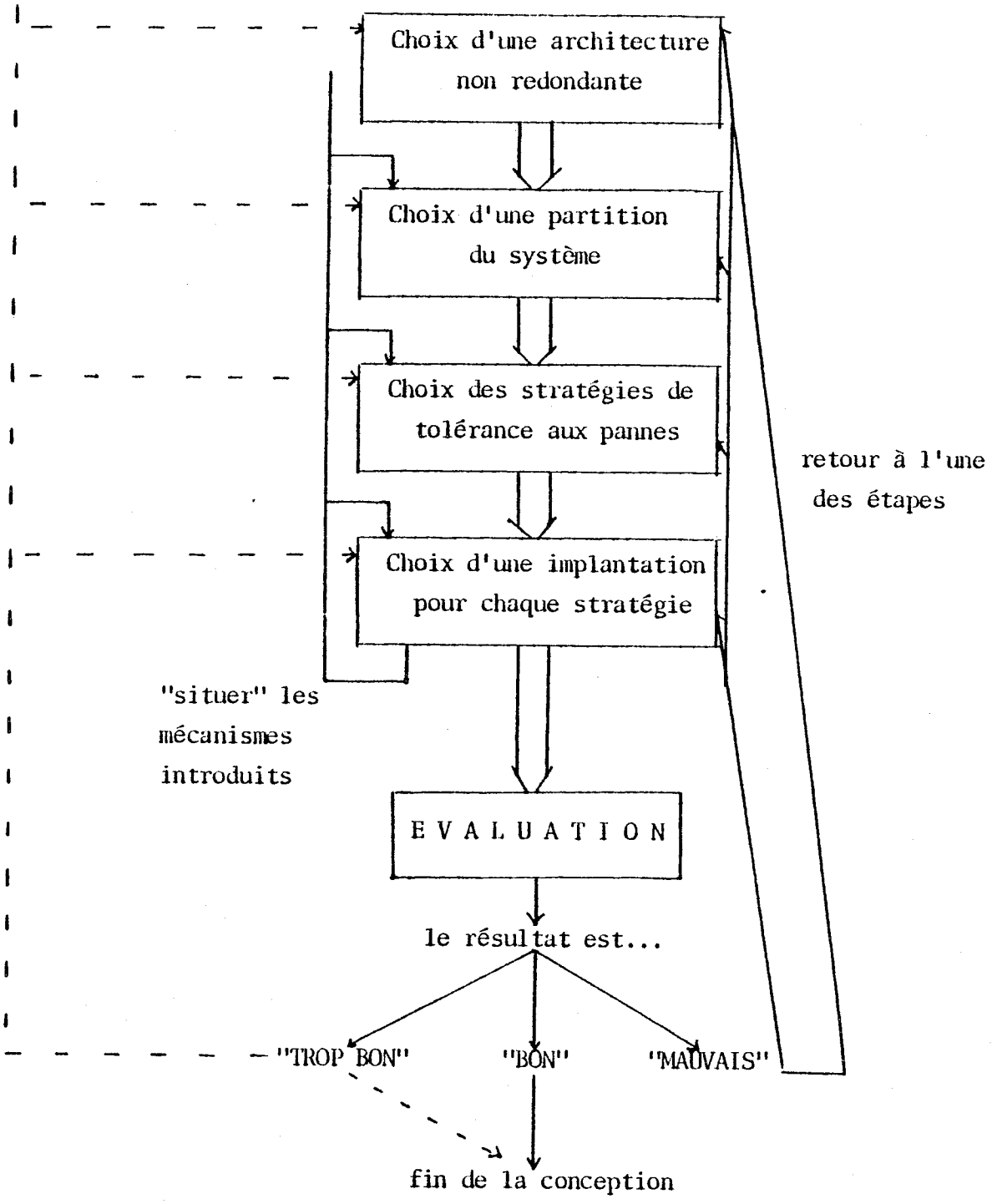


Figure 20 : Schéma de la démarche de conception proposée

II.4.3. Processus de conception

La recherche de la minimalité guidera le passage entre les diverses étapes décrites : en effet, il s'agit de minimiser le matériel mis en jeu (minimiser le taux de pannes intrinsèque) et de simplifier les mécanismes utilisés (faciliter la validation). Nous décrivons ci-dessous quelques "règles" de passage dans le processus de conception.

a) Initialisation - Premier passage

Le concepteur choisit une architecture minimale non redondante, qu'il laisse former un seul ensemble auquel il applique la stratégie "simplex" : ceci conduit à évaluer l'architecture minimale non redondante. Si cette évaluation donne déjà un résultat satisfaisant, cette architecture sera retenue ; dans le cas contraire, l'évaluation donne de précieux enseignements pour la poursuite de la conception : au vu de l'ordre de grandeur de l'amélioration à apporter, le concepteur aura une idée de la classe de stratégies à appliquer ; l'évaluation pourra faire apparaître les "points faibles" du système qu'il conviendra sans doute de placer dans des sous-ensembles à "stratégies très efficaces".

b) Autres passages

Ces passages consisteront en un enrichissement progressif de la structure : il s'agira de l'adoption ou de la remise en cause d'un choix relevant d'une des quatre étapes et des modifications qui en découlent pour les autres rubriques. Outre la recherche d'une solution minimale (enrichissement minimal), les choix seront guidés par les remarques suivantes :

. les choix de partition et d'implantation peuvent influer sur le choix de l'architecture non redondante : on pourra préférer à l'architecture minimale une architecture de complexité voisine, présentant des caractéristiques intéressantes pour la tolérance aux pannes : citons notamment la

capacité d'isolation et la facilité de diagnostic d'un système de communication, ou l'isolation et la protection des données dans le cas d'un vote (ou d'une comparaison) logicielle.

. Plus les évaluations deviennent fines, mieux elles renseignent sur l'utilité éventuelle de stratégies annexes pour améliorer les résultats de sûreté de fonctionnement : on trouvera dans les chapitres suivants des exemples d'évaluation de stratégies de redondance massives faisant apparaître l'influence de la latence des pannes : cette influence peut conduire le concepteur à ajouter à la stratégie de redondance massive, des stratégies à redondance sélective pour chaque unité (codes, tests en ligne, tests périodiques), afin de diminuer cette latence.

. Le résultat d'évaluation "trop bon" mérite un traitement particulier : dans la mesure où la recherche d'une solution minimale a guidé la conception, les économies pouvant être réalisées sans faire trop chuter le résultat, sont limitées : elles ne concernent pas le degré de redondance (choisi minimum), mais plutôt des détails d'implantation (fréquence de comparaison diminuée...) ou de partition (supprimer certaines liaisons dans la machine, c'est-à-dire modifier les sous-ensembles) ; en fait, de telles études ne se justifient vraiment que dans le cas où le système conçu sera produit en un grand nombre d'exemplaires, c'est-à-dire lorsque les économies réalisées contrebalancent largement les coûts d'étude supplémentaires ; enfin, l'obtention de résultats "trop bons" est souvent exploitée comme argument face à des clients potentiels ou aux autorités de certification.

II.5 - CONCLUSION

Le système de description proposé et appliqué dans ce chapitre, ainsi que la démarche de conception qui en déroule, semblent respecter le besoin de prise en compte permanente du problème de la validation. Cette prise en compte se manifeste au niveau de plusieurs "points-forts" exposés :

- le système doit pouvoir être décrit simplement (pratiquement entièrement en quatre rubriques),
- le système doit être clairement partitionné : pas d'élément particulier, rajouté a posteriori et nécessitant une analyse particulière ; possibilité de décomposer le problème de validation.
- la validation est intégrée dans la conception et sert de guide pour celle-ci.

Un système conçu en suivant scrupuleusement ces principes nous semble constituer un champ d'application des méthodes d'évaluation rigoureuses présentées dans la deuxième partie de ce mémoire.

CHAPITRE III

APPLICATION A LA CONCEPTION D'UN BI-CALCULATEUR A HAUTE FIABILITE

III.1 - INTRODUCTION

La base de la réalisation pratique présentée dans ce chapitre et étudiée dans la suite de cet ouvrage, est constituée par les travaux sur la conception d'une centrale anémométrique, présentés dans [CER78], [PUL79] et [CAS81a].
Le but de cette réalisation est double :

. d'une part, il s'agit d'expérimenter les idées et conclusions essentielles de ces travaux,

. d'autre part, il s'agit, en appliquant le système de description proposé dans le chapitre précédent, d'analyser les "points faibles" de l'architecture retenue initialement et de proposer d'autres solutions.

Le premier aspect est essentiellement abordé dans le chapitre VIII, qui montre la faisabilité et les "bonnes propriétés" des mécanismes logiciels qui font l'originalité du système.
Le présent chapitre est consacré au deuxième aspect, et est articulé autour de trois paragraphes principaux :

. le premier résume la structure du calculateur "centrale anémométrique" initialement proposé,

. le second traite de l'application du système de description pour l'analyse du calculateur et décrit les modifications qui résultent de cette analyse,

. le troisième précise l'architecture détaillée de la maquette expérimentale réalisée.

III.2 - DESCRIPTION DE LA CENTRALE ANEMOTRIQUE

III.2.1. Spécifications de sûreté de fonctionnement

Le système considéré doit délivrer un certain nombre de signaux dits très dangereux avec une infiabilité de 10^{-6} pour un temps de mission de 10 heures. Remarquons que la caractéristique demandée est la fiabilité :

l'absence de sortie (qui pourrait être obtenue dans un système avec arrêt et alarme) est aussi dangereuse que l'existence d'une sortie erronée.

III.2.2. Caractéristiques fonctionnelles et architecture non-redondante

L'application elle-même ("centrale anémométrique") ne sera pas décrite ici en détails. Nous nous contenterons de mentionner les caractéristiques de celle-ci, définissant ainsi la classe d'applications qui pourraient être implantées sur un tel système.

L'analyse de l'application a conduit à une décomposition fonctionnelle en deux tâches, telles que :

. chaque tâche peut être implantée sur un microcalculateur et ses circuits annexes,

. les communications entre tâches sont à faible débit ; les tâches étant d'autre part implantées sur des calculateurs voisins géographiquement, ces communications pourront être réalisées sous la forme "directe spécifique" par l'interconnexion de ports parallèles d'entrées-sorties,

. chaque tâche est connectée à un capteur propre d'entrée, considéré comme faisant partie du système et devant, à ce titre, être pris en compte dans l'analyse de fiabilité.

Cette décomposition conduit donc à définir l'architecture non-redondante présentée dans la figure 1 .

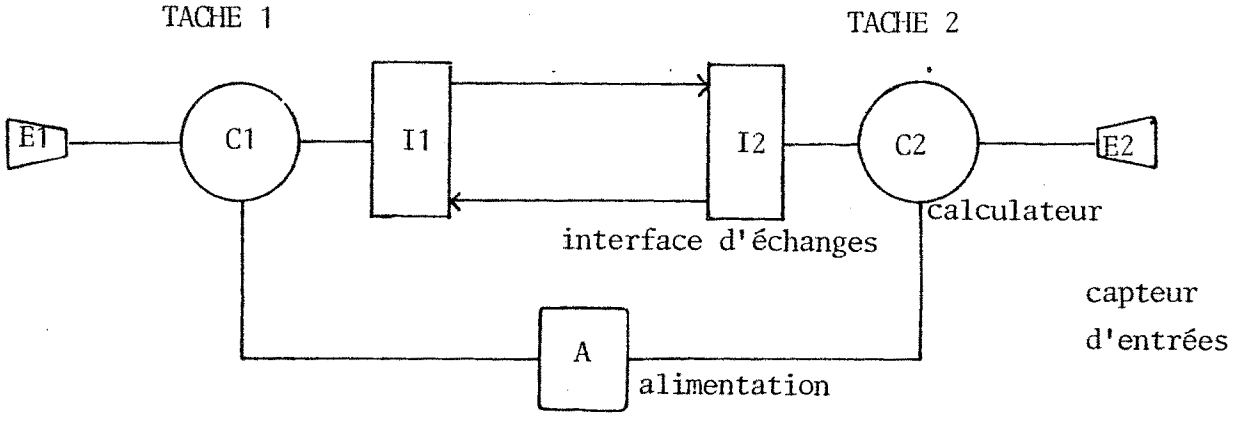


Figure 1 : Architecture non-redondante du système

Enfin, l'application présente deux caractéristiques importantes vis-à-vis de la tolérance aux pannes :

. chaque tâche est cyclique, les résultats d'un cycle étant indépendants du cycle précédent (cette propriété constitue une hypothèse forte permettant d'éliminer les problèmes de sauvegarde de contexte pour la reprise après panne).

. Le système externe recevant les sorties du calculateur est dit "intelligent" au sens où il sait reconnaître la bonne sortie à partir des sorties d'un ensemble de calculateurs, intervenant dans une même stratégie de tolérance aux pannes, et parmi lesquels un certain nombre peuvent être en panne.

III.2.3. Partition du système

Notons tout d'abord que l'alimentation n'a pas à être prise en compte dans l'étude : souvent commune avec d'autres équipements que le calculateur, elle est supposée rendue fiable par ailleurs.

L'étude concerne donc la structure composée des calculateurs (et leurs circuits annexes : extensions mémoires, ports d'entrées-sorties...) et des capteurs.

Il a été montré [PUL79] qu'une triplication de tout le système (TMR ou TMR/Simplex) ne permet pas d'atteindre les objectifs de fiabilité fixés : il convient donc de partitionner ce système et d'appliquer une stratégie propre à chaque sous-ensemble défini.

Un calculateur et ses circuits annexes forment un ensemble fortement connecté : un partitionnement au sein de cet ensemble poserait le problème de l'isolation et de la localisation des pannes dans les sous-ensembles définis, problème dont la résolution s'avèrerait lourde : ceci conduit à conserver les calculateurs entiers comme sous-ensembles cohérents et à ne partitionner le système

qu'en deux sous-ensembles, de taux de pannes au demeurant sensiblement équivalents : les capteurs et les calculateurs.

III.2.4. Stratégie "calculateurs"

La propriété de l'application précisant que les tâches sont cycliques, permet d'envisager une stratégie de type détection-localisation-remplacement : en effet, la limitation habituelle concernant la difficulté des problèmes de reprise dans la gestion de ces stratégies, est levée par cette propriété . Nous décrivons ci-dessous les mécanismes de détection, de localisation et de remplacement choisis.

a) Détection

Elle est assurée par redondance massive (duplication et comparaison).

La comparaison des résultats des deux unités d'un duplex est assurée par voie logicielle. Ce choix induit la création d'une liaison entre les deux unités pour permettre l'échange des résultats en vue de la comparaison. De plus, pour assurer la comparaison des résultats d'opération d'émission de messages vers l'autre tâche, et de réception de messages provenant de l'autre tâche, l'unité supplémentaire doit pouvoir "espionner" les émissions et réceptions de l'autre unité, ce qui introduit de nouvelles liaisons. La figure 2 présente l'architecture redondante du sous-ensemble calculateur obtenue.

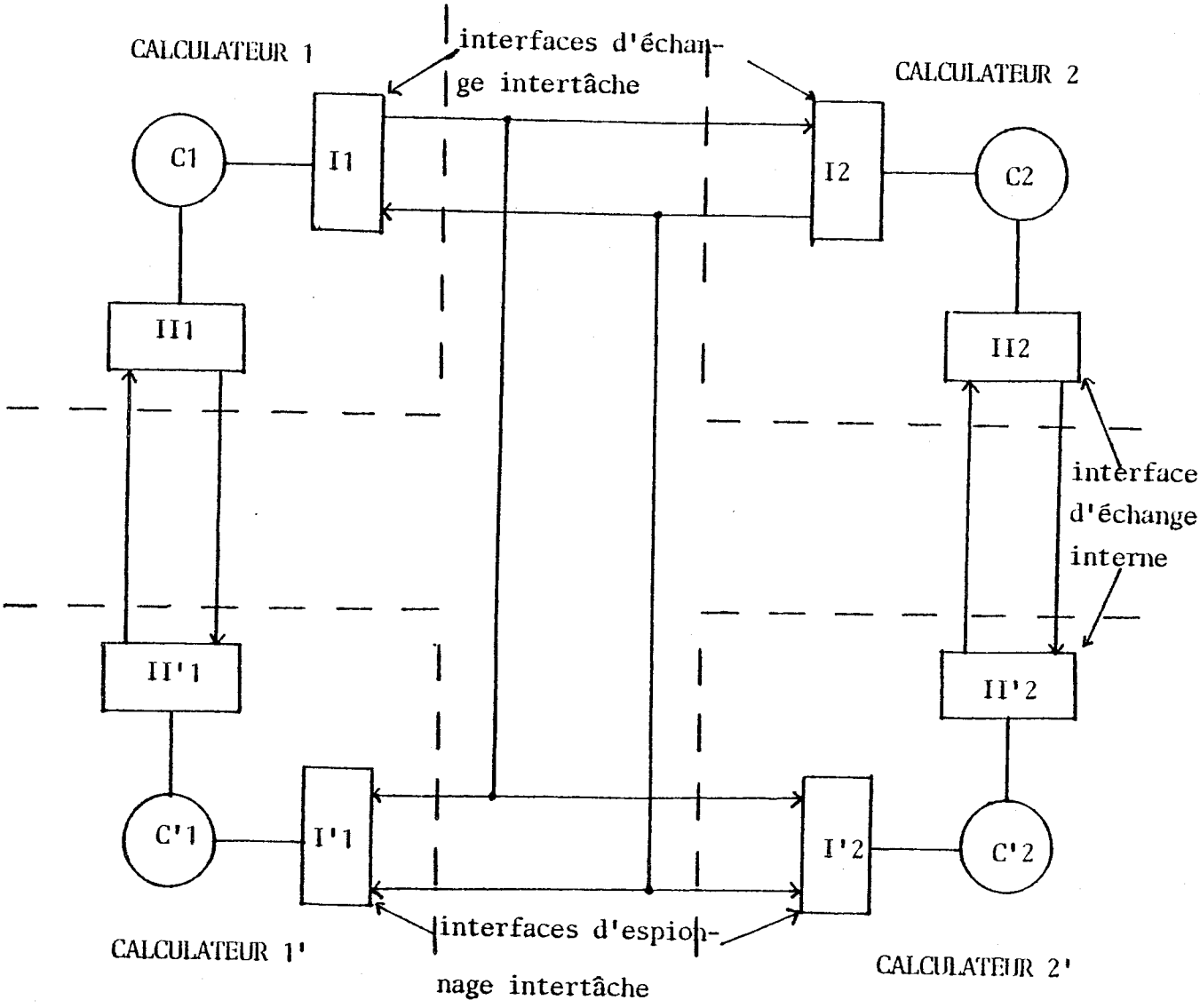


Figure 2 : Architecture redondante du sous-ensemble calculateur (réalisant la détection).

b) Localisation

A l'apparition d'un désaccord, la localisation de la panne dans l'une des deux unités nécessiterait un test-diagnostic très efficace, lourd à mettre en oeuvre.

Ceci conduit donc à éliminer les deux unités suspectes à l'apparition d'une panne.

c) Reconfiguration

L'originalité de la stratégie apparaît avec la définition de la politique de reconfiguration : après l'élimination d'un duplex (exemple (1,1')), le système se réduit au duplex restant (ex. (2,2')) ; nous pouvons alors constater que l'architecture restante présente la même structure que l'architecture non redondante définie en III.2.2. : un bi-calculateur à communications par ports parallèles d'entrées-sorties. L'idée est alors de faire reprendre, par le calculateur "de surveillance" du duplex restant (ex. 2'), la tâche du duplex défaillant, les interfaces d'échanges internes (ex. : II2, II'2) jouant désormais le rôle des interfaces d'échanges intertâches : les deux tâches sont exécutées en mode simplex.

Pour implanter cette reconfiguration, il faut d'une part que les calculateurs "de surveillance" soient capables d'exécuter les deux tâches (ressources suffisantes), d'autre part que le duplex valide soit averti d'un désaccord au sein de l'autre duplex. Le dispositif matériel présenté dans la figure 3 est proposé pour assurer cette fonction.

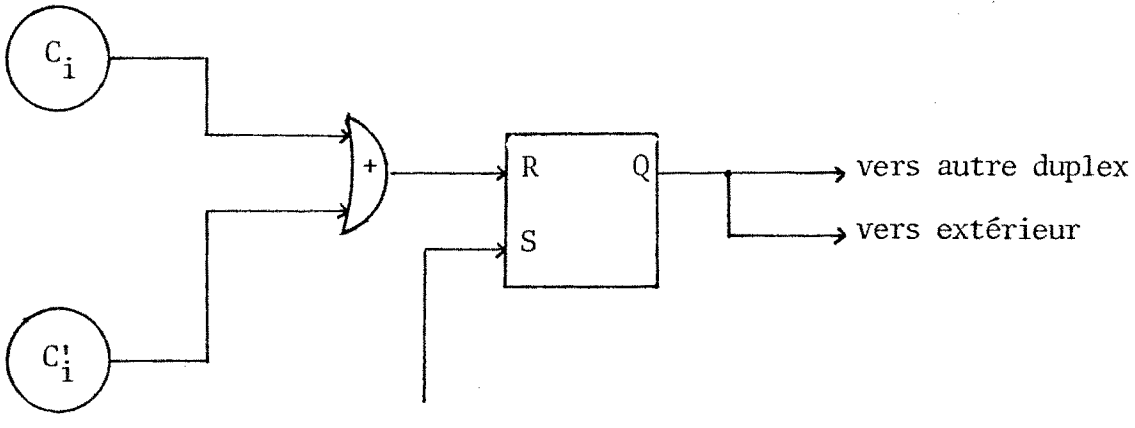


Figure 3 : Mécanisme d'ordre de reconfiguration

Le système se voit donc adjoindre deux mécanismes d'ordre de reconfiguration (un par tâche) ; remarquons que le signal d'ordre de reconfiguration peut être utilisé pour tuer le duplex en panne, c'est-à-dire l'éliminer réellement, par exemple en le rebouclant sur les entrées d'inhibition d'horloge de chaque calculateur.

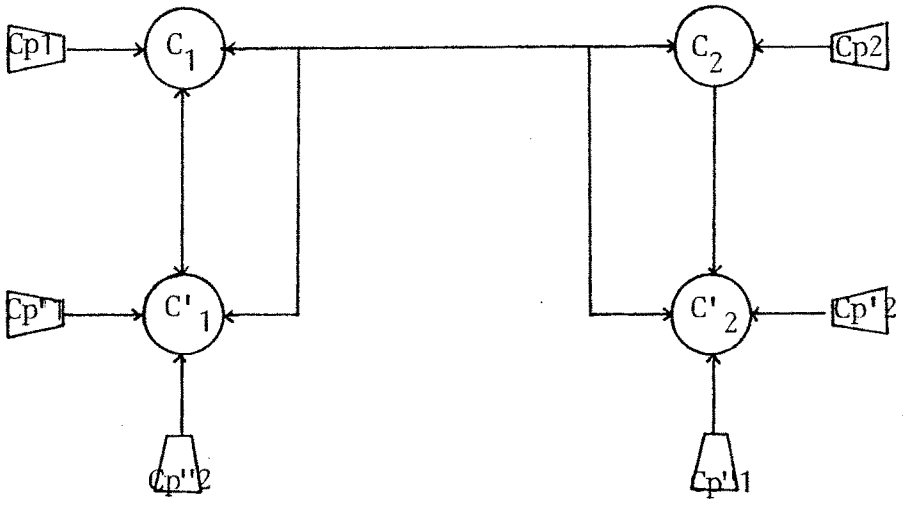
d) Conclusion

L'efficacité de cette stratégie, évaluée par ailleurs à des fins de comparaison avec d'autres stratégies dans [PUL79] et [CAS81], et de manière plus absolue dans la suite de ce travail (Chapitre VII), peut être justifiée ainsi : l'usage d'une réserve active permet de survivre à toutes les premières pannes avec un nombre minimum d'unités, c'est-à-dire avec un taux de pannes intrinsèque minimum, alors que des stratégies utilisant six unités (bi-TMR, bi-duplex avec réserve oisive), permettent, dans certains cas, de survivre à deux pannes (une panne affectant chaque tâche), mais aussi dans d'autres, de ne survivre qu'à une seule (deux pannes pour la même tâche) pour un taux de pannes intrinsèque plus important.

III.2.5. Stratégie "capteurs"

Dans l'application anémométrique, chaque capteur comporte une mémoire d'étalonnage propre, et est donc vu par un calculateur comme une zone mémoire. Il convient donc d'éviter l'utilisation simultanée d'un capteur par plusieurs calculateurs, utilisation qui obligerait à prévoir un mécanisme d'arbitrage. D'autre part, une première analyse de fiabilité a conclu à la nécessité d'une redondance d'ordre trois.

Cela induit une première solution A, simple : intégrer calculateurs et capteurs, selon le schéma :



Aucun capteur n'est partagé, et les capteurs sont tripliques.
L'analyse de fiabilité montre cependant que cette solution n'est pas suffisante.

Une autre solution classique, consiste à appliquer une stratégie tripliquée, indépendante des calculateurs. Mais cela implique que les trois capteurs associés à une tâche puissent être partagés par les deux calculateurs du duplex exécutant cette tâche.

L'analyse montrant que cette solution est trop bonne, PULOU propose une solution intermédiaire ne nécessitant pas le partage des capteurs (figure 4).

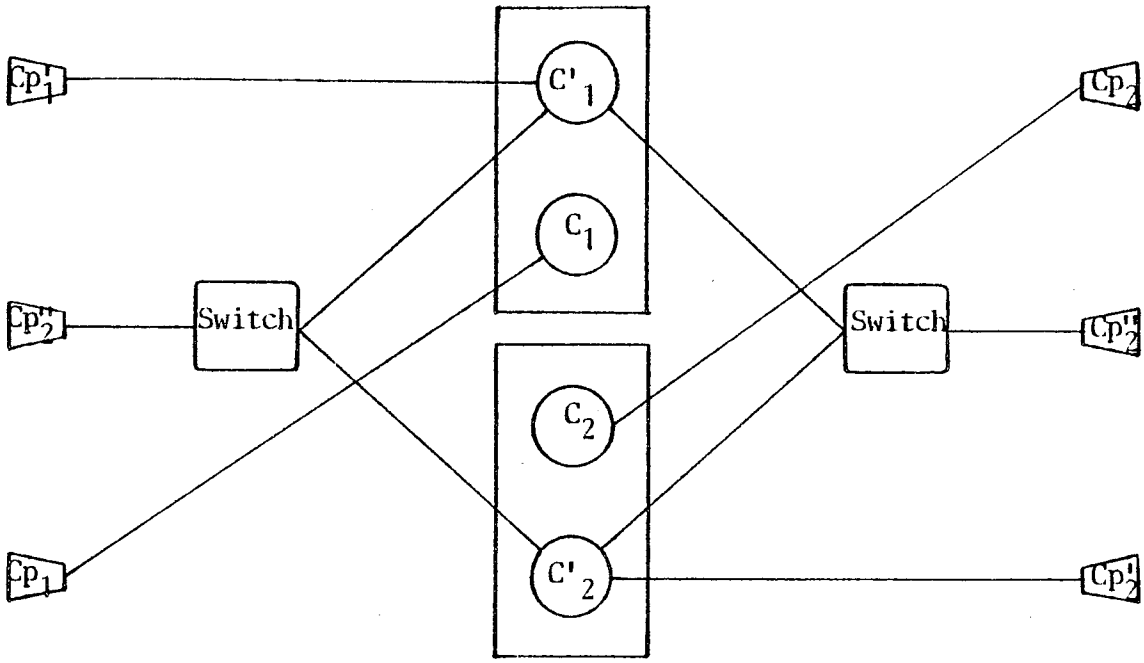


Figure 4 : Stratégie "optimisée" sur les capteurs

En fonctionnement duplex, les capteurs Cp''_1 et Cp''_2 sont aiguillés sur les unités C'_1 et C'_2 respectivement, et sont en réserve.

Avant toute panne, chaque calculateur acquiert les entrées sur son capteur propre ; un désaccord peut donc provenir soit d'une panne de capteur, soit d'une panne de calculateur ; la stratégie consiste alors à suspecter a priori le capteur propre de l'unité de surveillance (1' ou 2') et à le remplacer par le capteur partagé (via le "switch"). En cas de nouveau désaccord ou de désaccord persistant, la paire de calculateurs est supprimée et l'unité de réserve acquiert ses entrées sur le capteur partagé, correctement aiguillé par l'organe de "switch". La panne de l'organe de "switch" provoque l'échec de la stratégie capteurs.

III.2.6. Structure générale

La structure générale obtenue est présentée dans la figure 5 .

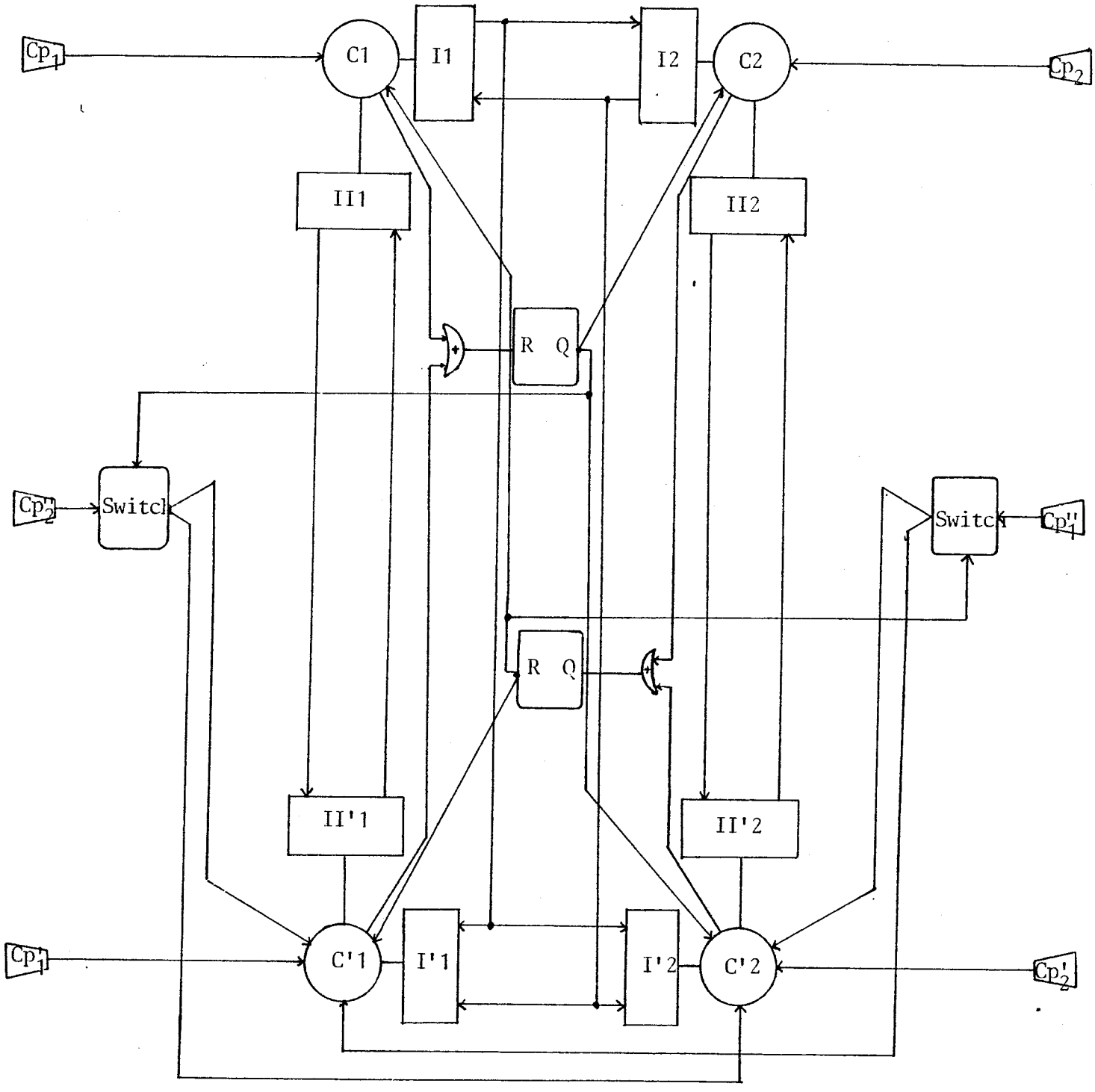


Figure 5 : Structure générale proposée

III.3. APPLICATION DU SYSTEME DE DESCRIPTION

III.3.1. Analyse

La structure non redondante a été clairement décrite dans le paragraphe précédent et peut être caractérisée ainsi :

<bi-calculateur à communications directes spécifiques par ports d'entrées-sorties parallèles>

La description de la partition du système fait ressortir les points suivants :

a) la stratégie "optimisée" sur les capteurs a pour effet de lier les effets des stratégies sur les capteurs et les calculateurs :

. la suppression d'une paire de calculateurs entraîne la perte de deux capteurs.

. les pannes de capteurs sont détectées à travers le mauvais fonctionnement qu'elles induisent dans les calculateurs et certaines d'entre elles provoquent la suppression de la paire de calculateurs.

b) Les mécanismes gérant la reconfiguration forment un sous-ensemble propre.

Il convient donc de décrire la partition du système ainsi :

< système = calculateur / mécanismes de reconfiguration >

La description des stratégies appliquées à ces sous-ensembles fait apparaître les faiblesses du système :

. la description de la stratégie globale appliquée au sous-ensemble "calculateurs" comprenant les capteurs, s'avère compliquée : la conséquence de cette complexité est que l'application d'un modèle fin de fiabilité, du type de ceux proposés dans la suite de l'ouvrage, est rendue très problématique par le grand nombre de cas de pannes à décrire ; il sera alors difficile de baser la validation du système sur des résultats quantitatifs "dignes de confiance".

. Les mécanismes de reconfiguration suivent une stratégie simplex et constituent un "point dur" du système ; ces mécanismes n'effectuant pas de tâches de calcul, l'effet de leurs pannes peut être caractérisé suivant deux modes :

- ordre abusif : en l'absence de désaccord, le mécanisme donne abusivement l'ordre de reconfiguration et fait "vieillir prématurément" le système.
- pas d'ordre : en présence d'un désaccord, le mécanisme ne donne pas l'ordre de reconfiguration : cette panne, sensible après une panne de calculateur, provoque l'échec de la mission.

Nous remarquons que les pannes de ces mécanismes doivent être prises en compte dans une analyse globale du système, ce qui pose également le problème de la complexité de l'analyse, d'autant que les calculateurs comportent aussi les modes de pannes proches de ceux décrits : obéir abusivement à un ordre de reconfiguration et ne pas obéir à un ordre de reconfiguration, ou envoyer abusivement un ordre de reconfiguration et ne pas envoyer d'ordre de reconfiguration : les mécanismes de reconfiguration apparaissent comme des intermédiaires "fragiles" (parce qu'en simplex) et "superflus" (ils constituent un organe centralisé de communication des ordres de reconfiguration, alors que la philosophie de "communication directe-spécifique" pourrait s'appliquer à ces messages particuliers).

III.3.2. Modifications apportées

Les modifications apportées sont destinées à remédier à ces faiblesses en clarifiant la partition du système. Il s'agit des modifications suivantes :

. Remise en cause de la stratégie "capteurs" et retour à une stratégie permettant le partitionnement < capteurs, calculateurs > :

l'économie (moins de lectures effectuées pour chaque entrée, moins de connexions) offerte par la stratégie optimisée est largement contrebalancée par

la quasi-impossibilité de validation et la complexité des mécanismes de gestion de la stratégie ; suivant la nature précise de l'application implantée (le type de capteurs utilisé), nous choisirons l'une des stratégies tripliquées (TMR, TMR-simplex, duplex réserve), ou bien d'intégrer les capteurs aux calculateurs.

. Intégration des organes de reconfiguration dans l'ensemble calculateur : chaque calculateur reçoit l'ordre de reconfiguration à partir d'un mécanisme qui lui est propre, de sorte qu'une panne d'un mécanisme n'affecte qu'un calculateur et est assimilable aux pannes "d'obéissance abusive" ou de "non obéissance à un ordre de reconfiguration" du calculateur. Ce mécanisme peut alors consister simplement en une porte "OU" connectée à l'entrée d'interruption du microcalculateur.

La nouvelle architecture obtenue est présentée dans la figure 6 .
Sa caractérisation suivant le système de description, est résumée par la figure 7 .

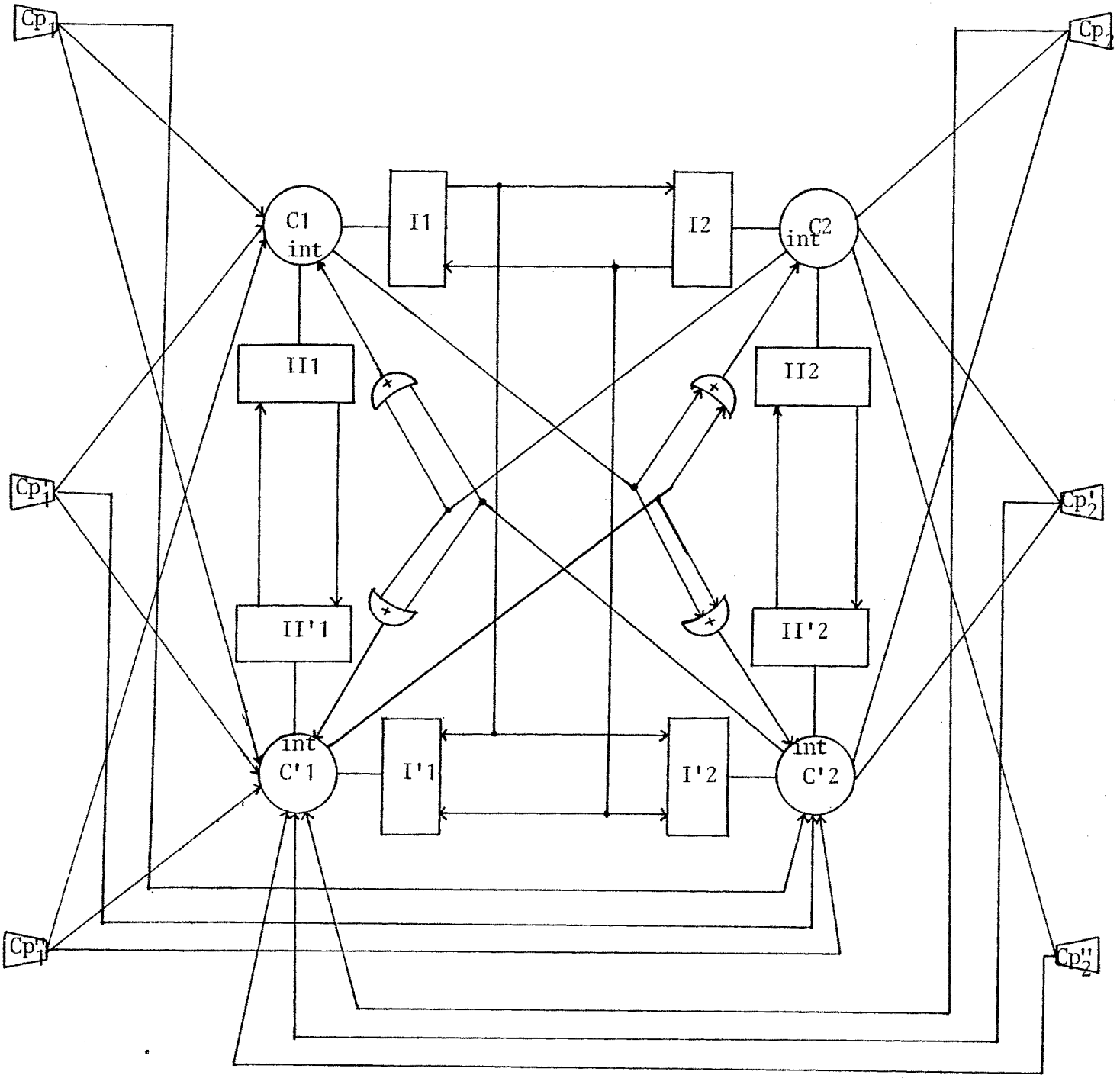


Figure 6 : Nouvelle architecture

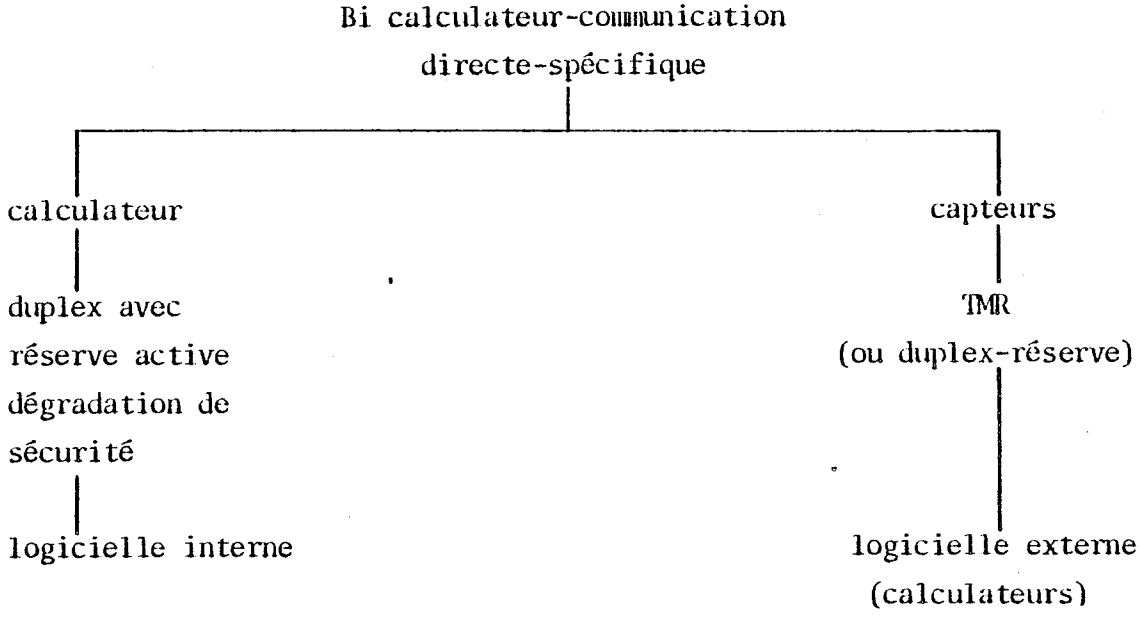


Figure 7 : Caractéristiques de la nouvelle architecture

III.4. CONFIGURATION EXPERIMENTALE : C. A. R. L., Calculateur A Reconfiguration Logicielle

Le système expérimental réalisé dans le cadre de ce travail, baptisé CARL (Calculateur A Reconfiguration Logicielle) est destiné à permettre l'expérimentation de l'implantation logicielle de mécanismes de tolérance aux pannes (voir chapitre VIII). Il ne supporte pas actuellement d'application spécifique et à ce titre ne comporte donc pas de capteurs.
Les entrées-sorties du système sont constituées d'une interface série parallèle par calculateur (connexion à une console) et de deux ports d'entrées-sorties parallèles de 4 x 4 bits (soit 8 x 4 bits au total) destinés à supporter les entrées et sorties spécifiques aux applications à implanter.
Remarquons que ce sont les calculateurs dits de surveillance (respectivement 1' et 2') qui nécessitent le plus d'interfaces : une pour la relecture des sorties des calculateurs dits actifs (respectivement 1 et 2), une pour la sortie de la tâche reprise en fonctionnement dégradé (resp. tâche 2 et tâche 1), une par capteur de la tâche exécutée en fonctionnement normal (resp. tâche 1 et tâche 2) et une par capteur de la tâche exécutée en fonctionnement dégradé. La configuration actuellement cablée suppose que le format d'entrée-sortie est de 4 bits et permet d'implanter des stratégies

TMR sur les entrées. En dehors de cette remarque, le système réalise "l'architecture modifiée" proposée dans le paragraphe précédent à l'aide des composants de la gamme INTEL 8748 [INT] :

le système est composé de quatre cartes calculateurs comprenant essentiellement :

. un microcalculateur monolithique 8748 (microprocesseur + 1K ROM + 64 octets de RAM)

. des extensions mémoires RAM (2114) et ROM (2716)

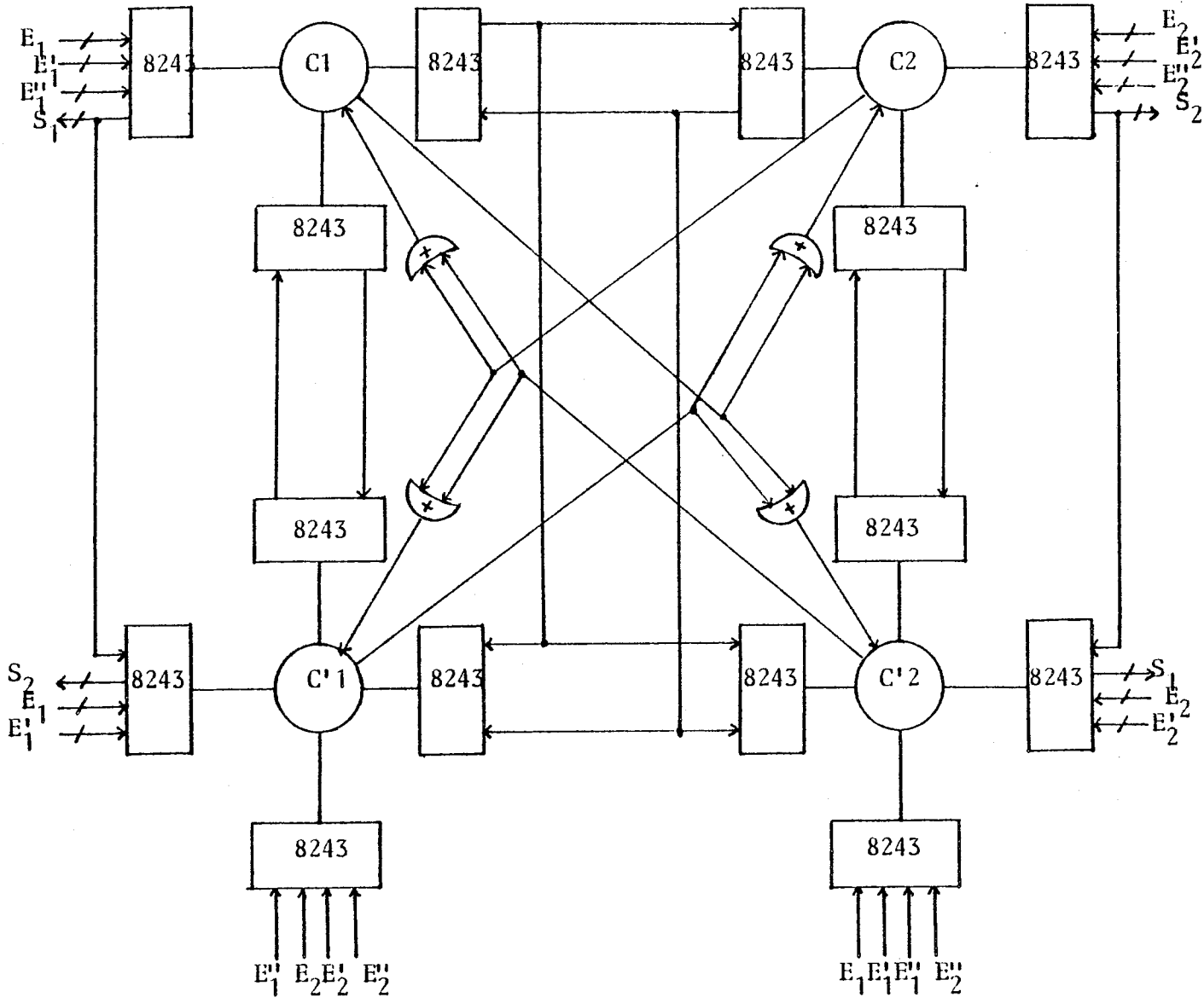
. des ports d'entrées-sorties 8243 (4 x 4 bits chaque) :

  1 pour les échanges inter-couples

  1 pour les échanges internes à un couple

  2 pour les entrées-sorties

Le schéma complet du système est présenté figure 8 . Le détail d'une carte calculateur est donné figure 9 .



Légende : C_i : ensemble comprenant : calculateur, extension mémoire, décodage d'adresse, port entrées-sorties séries

E_1, E''_1, E''_2 : entrées provenant des capteurs tripliqués de la tâche 1.

S_i : sortie envoyée vers l'extérieur intelligent comme sortie de la tâche 2

INT : entrée d'interruption d'un calculateur

Figure 8 schéma général du système expérimental

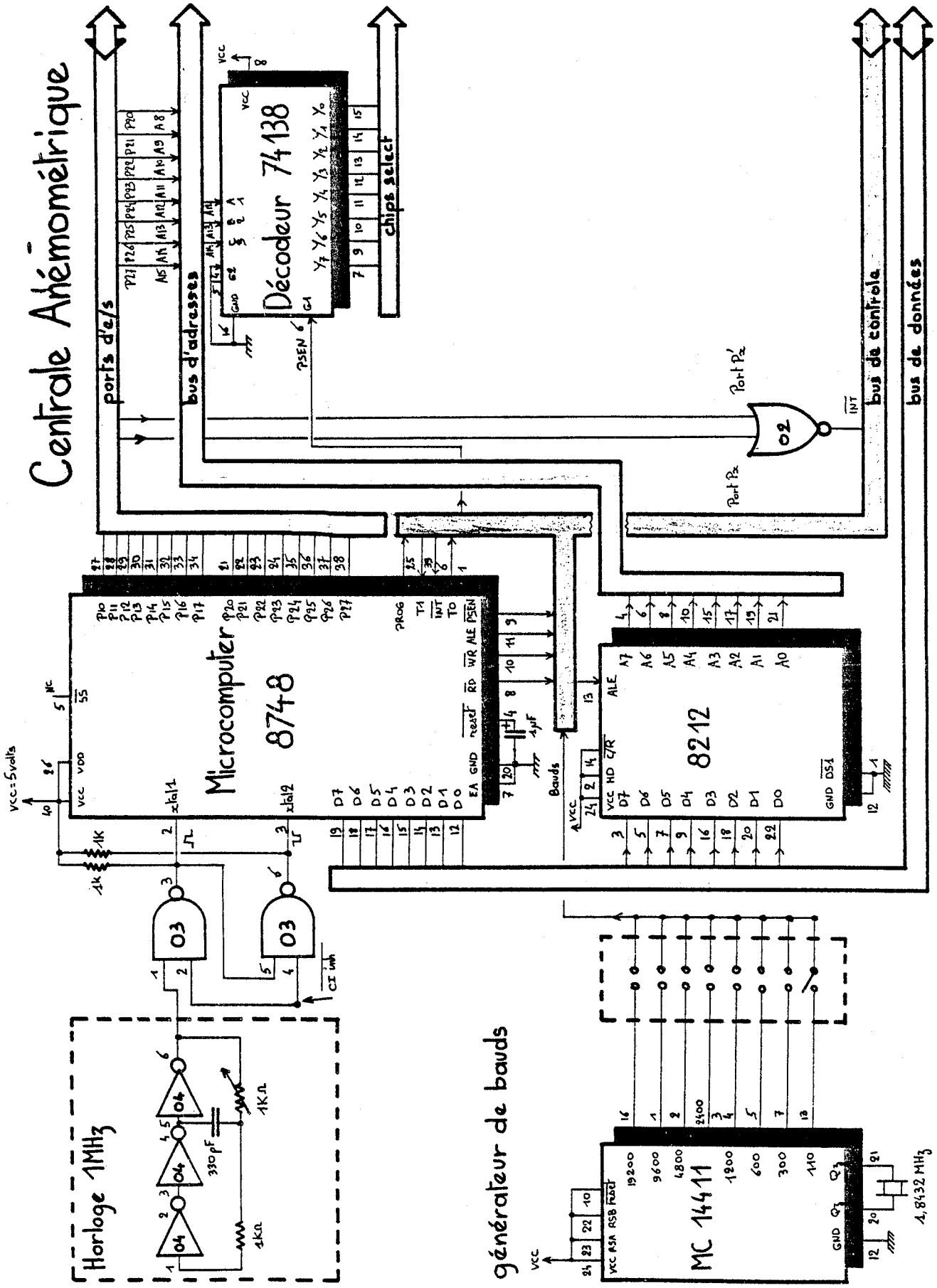
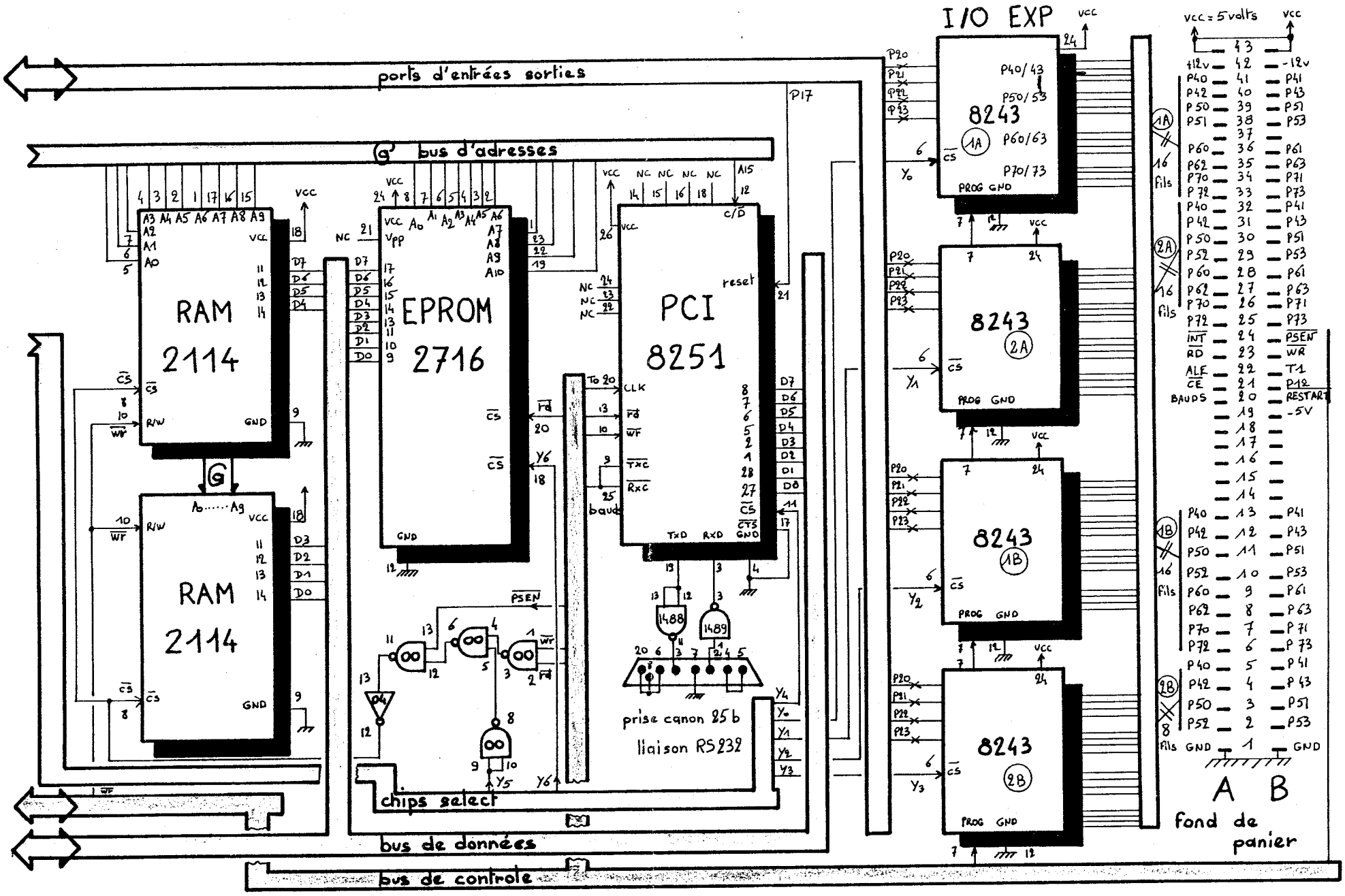


Figure 9 : Schéma d'une carte


ports d'entrées sorties
bus d'adresses
RAM 2114
RAM 2114
EPROM 2716
PCI 8251
I/O EXP
8243
8243
8243
8243
prise canon 25 b
liaison RS232
chips select
bus de données
bus de controle
fond de panier
A B
vcc = 5 volts
+12v
-12v
-5V
baud

C H A P I T R E I V

INTRODUCTION A LA METHODE D'EVALUATION

IV.1 - SITUATION DU PROBLEME

Nous nous proposons d'étudier, dans cette partie, le problème de l'évaluation de la sûreté de fonctionnement d'un système. Cette évaluation se situe dans le cadre du processus de validation du système et vise à répondre à la question suivante :

le système proposé satisfait-il aux spécifications de sûreté de fonctionnement requises ?

Il s'agit donc d'une évaluation absolue qu'il convient de distinguer d'évaluations relatives qui auraient par exemple pour objectif de comparer diverses solutions entre elles. Toute évaluation est faite par rapport à un modèle lequel est construit à partir d'un certain nombre d'hypothèses. Mais, dans le cas d'évaluations relatives, il peut suffire que les différentes solutions soient évaluées à partir du même jeu d'hypothèses, alors que dans le cas d'une évaluation absolue, il convient de "démontrer" l'adéquation des hypothèses utilisées au système étudié. En fait, il est peu vraisemblable que de telles démonstrations puissent être établies formellement (démontrer qu'un modèle représente la réalité). Nous devrons donc nous contenter d'expliciter très clairement les hypothèses et d'en discuter soigneusement la "validité". Il conviendra alors que ces hypothèses apparaissent réalistes aux autorités de certification ou aux utilisateurs du système.

Ceci nous conduit à essayer de réduire l'ensemble de ces hypothèses en appliquant une démarche non optimiste d'évaluation (lorsqu'une incertitude se présente, lever cette incertitude en choisissant le pire cas plutôt qu'en formulant une hypothèse à justifier). La recherche d'une telle démarche pose cependant le problème suivant : plus l'ensemble des hypothèses est réduit, plus la confiance en l'évaluation est grande, mais aussi plus le résultat de l'évaluation est pessimiste et peut s'avérer inacceptable. Il conviendrait

donc de trouver un compromis entre le caractère raisonnable des hypothèses adoptées et la valeur du résultat d'évaluation. La recherche et la discussion d'un tel compromis constitue un élément essentiel de cette partie.

Cette discussion portera essentiellement sur les facteurs mal connus qui mettent en défaut les stratégies de tolérance aux pannes. L'approche classique dite de couverture [BOU71], consiste à regrouper ces facteurs (pannes de mode commun, pannes transitoires, latence de manifestation des pannes favorisant des erreurs multiples, pannes affectant les mécanismes de détection) en un seul, la couverture, quantifiée par la proportion c du taux de pannes correspondant aux pannes efficacement prises en compte par la stratégie.

Nous montrerons en conclusion du chapitre VII en quoi notre approche, qui consiste à détailler ces facteurs, nous semble plus adaptée (les auteurs de ces travaux ont eux-mêmes mis l'accent sur la nécessité de détailler ces facteurs).

IV.2 - OBJET DE L'ETUDE

Nous chercherons à approcher ce compromis dans le cadre de l'étude de problèmes de complexité croissante qui feront chacun l'objet d'un chapitre :

- Etude de redondances hybrides et séquentielles ("self purging") à implantation externe (chapitre V),
- Etude d'un duplex à implantation interne (chapitre VI)
- Etude de la stratégie "Calculateur" du bi-calculateur décrit dans le chapitre III (chapitre VII).

Le chapitre V permettra plus particulièrement de poser les problèmes à résoudre (modéliser le processus de manifestation des pannes) et d'approcher ce compromis en proposant l'étude et l'application de divers modèles aux stratégies étudiées. Les chapitres VI et VII traiteront de l'applicabilité des méthodes développées à des stratégies dont la gestion et l'analyse s'avèrent de nature plus complexe.

Toutes les analyses qui seront développées sont basées sur deux hypothèses de base, communes à tous les systèmes qui seront étudiés, qui permettent de délimiter le cadre de notre étude.

Nous les poserons comme suit :

H_o) Nous considérons comme pannes uniquement les défaillances franches ("catalectiques") survenant de façon aléatoire dans les composants électroniques.

L'étude ne considère donc pas

. Les catastrophes naturelles (séîsmes, foudre...)

. les erreurs de conception

. les "erreurs transitoires" dues à des phénomènes divers (parasites électromagnétiques, particules α des substrats...)

dont les effets devront être évalués par ailleurs.

H_1) Les composants considérés sont sensés avoir un taux de pannes constant, que nous supposerons connu, grâce à un modèle couramment admis : Military Handbook 217 B [MIL], Recueil de Fiabilité du CNET [CNET]. Cette hypothèse suppose notamment l'usage de composants ayant subi un vieillissement préalable (éventuellement accéléré) permettant d'éliminer les pannes de jeunesse et systématiquement remplacés avant qu'ils ne rentrent dans leur période d'usure.

Les stratégies étudiées sont des stratégies à redondance massive mettant en jeu n unités. Les analyses effectuées s'appuieront également sur l'hypothèse suivante, que le concepteur doit veiller à rendre admissible :

H_2) Les n unités sont correctement isolées entre elles et vis-à-vis des systèmes externes (ou sous-ensembles externes), de sorte que nous considèrerons que les processus d'occurrence de pannes dans chaque unité sont indépendants.

Enfin, il convient de préciser le contexte opérationnel dans lequel ces stratégies sont utilisées, d'une part parce que cela s'avère nécessaire pour construire les modèles d'évaluation, d'autre part, parce que nous constaterons que certaines hypothèses conduiront à des résultats acceptables dans certains

contextes et inacceptables dans d'autres. Nous considèrerons essentiellement deux "contextes extrêmes".

Contexte A

Ce contexte correspond au cas des systèmes embarqués : le système, initialement sans panne, est utilisé pour une durée de mission T, courte (exemple T = 10 heures pour un avion) pendant laquelle il ne peut pas être réparé. Nous nous intéresserons alors à sa fiabilité de mission R(T) ou à sa sécurité de mission S(T).

Contexte B

Il correspond au cas de systèmes terrestres, par exemple de contrôle industriel : un parc de systèmes fonctionne continûment.

Ces systèmes étant réparés ou changés quand :

- une panne est détectée en cours de fonctionnement,
- une erreur se matérialise dans l'environnement (incident, accident...),
- l'âge d'un système devient incompatible avec l'hypothèse H1 de taux de pannes constant.

La population tend alors vers un régime stationnaire limite, et nous nous intéresserons aux taux limite constants (taux d'insécurité TIS).

Le contexte A correspond au cas du bi-calculateur étudié et fera donc l'objet d'études détaillées dans les trois chapitres. Cependant, il convient de remarquer que, dans ce contexte, les analyses sont basées sur le fait qu'à l'instant t égal à zéro, le système est supposé sans panne. Ceci implique que le système peut-être parfaitement testé et éventuellement réparé entre deux missions consécutives. L'usage de composants à haute intégration rend cette hypothèse de plus en plus irréaliste. Nous l'utilisons ici comme "hypothèse d'école" tout en sachant que la situation A sera de plus en plus justiciable de modèles se rapprochant de ceux utilisés pour le contexte B : c'est pourquoi nous montrons, sur un exemple simple (chapitre V), comment l'approche proposée et développée pour le contexte A, s'applique pour le contexte B.

C H A P I T R E V

EVALUATION DE STRATEGIES

A IMPLANTATION EXTERNE

V.1 - DEFINITION DU SYSTEME

Considérons le système représenté à la figure 1 .

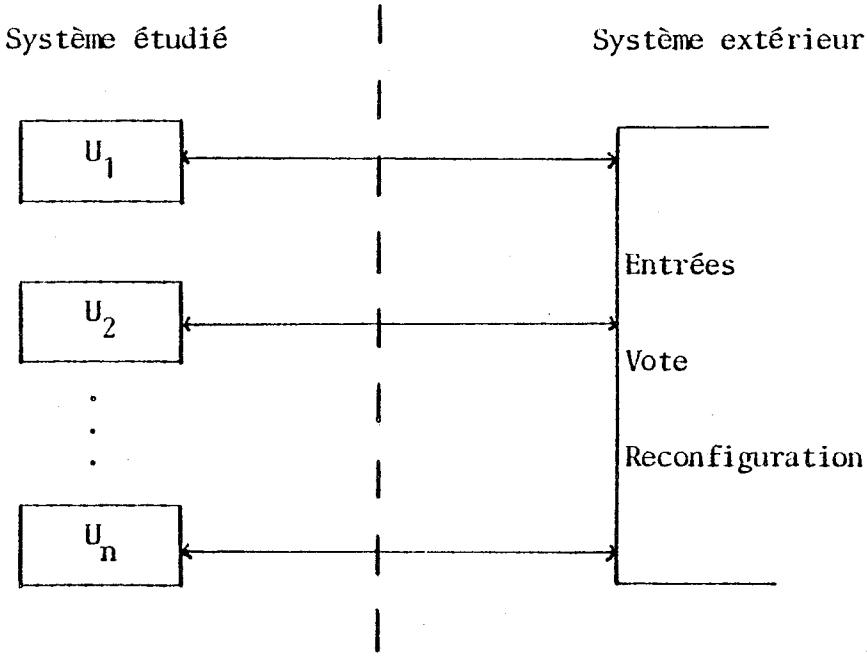


Figure 1 : Système à stratégie hybride ou séquentielle à décision externe.

Le système étudié est constitué de n unités communiquant avec un système externe qui leur fournit les entrées, qui vote sur leurs sorties et qui décide de la reconfiguration du système. Ce système externe peut être réalisé de nombreuses façons : redondant, mécanique fluidique, à relais ou logique de sécurité....

Nous supposons qu'il est soit parfaitement sûr, soit que sa sûreté de fonctionnement peut être évaluée isolément (système bien partitionné : voir chapitre II) : le problème des défaillances de vote, d'entrée ou de reconfiguration, n'a donc pas à être pris en compte dans notre analyse.

Les n unités du système étudié sont semblables et ont approximativement le même taux de pannes λ (les unités peuvent éventuellement être construites

à partir de composants différents pour éviter des pannes de mode commun ; de manière pessimiste, λ sera choisi comme étant le maximum des taux de panne des unités).

Nous étudierons sur ce système les stratégies suivantes :

. TMR avec n-3 réserves, noté aussi nH :

Le système externe vote sur les sorties de 3 unités actives. Si une unité présente une sortie en désaccord avec les sorties majoritaires (les deux autres unités), elle est désactivée et une réserve est activée à sa place. Le système s'arrête s'il n'y a pas de majorité.

. NMR séquentiel ("self purging") noté aussi nMRSP :

Le système externe vote sur les sorties de m (initialement n) unités. Toute unité qui diffère de la majorité est désactivée (c'est-à-dire n'est plus prise en compte par le système externe), de sorte que le nombre m diminue (la stratégie consiste successivement en un NMR puis (N-1)MR...). Le système s'arrête lorsqu'il n'y a pas de majorité.

. Duplex D : c'est le cas limite des deux stratégies précédentes : initialement, il y a deux unités actives ; dès qu'un désaccord apparaît, le système s'arrête. C'est sur cette stratégie que les analyses relevant du contexte B (parc de système en fonctionnement continu) seront appliquées. La stratégie NMR ordinaire (ou combinatoire) sera également utilisée à des fins de comparaison.

V.2 - MODELE 0 A LATENCE NULLE

Hypothèse implicite :

H3) Nous supposons que le temps qui sépare l'apparition d'une panne dans une unité et une erreur en sortie de cette unité (latence d'erreur), est très faible et peut être négligé.

Sous cette hypothèse, nous obtenons les évaluations classiques suivantes :

Contexte A

Posons $r = e^{-\lambda T}$ (r est la fiabilité de mission d'une unité)

.NMR combinatoire

Il fonctionne tant qu'une majorité d'unités fonctionne. En notant $\underline{x}$ la partie entière de x, nous obtenons :

$$R_{NMRO}(T) = \sum_{p=0}^{\underline{\frac{n-1}{2}}} C_n^p \, r^{n-p} \, (1-r)^p$$

.NMR séquentiel et TMR réserve

En supposant que des unités actives et inactives ont le même taux de pannes, (les réserves peuvent être actives sur des tâches non critiques : principe de dégradation progressive), ces systèmes donnent la même évaluation : les pannes étant immédiatement détectées dès que l'unité affectée est mise en service, le système fonctionne tant qu'il reste deux unités saines.

$$R_{NMRSPO}(T) = R_{NMO}(T) = \sum_{p=0}^{N-2} C_N^p \, r^{N-p} \, (1-r)^p$$

- Duplex

Le duplex est parfaitement sûr $S_{DO}(T) = 1$

Contexte B (Duplex)

Le duplex étant parfaitement sûr, nous obtenons le taux d'insécurité :

$TIS_{DO} = 0$

Ces modèles théoriques sont très largement utilisés pour des évaluations relatives ([BOU71], [MAT75]) mais semblent peu acceptables pour une certification car l'hypothèse H3 de temps de latence nul est clairement optimiste.

V.3 - MODELE PESSIMISTE 1

Cette dernière constatation conduit à essayer d'évaluer les systèmes sans ajouter d'hypothèse au jeu de base H_o, H_1, H_2 :

Contexte A

- Duplex

Pour une mission de durée T, les seuls cas pouvant conduire à une situation d'insécurité sont les cas où deux unités sont tombées en panne au cours de la mission. Pour ne pas poser d'hypothèse supplémentaire, la démarche pessimiste conduit à considérer que tous ces cas conduisent à une situation d'insécurité (toute première panne ne se manifeste qu'à l'apparition d'une deuxième panne et de manière cohérente avec celle-ci).

Nous obtenons l'insécurité de mission :

$$IS_1(T) = (1-e^{-\lambda T})^2$$

Soit pour T petit devant $\frac{1}{\lambda}$:

$$IS_1(T) = (\lambda T)^2$$

Nous remarquons que cette évaluation, bien que très pessimiste, ne donne pas de trop mauvais résultats :

ainsi, pour $\lambda = 10^{-5}\ h^{-1}$ et $T = 10\ h$

$$IS_1(T) = 10^{-8}$$

Ce type d'évaluation se généralise aux autres stratégies (avec des résultats moins favorables) en appliquant le principe pessimiste suivant :

Principe pessimiste :

les pannes ne se manifestent que dans les cas critiques où le principe de la comparaison ou du vote devient inefficace.

- TMR-réserves

Cette stratégie se ramène à la stratégie TMR ordinaire sans réserves :
en effet, si deux unités tombent en panne au cours de la mission, le principe pessimiste nous fait admettre qu'elles se manifestent simultanément de sorte que le voteur élimine systématiquement l'unité saine :

$$R_{NH1}(T) = R_{TMRO}(T) = 3r^2 - 2r^3 \quad (\text{où } r = e^{-\lambda T})$$

- NMR séquentiel

De même, cette stratégie se ramène à la stratégie NMR ordinaire :
selon le principe pessimiste, les pannes ne se manifestent que lorsqu'une majorité d'unités est tombée en panne : d'une part le vote est mis en défaut et laisse passer une erreur, et d'autre part les unités saines sont éliminées.

$$R_{NMRSP1}(T) = R_{NMRO}(T) = \sum_{p=0} C_n^p \, r^{n-p} \, (1-r)^p \qquad (\text{où } r = e^{-\lambda T})$$

Contexte B (Duplex)

La stratégie Duplex en situation B peut se représenter par le modèle de Markov de la figure 2

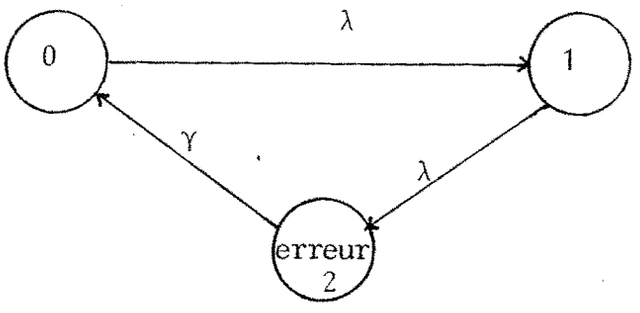


Figure 2 : Duplex externe-contexte B-modèle 1

L'état 0 correspond à "deux unités saines"

L'état 1 correspond à "une des deux unités en panne, l'autre saine"

La transition de l'état 0 à l'état 1 correspond à l'apparition d'une panne dans l'une des unités : son taux est donc 2λ.

La transition de l'état 1 à l'état d'erreur se produit dès que la deuxième panne apparait (principe pessimiste) ; son taux est λ.

Le système est alors remis à neuf avec un temps de réparation négligeable (taux γ très grand) et se trouve alors dans l'état 0.

Ce système atteint un régime stationnaire, et en désignant par $P_i(\infty)$ la probabilité qu'a le système d'être dans l'état i en régime stationnaire, nous montrons que :

$$TIS_1 = \lambda \; P_1(\infty)$$

Démonstration

Par définition, nous avons :

$$TIS(t) = \lim_{\Delta \to 0} \frac{1}{\Delta} \text{ Prob } [E(t + \Delta) = 2 \; / \; E(t) \neq 2]$$

et $$\lambda = \lim_{\Delta \to 0} \frac{1}{\Delta} \text{ Prob } [E(t + \Delta) = 2 \; / \; E(t) = 1]$$

Or : $$\text{Prob } [E(t + \Delta) = 2 \; / \; E(t) \neq 2] = \frac{\text{Prob } [E(t + \Delta) = 2, \; E(t) \neq 2]}{\text{Prob } [E(t) \neq 2]}$$

Et, comme les états sont exclusifs :

$$\frac{\text{Prob}\,[E(t+\Delta) = 2,\ E(t) \neq 2]}{\text{Prob}\,[E(t) \neq 2]} = \frac{\text{Prob}\,[E(t+\Delta) = 2,\ E(t) = 0]}{\text{Prob}\,[E(t) \neq 2]} + \frac{\text{Prob}\,[E(t+\Delta) = 2,\ E(t) = 1]}{\text{Prob}\,[E(t) \neq 2]}$$

Le deuxième terme s'écrit :

$$\text{Prob}\,[E(t+\Delta) = 2 \,/\, E(t) = 1] \cdot \frac{\text{Prob}\,[E(t) = 1]}{\text{Prob}\,[E(t) \neq 2]}$$

En divisant par Δ et en faisant tendre Δ vers 0, on constate que le premier terme tend vers 0 car il n'y a pas de transition directe de 0 à 2, et que :

$$TIS(t) = \lambda \cdot \frac{\text{Prob}\,[E(t) = 1]}{\text{Prob}\,[E(t) \neq 2]} = \lambda \cdot \frac{P_1(t)}{1 - P_2(t)}$$

et en régime stationnaire :

$$TIS = \lambda \cdot \frac{P_1(\infty)}{1 - P_2(\infty)}$$

Par hypothèse, nous avons considéré γ très petit, de sorte que :

$P_2(\infty) \simeq 0$

d'où, $TIS = \lambda\, P_1(\infty)$

Après résolution du système d'équation représentant le graphe, nous obtenons :

$$TIS_1 = \frac{2\lambda}{3}$$

(notons enfin que l'hypothèse de temps de réparation nul conduit à construire un graphe équivalent au graphe de la figure 3).

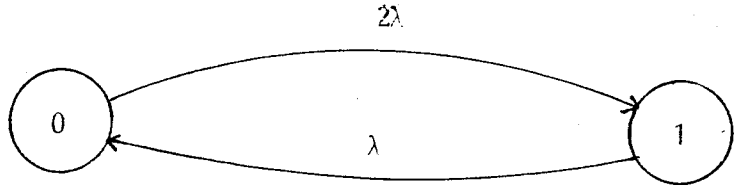


Figure 3 : Duplex externe-contexte B-modèle 1'(simplifié)

Dans ce contexte, le résultat s'avère extrêmement pessimiste, puisqu'un système simplex aurait, sous les mêmes hypothèses, un taux d'insécurité égal à λ

Ce résultat montre la difficulté de se passer d'hypothèses plus favorables sur la façon dont les pannes se manifestent sous forme d'erreurs.

V.4 - MODELE A TAUX D'ERREUR UNIQUE

V.4.1. Principe

Ce type de modèle a été utilisé pour l'évaluation du calculateur SIFT ([WEN78]). Le processus de manifestation des pannes sous forme d'erreurs est décrit par l'hypothèse H4 :

H4) La latence d'erreur a une distribution indépendante du temps, exponentielle à taux constants μ.

Cette hypothèse permet de conserver un modèle Markovien et introduit un nouveau paramètre μ qu'il conviendra d'estimer ou de traiter par pire cas. Elle s'appuie sur l'indépendance de processus de manifestation par rapport au temps, ce qui suppose un environnement engendrant des entrées aléatoires stationnaires (cette hypothèse sous-jacente doit être discutée) ; enfin, elle semble criticable, eût égard aux travaux sur la latence d'erreur [SHE75]. Cette critique justifiera l'étude des modèles suivants ; néanmoins, l'application de cette hypothèse semble intéressante car elle constitue une bonne introduction à ces modèles.

V.4.2. Application à la stratégie duplex-contexte A

V.4.2.1. Construction du modèle

Un modèle incomplet peut être initialement construit comme suit (figure 1).

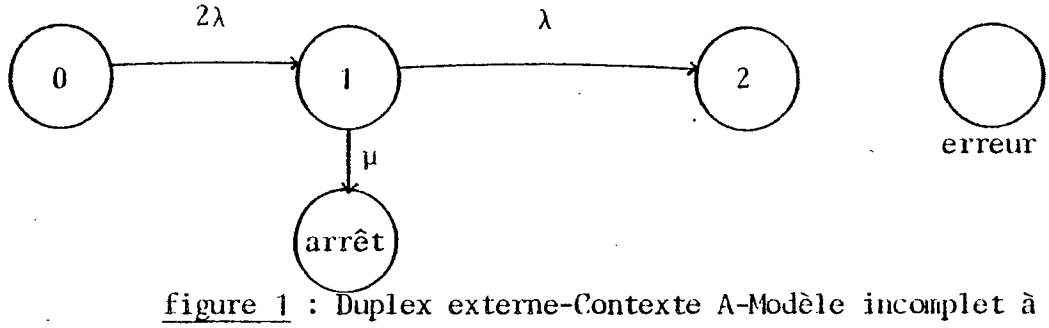


figure 1 : Duplex externe-Contexte A-Modèle incomplet à taux d'erreur unique.

0 : état initial : 2 unités saines

1 : une unité en panne latente

2 : les deux unités en panne latente.

La manifestation de la panne latente de l'état 1 (évènement de taux μ) amène à l'état d'arrêt.

A partir de l'état 2, trois types d'évènements (auxquels il faudra affecter un taux pour compléter le modèle) peuvent se produire :

a) les deux pannes se manifestent à des instants différents : cet évènement conduit à l'arrêt.

b) les deux pannes se manifestent simultanément mais produisent des erreurs distinctes : cet évènement conduit à l'arrêt.

c) les deux pannes produisent simultanément la même erreur, qui se propage vers l'extérieur.

La distinction entre ces trois types d'évènements exigerait que les mécanismes de manifestation des pannes ainsi que les modes de pannes des unités, soient étudiés en détail : cela nécessiterait d'enrichir l'ensemble des hypothèses, ce qui serait contraire au principe de la démarche adoptée. Si nous nous limitons au jeu d'hypothèses (H_o, H_1, H_2, H_4), dans le cadre d'une démarche non optimiste, nous sommes conduits à supposer que tous les évènements sont de type c) :

Principe pessimiste :

Dans les situations critiques, les pannes se manifestent simultanément en produisant la même erreur.

Le modèle peut être complété comme suit :

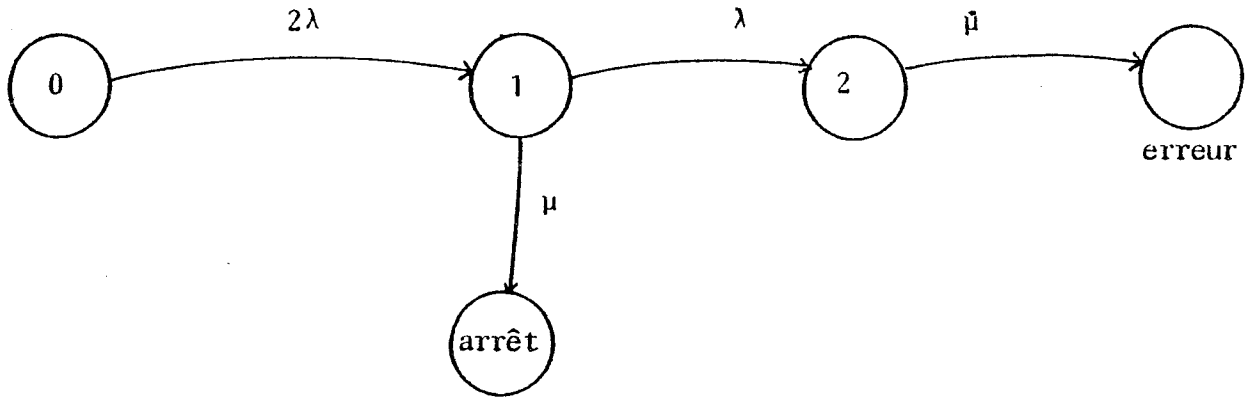


Figure 2 : Duplex externe - Contexte A - Modèle à taux d'erreur unique

V.4.2.2. Calcul du modèle

Les conditions initiales sont :

$P_o(0) = 1$; $P_1(0) = P_2(0) = P_{arrêt}(0) = P_{erreur}(0) = 0$

Ce modèle peut être décrit par le système d'équation suivant :

$$\frac{dP_o(t)}{dt} = -2\lambda P_o(t)$$

$$\frac{dP_1(t)}{dt} = 2\lambda P_o(t) - (\lambda+\mu) P_1(t)$$

$$\frac{dP_2(t)}{dt} = \lambda P_1(t) - \mu P_2(t)$$

$$\frac{dP_{arrêt}(t)}{dt} = \mu P_1(t)$$

$$\frac{dP_{erreur}(t)}{dt} = \mu P_2(t)$$

$$P_o(t) + P_1(t) + P_2(t) + P_{arrêt}(t) + P_{erreur}(t) = 1$$

En appliquant la transformation de Laplace (où $\tilde{f}(p)$ est la transformée de f(t), p étant la variable de Laplace), le système s'écrit :

$$p\tilde{P}_o - 1 = -2\lambda \tilde{P}_o$$

$$p\tilde{P}_1 = 2\lambda \tilde{P}_o - (\lambda+\mu) \tilde{P}_1$$

$$p\tilde{P}_2 = \lambda \tilde{P}_1 - \mu \tilde{P}_2$$

$$p\tilde{P}_{arrêt} = \mu \tilde{P}_1$$

$$p\tilde{P}_{erreur} = \mu \tilde{P}_2$$

$$(\tilde{P}_o + \tilde{P}_1 + \tilde{P}_2 + \tilde{P}_{arrêt} + \tilde{P}_{erreur} = \frac{1}{p})$$

La résolution du système donne :

$$\tilde{P}_{erreur} = \frac{2\lambda^2 \mu}{p(p+\mu)(p+\mu+\lambda)(p+2\lambda)}$$

Comme nous ne nous intéressons qu'aux valeurs de T telles que λT soit petit, nous pouvons obtenir un résultat approché en ne considérant que le premier terme du développement limité de $\tilde{P}_{erreur}$ au voisinage de $\lambda = 0$.

$$\tilde{P}_{erreur}/_{\lambda=0} = 0$$

$$\frac{\partial \tilde{P}_{erreur}}{\partial \lambda}/_{\lambda=0} = 0$$

$$\frac{\partial^2 \tilde{P}_{erreur}}{\partial \lambda 2}/_{\lambda=0} = \frac{4\mu}{p^2(p+\mu)^2}$$

Nous obtenons donc :

$$\tilde{P}_{erreur} \simeq \frac{2\mu\lambda^2}{p^2(p+\mu)^2}$$

La décomposition en éléments simples donne :

$$\tilde{P}_{erreur} \simeq 2\lambda^2\left[\frac{-\frac{2}{\mu^2}}{p} + \frac{\frac{1}{\mu}}{p^2} + \frac{\frac{1}{\mu^2}}{p+\mu} + \frac{\frac{1}{\mu}}{(p+\mu)^2}\right]$$

Et par passage à l'inverse :

$$IS_2(T) = P_{erreur}(T) = 2\lambda^2\left[-\frac{2}{\mu^2} + \frac{T}{\mu} + \frac{2}{\mu^2}e^{-\mu T} + \frac{T}{\mu}e^{-\mu T}\right]$$

Nous poserons dans la suite $F(T,\mu) = -\frac{2}{\mu^2} + \frac{T}{\mu} + \frac{2}{\mu^2}e^{-\mu T} + \frac{T}{\mu}e^{-\mu T}$

La figure 3 donne la courbe de P_{erreur} en fonction de μ pour

$T = 10$ heures et $\lambda = 10^{-5}$ heures^{-1}

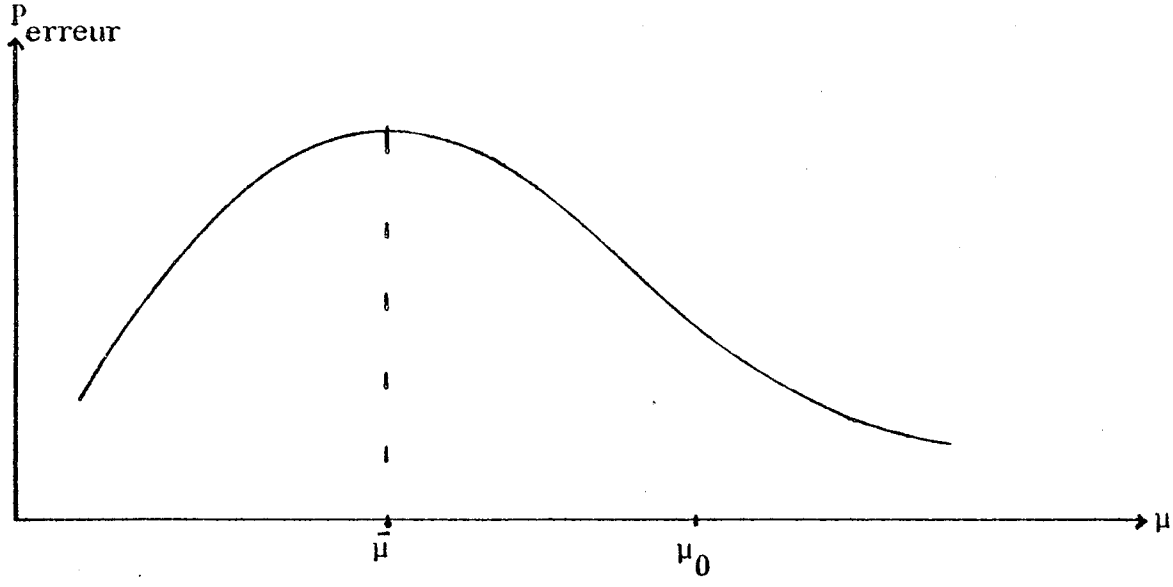


Figure 3 : $IS_2 = f(\mu)$

V.4.2.3. Utilisation du modèle

L'examen de la courbe amène deux remarques préliminaires :

a) Cette courbe a un maximum $\bar{\mu}$: son existence peut être perçue en constatant que μ varie entre 0 et l'infini et que pour ces cas limites l'insécurité est nulle :

· pour μ_∞ : les pannes se manifestent immédiatement (voir modèle 0)

· pour $\mu=0$: les pannes ne se manifestent jamais

En résolvant l'équation $\frac{\partial P_{erreur}}{\partial \mu} = 0$, nous obtenons :

$$\bar{\mu} = \frac{2,7}{T}$$

b) Pour $\mu > \bar{\mu}$ la courbe est décroissante et l'insécurité tend vers 0 lorsque μ tend vers l'infini ; de plus, pour μT grand devant 1, nous avons :

$$IS_2(T) \simeq 2\lambda^2 \frac{T}{\mu}$$

Ces deux remarques montrent qu'il y a deux utilisations possibles du modèle :

· Approche de pire cas

Pour ne pas ajouter d'hypothèses, nous considèrerons que l'insécurité est la pire insécurité, c'est-à-dire :

$$IS_2(T) = IS_2(T,\bar{\mu})$$

Ainsi pour $\lambda = 10^{-5}\ h^{-1}$ et $T = 10\ h$

$$IS_2(T) = 0,28.10^{-8}$$

Cette évaluation n'améliore que très peu celle obtenue par le modèle pessimiste 1 (d'un facteur 3, ce qui n'est guère significatif).

Hypothèse numérique sur μ

Si la réalisation du système permet de postuler, de manière réaliste, l'existence d'une borne minimum pour μ, μ_o grande devant $\bar{\mu}$ (assurant que la latence est inférieure à une latence maximum $\frac{1}{\mu_o}$), alors nous obtenons :

$$IS_2(T) = IS_2(T,\mu_o)$$

et si $\mu_o T$ est grand devant 1, nous obtenons :

$$IS_2(T) \simeq 2\lambda^2 \frac{T}{\mu_o}$$

V.4.3. Stratégie 4MRSP

Le modèle de cette stratégie est donné par la figure 4 :

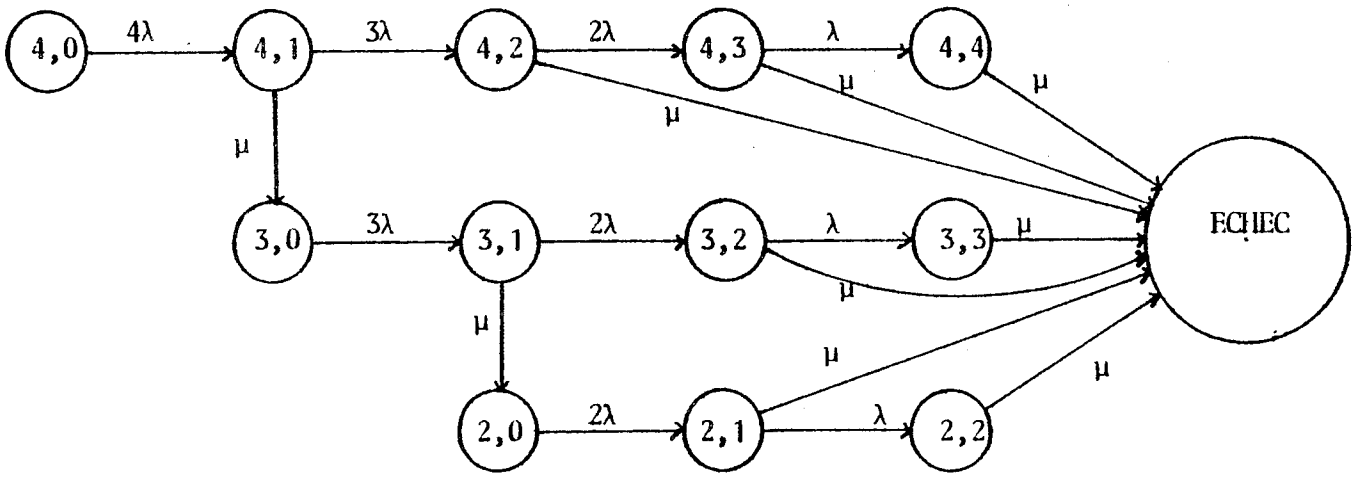


Figure 4 : 4MRSP - Modèle à taux d'erreur unique

Les états sont indicés par un couple (n_a, n_p) :

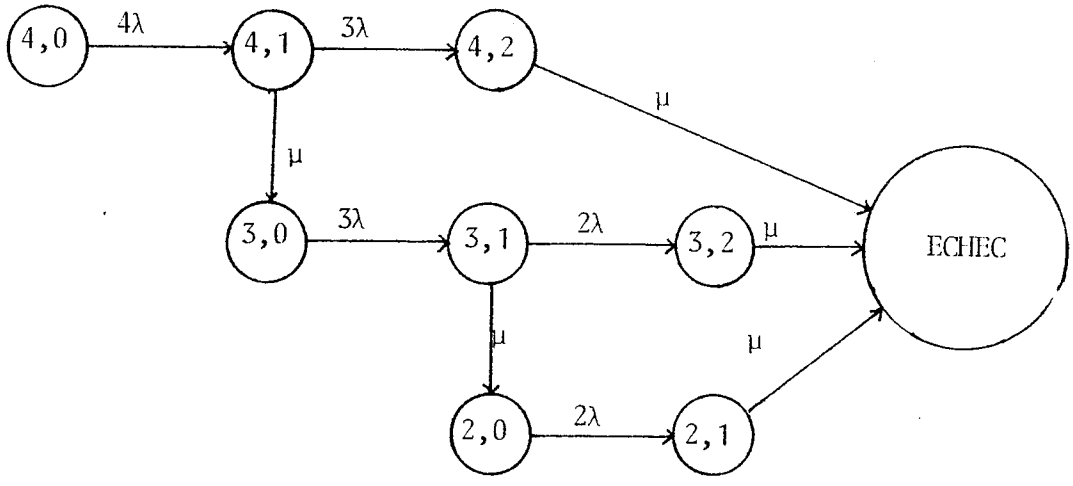
n_a = nombre d'unités actives

n_p = nombre d'unités en panne

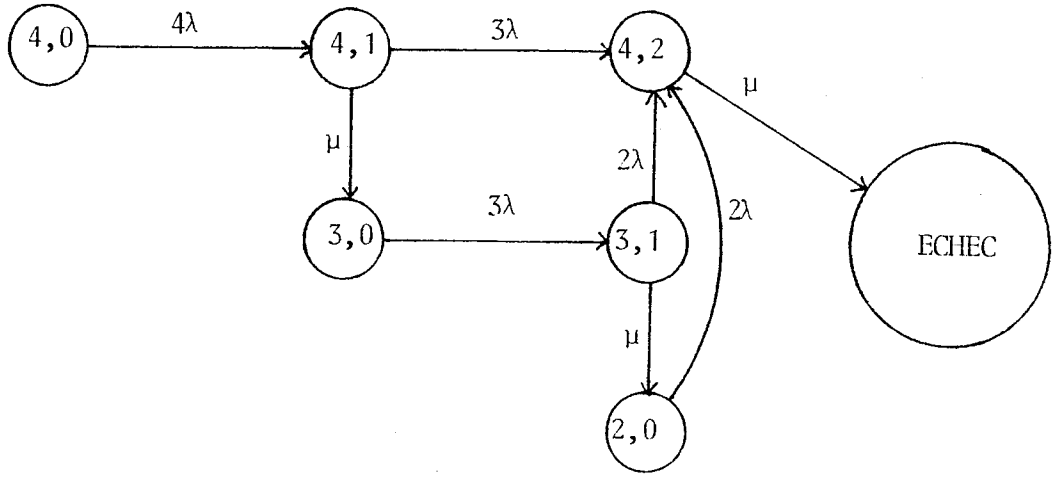
Comme le paramètre à évaluer est la fiabilité, les états "erreur" et "arrêt" ne sont pas distingués et sont regroupés dans l'état "échec". De plus, seule la probabilité de cet état "échec" (l'infiabilité) nous intéresse. Nous pouvons donc, par une technique classique, fusionner les états ayant même futur :

nous pouvons ainsi fusionner les états (4,2), (4,3) et (4,4) en un état (4,2), les états (3,2) et (3,3) en un état (3,2) et les états (2,1) et (2,2) en un état (2,1). Nous pouvons ensuite regrouper ces trois groupements (4,2), (3,2) et (2,1) en un seul (4,2). Enfin, les états (3,1) et (2,0) peuvent être regroupés. La figure 5 présente cette réduction :

a)



b)



c)

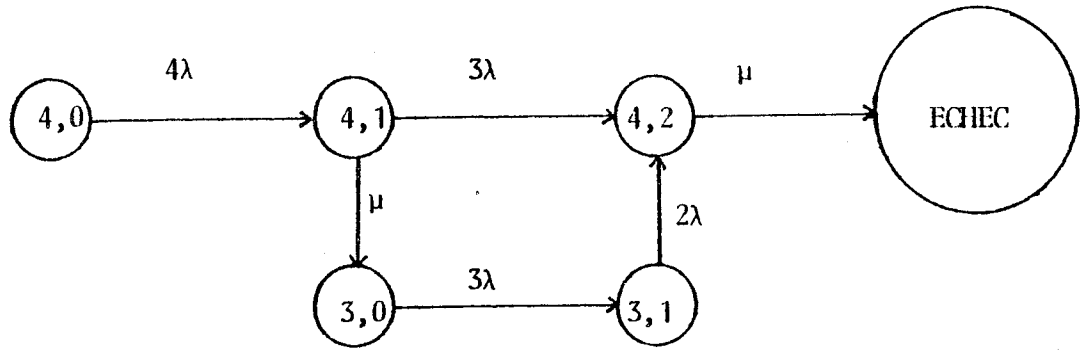


figure 5 : 4MRSP : réduction du modèle à taux d'erreur unique

Une analyse similaire à celle présentée en V.4.2. donne :

$$\tilde{P}_{\text{échec}} = \frac{\mu}{p}\tilde{P}_{4,2} = \frac{\mu}{(\mu+p)p}(3\lambda\tilde{P}_{4,1} + 2\lambda\tilde{P}_{3,1})$$

L'analyse au premier ordre du premier terme $\frac{3\mu\lambda}{p(\mu+p)}\tilde{P}_{4,1}$

donne l'expression : $12\ \lambda^2\ F(T,\mu)$

Le deuxième terme donne une expression de l'ordre de λ^3 qui ne peut pas être négligé : des valeurs grandes de μ peuvent rendre cette quantité supérieure au premier terme. Ce deuxième terme représente la probabilité qu'une succession de trois pannes conduise à l'échec (chemin(4,0) → (4,1) → (3,1) → (4,2) → échec) : cette probabilité est majorée par $IR_{4MRSPO}(T)$ qui, pour les faibles valeurs de T vaut $4(\lambda T)^3$ (choix de 3 pannes parmi 4).

Nous obtenons :

$$P_{\text{échec}}(T) = 12\ \lambda^2\ F(T,\mu) + 4(\lambda T)^3$$

Nous constatons que $P_{échec}(T)$ a la même dépendance en μ que l'insécurité du duplex étudié en V.4.2. Les mêmes conclusions s'imposent ; il y a deux utilisations possibles du modèle : approche de pire cas et hypothèse numérique sur μ.

V.4.4. Stratégie TMR à une réserve

Le modèle est le suivant (figure 6).

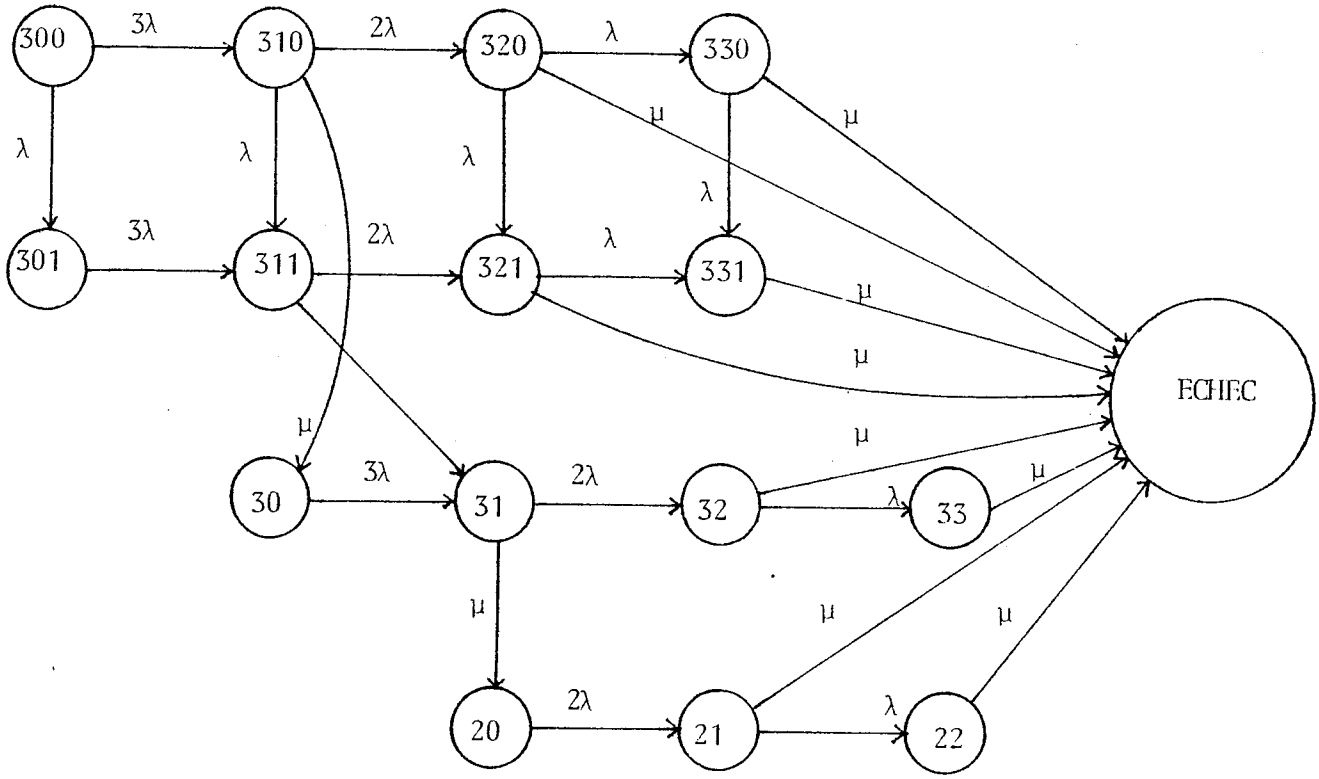


Figure 6 : TMR à une réserve - Modèle à taux d'erreur unique.

Les états matériels sont représentés par un triplet (n_a, n_p, r)

où : n_a : nombre d'unités actives

n_p : nombre d'unités actives en panne latente

r : nombre binaire : 0 réserve saine

1 réserve en panne

Ce paramètre r disparaît quand la réserve devient active.

Une réduction par fusion (fusion de (3,2,0), (3,3,0), (3,2,1), (3,3,1) (3,2), (3,3), (2,1), (2,0) d'une part et de (3,1,1), (3,1), (2,0)) et une analyse similaire à celles des paragraphes précédents donnent :

$P_{échec}(T) = 6\lambda^2 F(T,\mu) + 4 (\lambda T)^3$

Cette expression est semblable à celles trouvées pour le duplex et le 4MRSP : les mêmes analyses s'y appliquent donc.

V.4.5. Stratégie TMR à deux réserves

Le modèle correspondant est complexe et ne sera pas donné ici. Il paraît cependant clair que la fiabilité du TMR à deux réserves est limitée par la même suite d'évènements que le TMR à une réserve : panne de deux unités du TMR initial se manifestant simultanément de façon cohérente ; cette suite d'évènements correspond au terme $6\lambda^2 F(T,\mu)$.

Nous avons donc :

$P_{échec}(T) = 6\lambda^2 F(T,\mu) + o(\lambda T)^3$

Les mêmes analyses s'appliquent donc encore. De plus, il apparaît que, sauf pour des valeurs très grandes de μ (latence très faible : on se rapproche du modèle 0) qui rendent $6\lambda^2 F(T,\mu)$ comparable à $o(\lambda T)^3$, le TMR à deux réserves n'aura donc pas une fiabilité meilleure que le TMR à une réserve.

V.4.6. Stratégie 5MRSP

Le modèle utilisant la même notation que celui du 4MRSP (V.4.3.), est donné par la figure 7 .

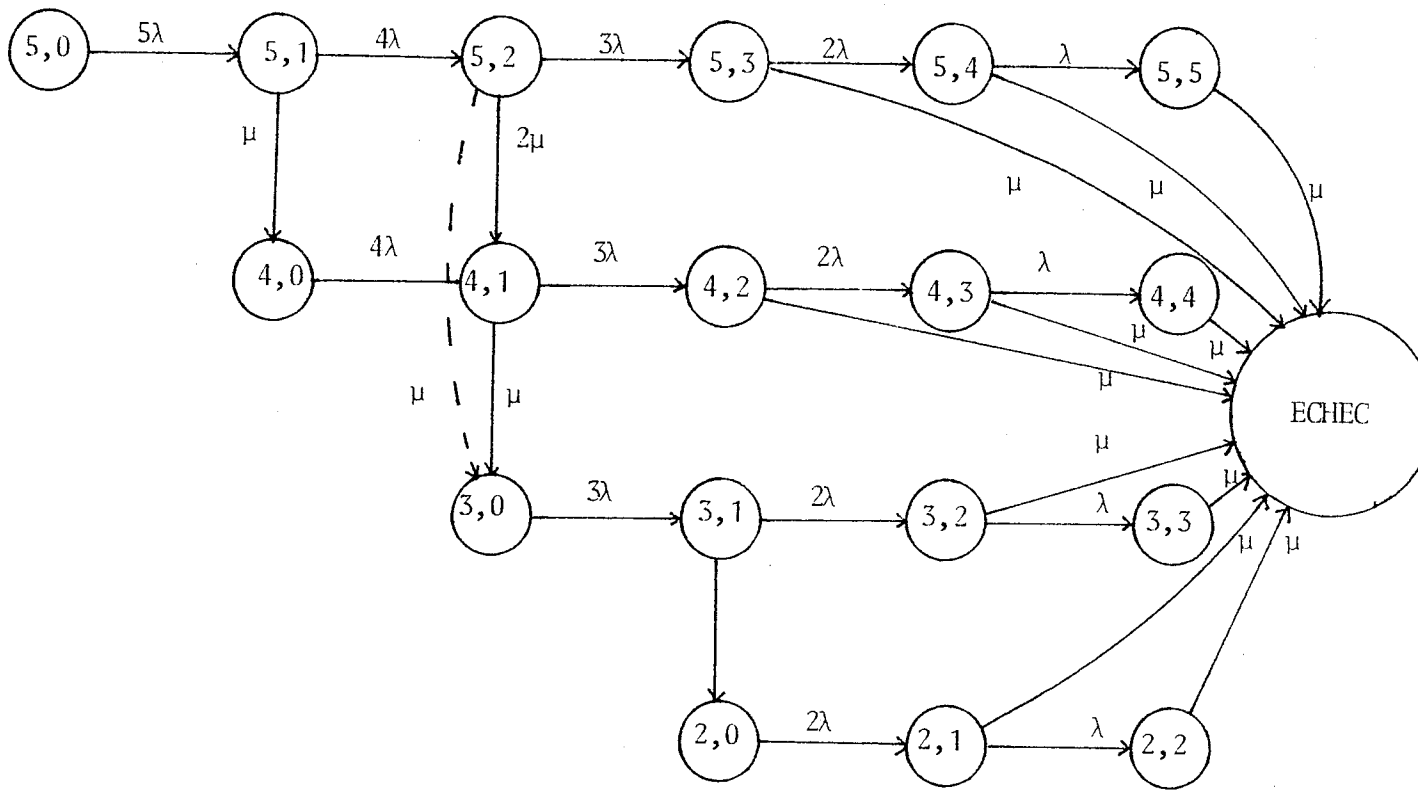


Figure 7 : 5MRSP - Modèle à taux d'erreur unique.

Le seul problème rencontré dans la construction de ce modèle, concerne l'état (5,2) : la manifestation des deux pannes latentes, même si elle est simultanée et cohérente, n'est pas critique pour le système (elle amène l'élimination des deux unités et conduit à l'état (3,0)) ; nous avons fait ici un choix plus pessimiste en considérant que :

"Dans l'état (5,2), chaque panne se manifeste indépendamment de l'autre. Le taux de manifestation est alors 2μ et la manifestation conduit à l'état (4,1) (une panne reste latente)".

Ce choix n'est pas homogène avec le principe adopté dans le cas des situations critiques (principe pessimiste énoncé en V.4.2.1.). L'homogénéité aurait consisté à tracer l'arc dessiné en pointillés sur la figure : ce choix serait cependant plus optimiste (avec le choix effectué, il faut deux

transitions de taux 2μ et 3λ pour atteindre un état dangereux (4,2) alors qu'avec ce dernier, il faudrait trois transitions de taux μ, 3λ , 2λ pour atteindre un état dangereux (3,2)). La même méthode d'analyse et de calcul que précédemment est appliquée à ce modèle ; nous obtenons :

$$P_{\text{échec}}(T) = 60\lambda^3 F'(T,\mu) + 5\,\lambda T^4$$

$$\text{avec } F'(T,\mu) = \frac{5}{4\mu^3} - \frac{3T}{2\mu^2} + \frac{T^2}{2\mu} - \frac{e^{-\mu T}}{\mu^3} + \frac{T^2 e^{-\mu T}}{2\mu} - \frac{e^{-2\mu T}}{4\mu^3}$$

De même que la fonction F rencontrée dans les précédentes analyses, F' admet un maximum en $\bar{\mu}$.

$$\bar{\mu} = \frac{3.5}{T} \quad \text{et} \quad F'(T,\bar{\mu}(T)) = 0{,}05316\ T^3$$

De même que précédemment, nous pouvons :

- Appliquer l'approche de pire cas en prenant

$$IR_{5MRSP2} = 60(\lambda T)^3 \times 0{,}05316$$

- Poser une hypothèse numérique sur μ : $\mu \geq \mu_o$. Si $\mu_o T$ est grand devant 1, nous obtenons :

$$IR_{5MRSP2} = 30\,\frac{\lambda^3 T^2}{\mu_o} + 5(\lambda T)^4$$

Notons enfin que l'utilisation du modèle dit "homogène" (arc en pointillés) donne des résultats très semblables :

$$\bar{\mu} = \frac{3{,}7}{T} \ ; \ F''(T,\bar{\mu}(T)) = 0.045.T^3$$

Dans le cas où T est grand devant l'unité, le résultat est même identique :

$$P_{\text{échec}}(T) = 30\lambda^3 \frac{T^2}{\mu} + 5(\lambda T)^4$$

V.4.7. Conclusion sur les analyses du contexte A

Le tableau de la figure 8 résume les résultats obtenus dans le contexte A pour les différentes stratégies étudiées ; il présente les résultats obtenus pour les valeurs = 10^{-5} h^{-1}, T = 10 heures à l'aide des modèles optimiste et pessimiste et du modèle à taux d'erreur unique traité par pire cas.
La lecture de ce tableau amène les remarques suivantes :

- l'approche de pire cas sur le modèle à taux d'erreur n'améliore que très peu les résultats du modèle pessimiste : cette amélioration correspond à un facteur 3 (c'est-à-dire moins qu'un ordre de grandeur).
- les latences moyennes correspondant au pire cas ($\frac{1}{\mu}$) n'ont rien de déraisonnables: leur ordre de grandeur est l'heure.
- dans le modèle à taux d'erreur traité par pire cas, il s'avère que :
  . le TMR à une réserve (4H) est plus fiable que le 4MRSP
  . l'adjonction d'une réserve supplémentaire (5H) n'améliore pas la fiabilité : pour ce modèle, une réserve semble être un optimum.
  . en revanche, la fiabilité de la stratégie nMRSP croît avec le degré de redondance n.

Grandeur évaluée	Redondance (ou stratégie)	MODELE OPTIMISTE	MODELE PESSIMISTE	MODELE A TAUX D'ERREUR (PIRE CAS)		
				résultat	$\bar{\mu}$ (h^{-1})	latence $\frac{1}{\bar{\mu}}$ (h)
Insécurité	Duplex	0	10^{-8}	$0,28.10^{-8}$	0,27	3,68
Infiabilité	TMR	3.10^{-8}	3.10^{-8}	3.10^{-8}	∞	0
Infiabilité	5MR	10^{-11}	10^{-11}	10^{-11}	∞	0
Infiabilité	4H	4.10^{-12}	3.10^{-8}	$0,84.10^{-8}$	0,27	3,68
Infiabilité	5H	5.10^{-16}	3.10^{-8}	$0,84.10^{-8}$	0,27	3,68
Infiabilité	4MRSP	4.10^{-12}	6.10^{-8}	$1,68.10^{-8}$	0,27	3,68
Infiabilité	5MRSP	5.10^{-16}	10^{-11}	$3,2.10^{-12}$	0,35	2,84

Figure 8 : Tableau récapitulatif des résultats obtenus à l'aide des 3 modèles dans le contexte A.

V.4.8. Stratégie Duplex-Contexte B

Le modèle obtenu par transformation du modèle du contexte A, est donné dans la figure

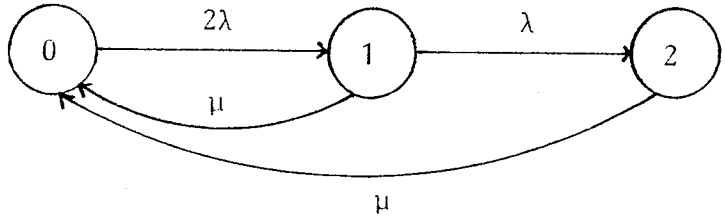


Figure : Duplex externe - Contexte B - Modèle à taux d'erreur unique.

Le taux d'insécurité est le taux des évènements "transition de l'état 2 à l'état initial" (le principe pessimiste nous fait considérer que, dans l'état de double panne latente, toutes les manifestations sont catastrophiques).

$$TIS_2 = \mu \, P_2(\infty)$$

Le calcul du régime permanent donne :

$$TIS_2 = \mu \, P_2(\infty) = \lim_{p \to 0} \; p \, \mu \, \tilde{P}_2 = \frac{2\lambda^2 \mu}{\mu^2 + 3\lambda\mu + 2\lambda^2}$$

De même que précédemment, deux approches sont possibles :

Approche de pire cas

TIS_2 admet un maximum en μ donné par

$\bar{\mu} = \sqrt{2} \, \lambda$

auquel correspond $TIS_2 = \frac{2\sqrt{2} \, \lambda}{4+3\sqrt{2}} \simeq 0{,}343 \, \lambda$

L'amélioration par rapport au modèle pessimiste (V.3) est faible (d'un rapport 2) ; le taux d'erreur $\bar{\mu}$ considéré est très faible (de l'ordre du taux de panne), ce qui correspond à une latence moyenne très grande :

Le résultat apparaît donc comme très pessimiste.

Hypothèse numérique sur μ

Il est possible d'obtenir une évaluation efficace si on peut supposer μ supérieur à une borne μ_o, grande devant λ :

$$TIS_2 = \frac{2\lambda^2}{\mu}$$

V.4.9. Conclusion sur le modèle à taux d'erreur

Ce paragraphe a étudié le modèle à taux d'erreur proposé dans [CAS81b] et que nous avons également utilisé dans [CAS81c]. Ce modèle introduit dans l'évaluation un nouveau paramètre, le taux d'erreur des pannes, dont la valeur est difficile à estimer. Nous avons montré que dans les cas étudiés ici, une démarche pessimiste permet de se dispenser de cette évaluation, mais donne des résultats très médiocres, n'améliorant que très peu les résultats du modèle pessimiste précédent. Les évaluations peuvent être améliorées en posant des hypothèses numériques sur la valeur de ce taux d'erreur ; cependant, ces hypothèses posent le problème suivant : l'hypothèse sous-jacente à ce modèle postule que toutes les pannes ont le même taux de manifestation ; cette hypothèse n'est évidemment que peu réaliste : certaines pannes se manifestent immédiatement (composant "grillé") alors que d'autres ont une probabilité très faible de se manifester (pannes dont la manifestation a lieu pour une configuration très précise de l'état interne). Ce modèle n'est donc qu'une approximation de la réalité ; pour estimer ce taux d'erreur unique, il faut alors décider si :

- il correspond au plus petit taux de manifestation sur l'ensemble des pannes (il a alors peu de chance d'être nettement supérieur au taux de pire cas : le résultat risque de ne pas être satisfaisant).

- Il correspond au taux moyen sur l'ensemble des pannes et est alors vraisemblablement très grand (de nombreuses pannes ont une faible latence).

Rien ne permet, dans ce modèle, de décider entre ces deux possibilités : cela justifie l'étude approfondie présentée dans les modèles suivants (V.5, V.6).

V.5. MODELE A TAUX D'ERREUR DISTRIBUE

V.5.1. Hypothèse de base

Ce modèle est basé sur l'hypothèse H5, déduite en première approximation, de la théorie de la latence d'erreur [SHE75] :

H5) Chaque panne a une latence distribuée exponentiellement avec un taux d'erreur μ_p.

Cette hypothèse n'est qu'une approximation (on montre que les pannes de type séquentiel ont une distribution multiexponentielle et non exponentielle) ; elle doit être considérée comme un intermédiaire de raisonnement, qui permet d'esquisser des solutions au problème général posé par le modèle à latence d'erreur distribuée de façon quelconque, qui sera développé au V.6.

Considérons une unité de taux de panne λ et la suite des valeurs possibles des taux d'erreurs de ces pannes $(\mu_1,\ldots, \mu_i,\ldots)$, rangées de façon croissante. Nous désignons par λ_i le taux des pannes ayant pour taux d'erreur μ_i. Nous avons :

$$\lambda = \sum_i \lambda_i$$

et nous pouvons définir la distribution de probabilité de μ

$$P_\mu(\mu_i) = \text{Prob}(\mu=\mu_i) = \frac{\lambda_i}{\lambda}$$

Le problème consiste à obtenir des évaluations alors que cette distribution est inconnue.

V.5.2. Stratégie duplex-Contexte A

V.5.2.1. Construction du modèle

La figure 9 présente le modèle correspondant :

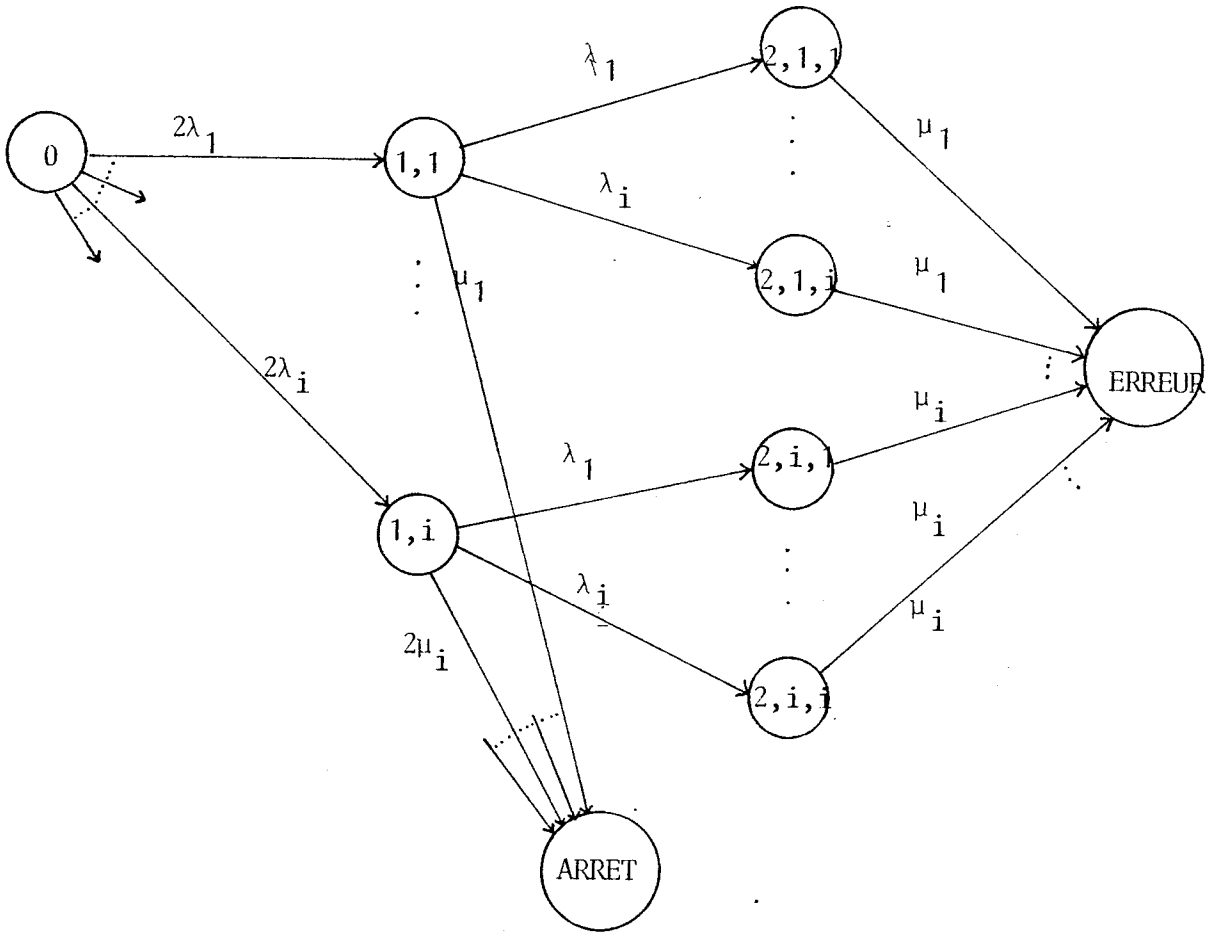


Figure 9 : Duplex externe - Contexte A - Modèle non réduit à taux d'erreur distribué

Nous avons détaillé dans cette figure les successions de types de pannes conduisant à l'erreur. Les états sont indicés par le nombre de pannes, suivi de leur type. Notons que par hypothèse pessimiste, nous avons considéré que les doubles pannes se manifestent au moment de la manifestation de la première panne apparue : les taux de manifestation menant à l'erreur sont indicés par le type de la première panne. Ce graphe se réduit par fusion des états (2,i,j) en un seul (2,i) où l'on entre avec le taux $(\lambda_1 + \lambda_2 + \ldots + \lambda_i + \ldots) = \lambda$. Le graphe réduit est donné dans la figure 10.

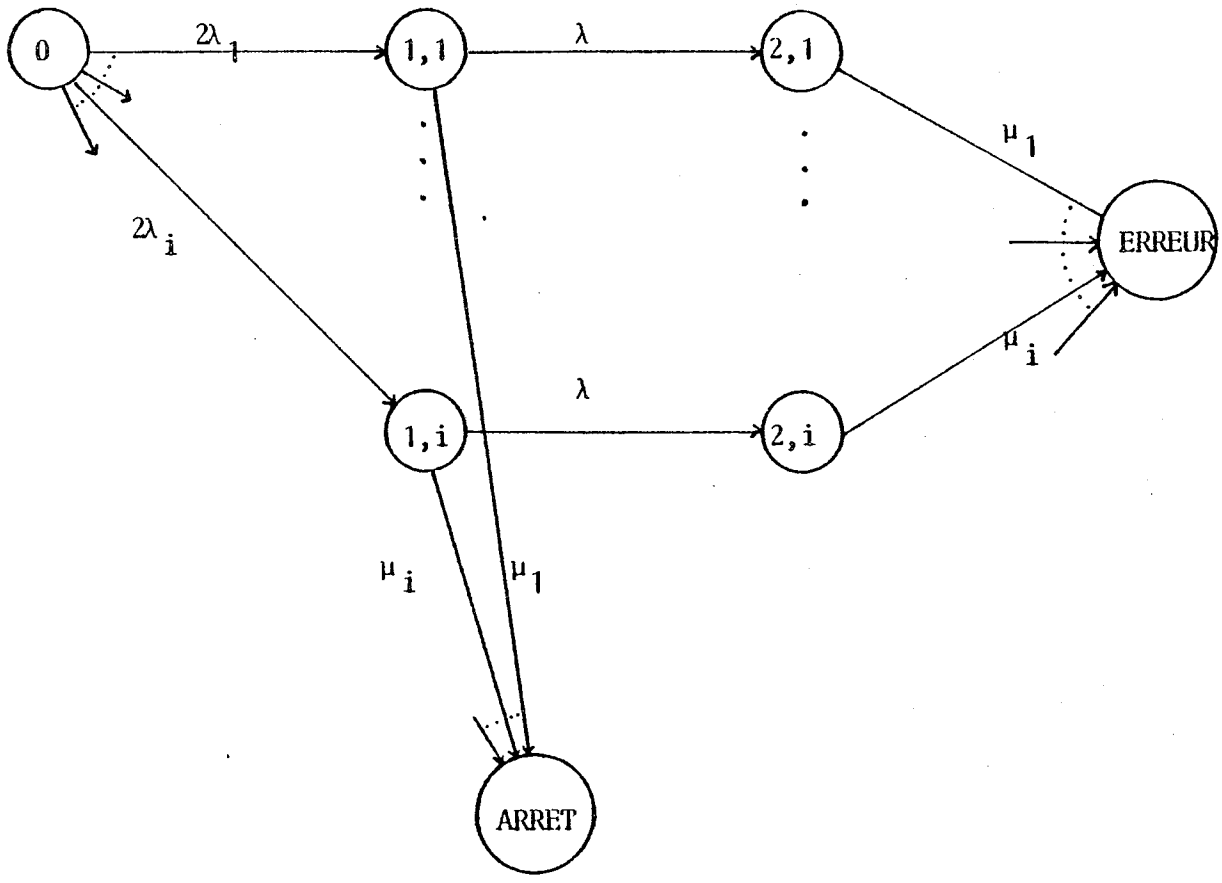


Figure 10: Duplex externe - Contexte A - Modèle à taux d'erreur distribué

Proposition

L'insécurité IS_3 donnée par ce modèle est :

$$IS_3(T) = E_\mu\ (IS_2(T)) = E_\mu\ (2\lambda^2\ F(T,\mu))$$

où E_μ représente l'espérance mathématique par rapport à la variable aléatoire μ : $E_\mu\ (X(\mu)) = \sum_{i=1}^{\infty} p_\mu\ (\mu_i)\ X(\mu_i)$

et où $IS_2(T)$ est l'insécurité donnée par le modèle à taux d'erreur (V.4).

Démonstration

Cette proposition correspond à une technique classique de transformation des processus de Markov. Nous pouvons l'obtenir par le calcul de la façon suivante :

Nous avons :

$$P_{erreur}\ (T) = \int_o^T \sum_{i=1}^{\infty} \mu_i\ P_{2i}\ (t)\ dt = \sum_{i=1}^{\infty} \int_o^T \mu_i\ P_{2i}\ (t)\ dt$$

En définissant $P_{erreuri}(T) = \int_o^T \mu_i\ P_{2i}(t)\ dt$

$$P_{erreur}(T) = \sum_{i=1}^{\infty} P_{erreuri}\ (T)$$

D'autre part; nous avons :

$$\frac{dP_{1i}(t)}{dt} = \lambda_i\ P_o(T) - (\lambda + \mu_i)\ P_{1,i}\ (T)$$

$$P_o\ (T) = e^{-2\lambda T}$$

Définissons $P_{oi}(T) = \frac{\lambda_i}{\lambda} P_o(T)$

Nous avons alors :

$$P'_{1i}(T) = \lambda P_{oi}(T) - (\lambda + \mu_i) P_{1,i}(T)$$

$P_{erreuri}(T)$ est alors donné par le modèle suivant (figure 11)

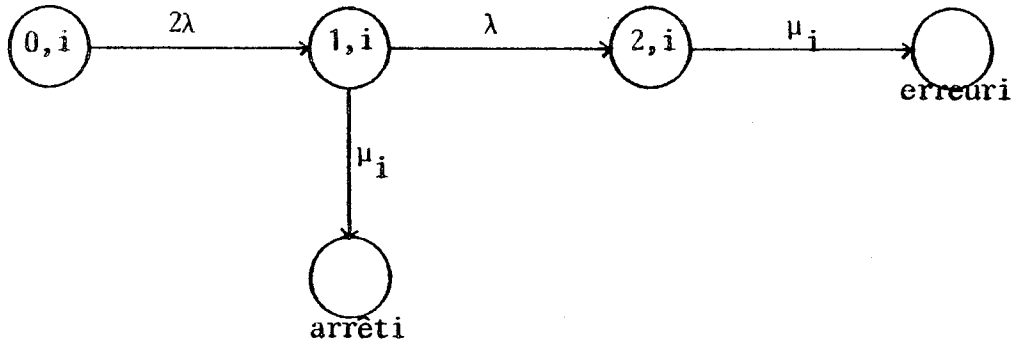


Figure 11 : Modèle donnant $P_{erreur\ i}$

Ce modèle correspond au modèle à taux d'erreur unique (V.4.2.) avec pour état initial : $P_{oi}(0) = \frac{\lambda_i}{\lambda}$ $P_{1i}(0) = P_{2i}(0) = P_{erreur\ i}(0) = 0$

D'où :

$$P_{erreur\ i}(T) = \frac{\lambda_i}{\lambda} 2\lambda^2 F(T,\mu_i)$$

$$P_{erreur}(T) = \sum_{i=1}^{\infty} \frac{\lambda_i}{\lambda} 2\lambda^2 F(T,\mu_i)$$

V.5.2.2. Utilisations du modèle

La distribution p_μ étant inconnue, la question de l'exploitation de ce modèle pour obtenir des évaluations, se pose : trois approches sont possibles :

a) approche de pire cas

Nous avons vu (V.4.2.2.) que $IS_2(T,\mu)$ admet un maximum atteint pour $\mu = \bar{\mu}(T)$.

Nous pouvons alors écrire :

$$E_\mu[IS_2(T,\mu)] \leq IS_2(T,\bar{\mu})$$

Nous retrouvons l'approche de pire cas du modèle à taux d'erreur unique ; nous avons vu qu'elle était relativement inefficace.

b) Hypothèse numérique sur la valeur minimum de μ

La fonction $IS_2(T,\mu)$ est décroissante au-delà de $\bar{\mu}$;

si on peut supposer qu'il existe une valeur de μ minimum, μ_o, telle que $\mu_o \geq \bar{\mu}(T)$ et que :

aucune panne n'a un taux d'erreur inférieur à μ_o (ou encore : $\forall\ \mu \leq \mu_o$, $p_\mu(\mu)=0$), nous pouvons faire la majoration :

$$E_\mu(IS_2(T,\mu)) \leq IS_2(T,\mu_o)$$

Nous pouvons ainsi obtenir des évaluations efficaces.

Cependant, une telle hypothèse est clairement irréaliste puisqu'il est reconnu que, dans des circuits complexes, si beaucoup de pannes se manifestent rapidement, d'autres peuvent avoir une latence extrêmement grande, c'est-à-dire un taux d'erreur très petit.

Remarquons que ces pannes sont sans doute peu nombreuses, mais en toute rigueur, leur probabilité n'est pas nulle, de sorte que l'hypothèse proposée ne s'applique pas.

Nous cherchons donc s'il est possible, par adjonction d'une hypothèse numérique réaliste, d'améliorer l'évaluation très pessimiste obtenue par l'approche de pire cas. Une solution consisterait à pouvoir considérer μ_o, non pas comme le plus petit taux d'erreur sur l'ensemble des pannes, mais comme la moyenne des taux d'erreurs sur l'ensemble des pannes (ce qui peut alors être considéré comme réaliste). Naturellement, cette solution ne serait acceptable que s'il était possible d'opérer ce remplacement sans ajouter d'autres hypothèses optimistes. La troisième approche proposée montre cette possibilité.

c) Hypothèse numérique sur la valeur moyenne de la latence $l = \frac{1}{\mu}$:

Approche par concavité :

Nous allons tirer profit de la propriété de Jensen [KAU77] :

"Soit f une fonction concave de la variable x ; quelle que soit une distribution de probabilité sur x, l'inégalité suivante est vérifiée :

$$E(f(x)) \leq f(E(x))$$

La fonction $IS_2(T,\mu)$ n'étant pas concave sur $[\bar{\mu}(T), +\infty[$ (voir figure 12) cette propriété ne s'applique pas directement.

Le changement de variable :

$$l = \frac{1}{\mu}$$

$$G(T,l) = F(T,\frac{1}{l})$$

donne :

$$E_l(G(T,l)) = E_\mu(F(T,\mu))$$

et $\quad G(T,l) = Tl - 2l^2 + Tle^{-T/l} + 2l^2 e^{-T/l}$

Cette fonction admet un maximum de $\bar{l}(T) = \frac{1}{\bar{\mu}(T)}$ et l'étude de sa dérivée seconde montre qu'elle est concave entre 0 et $\bar{l}(T)$ (voir figure 12).

Cette fonction G peut alors être majorée par la fonction G' définie ainsi :

$$G'(T,1) = \begin{cases} \text{si } 0 \leq 1 \leq \bar{I}(T) \text{ alors } G(T,1) \\ \\ \text{sinon } G(T,\bar{I}(T)) \end{cases}$$

G' est une fonction concave et nous obtenons :

$$E_\mu(F(T,\mu)) = E_1(G(T,1)) \leq E_1(G'(T,1)) \leq G'(T,E_1(1))$$

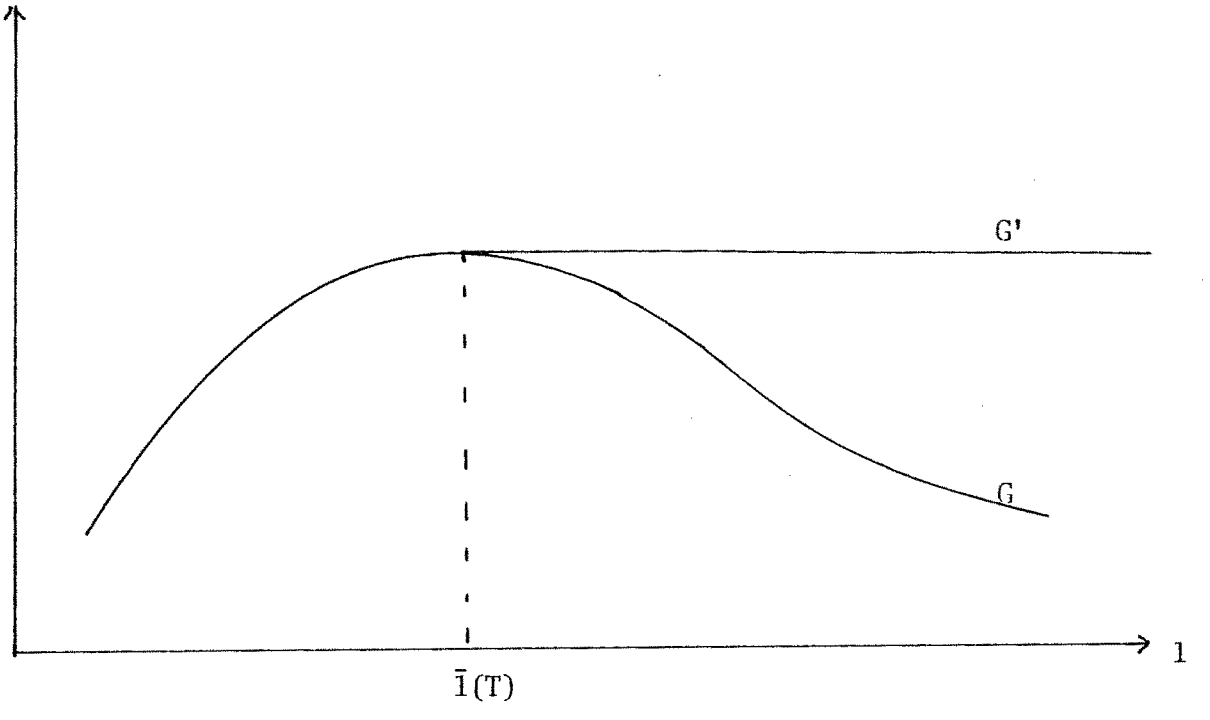


Figure 12 : définition des fonctions G(T,1) et G'(T,1)

Or $E_1(1)$ n'est autre que la latence moyenne $E_L(L)$ sur l'ensemble des pannes pour la loi de latence définie plus haut par l'hypothèse H5 :

$$p_L(L) = \sum_{\mu_i} p_\mu(\mu_i)\ \mu_i\ e^{-\mu_i L}$$

En effet, par définition : $E_1(1) = \sum_{1_i} p_1(1_i) \cdot 1_i = \sum_{\mu_i} p_\mu(\mu_i) \frac{1}{\mu_i}$

et $E_L(L) = \sum_{\mu_i} p_\mu(\mu_i) \int_0^\infty \mu_i\ e^{-\mu_i L}.L..dL = \sum_{\mu_i} p_\mu(\mu_i) \frac{1}{\mu_i}$

Nous résumerons les résultats de cette approche dans la proposition suivante :

Proposition

Sous les hypothèses H_o, H_1, H_2 et H_5, et en supposant que la latence d'erreur moyenne sur l'ensemble des pannes est $L_o \leq \bar{I}(T)$, l'insécurité du duplex externe au temps T est majorée par :

$$2\,\lambda^2\, F(T, \frac{1}{L_o})$$

Si de plus L_o est petit devant T, l'insécurité est majorée par

$$2\,\lambda^2\, T\, L_o$$

(En revanche, si L_o est supérieure à $\bar{I}(T)$, la majoration obtenue est la majoration de pire cas $2\,\lambda^2\, F(T, \frac{1}{\bar{I}(T)})$.

Cette proposition ne modifie pas les résultats obtenus par le modèle à taux d'erreur unique ou par le modèle à taux d'erreur distribué avec hypothèse numérique sur la valeur minimum de μ. En revanche, elle réduit considérablement la portée de l'hypothèse numérique qu'il faut faire pour obtenir une évaluation efficace : il est plus réaliste de supposer que la latence moyenne sur l'ensemble des pannes est petite et en particulier plus petite que $\bar{I}(T)$, que de faire la même hypothèse sur la plus grande des latences moyennes par panne.

D'autre part, cette proposition apporte une réponse à la question posée en conclusion du paragraphe V.4. :

dans le modèle à taux d'erreur unique, nous pouvons prendre comme valeur de ce taux l'inverse de la latence moyenne sur l'ensemble des pannes.

V.5.3. Généralisation aux autres stratégies - Contexte A

L'analyse de la fiabilité des stratégies de redondance d'ordre plus élevé que le duplex se mène de la même façon.

Cependant, ces analyses seront plus complexes : en effet, notre principe pessimiste veut que, lorsqu'il y a dans le système plusieurs pannes latentes, elles se manifestent avec le taux d'erreur de la première de ces pannes, μ "tiré au sort" dans la distribution de μ.
Or ces redondances prévoient aussi des reconfigurations à partir desquelles le système est sain et pourra retomber en panne. Il se peut donc qu'il faille tirer au sort plusieurs valeurs de successivement au cours de la vie du système. Le modèle comporte alors des paramètres μ, μ', μ''... dont il faut calculer la moyenne sur l'ensemble de ces variables aléatoires, indépendantes entre elles et équidistribuées.

Cependant, dans le cas des stratégies pour lesquelles l'existence de deux pannes peut être critique (4H, 5H, 4MRSP), le cas prépondérant dans l'infiabilité s'avère être le cas de ces deux pannes cachées se manifestant simultanément, rendant inutiles les dispositifs de reconfiguration : ce cas correspond au "plus court chemin en λ", c'est-à-dire au terme de plus petit ordre en λ.

Nous pouvons alors montrer que l'analyse au premier ordre conduit à des expressions ne dépendant que d'un seul taux distribué, semblables à celles fournies par le modèle à taux d'erreur unique ; nous obtenons :

$$IR_{4H3}(T) = 6\,\lambda^2 \underset{\mu}{E}\,(F(T,\mu)) + 4(\lambda T)^3$$

$$IR_{5H3}(T) = 6\,\lambda^2 \underset{\mu}{E}\,(F(T,\mu)) + o(\lambda T)^3$$

$$IR_{4MRSP3}(T) = 12\,\lambda^2 \underset{\mu}{E}\,(F(T,\mu)) + 4(\lambda T)^3$$

Les mêmes analyses que pour le duplex s'appliquent. En particulier, nous pouvons remplacer, dans les expressions obtenues par le modèle à taux d'erreur unique, ce taux par l'inverse de la latence moyenne sur l'ensemble des pannes.

En revanche, le cas de la stratégie 5MRSP est plus complexe car plusieurs taux distribués doivent être pris en compte. Nous ne traiterons pas ce cas ici, car nous montrerons dans le paragraphe V.6. comment le modèle à latence distribuée s'y applique plus simplement.

V.5.4. Stratégie duplex - Contexte B

Par une analyse similaire, le taux d'insécurité s'écrit :

$$TIS_3 = E_\mu (TIS_2(\mu))$$

La fonction $TIS_2(\mu) = \frac{2\lambda^2\mu}{\mu^2+3\lambda\mu+\lambda^2}$ a pour maximum atteint pour $\bar{\mu} = \sqrt{2}\,\lambda$, mais n'est pas concave sur $[\bar{\mu}, +\infty[$.

En revanche, $TIS_2(\frac{1}{\bar{I}})$ l'est sur $[0,\bar{I}]$ avec $\bar{I} = \frac{1}{\mu}$

et les mêmes analyses s'appliquent.

En particulier, si on suppose une latence moyenne sur l'ensemble des pannes L_o petite devant $\frac{1}{\lambda}$, nous aurons :

$$TIS_3 \leq 2\lambda^2 L_o$$

V.5.5. Conclusion sur le modèle à taux d'erreur distribué

Le modèle à taux d'erreur distribué permet d'approcher, par analyse concave, des résultats d'évaluation acceptables, sinon en pratique, du moins d'un point de vue théorique : en effet, le problème pratique de l'évaluation de la latence moyenne reste posé. Néanmoins, même du point de vue théorique, deux problèmes subsistent :

d'une part, l'utilisation du modèle est compliquée, dès que plusieurs taux d'erreur distribués apparaissent.

D'autre part, le modèle à taux d'erreur distribué n'est qu'une approximation de la réalité : nous savons qu'il existe des pannes, de type séquentiel, qui n'ont pas un taux d'erreur constant.

Le modèle présenté dans le paragraphe suivant vise à lever cette hypothèse de taux d'erreur constant.

V.6. MODELE A LATENCE DISTRIBUEE

V.6.1. Hypothèse de base

Ce modèle repose sur le principe suivant :
nous supposons que chaque fois qu'une panne apparaît dans une unité à l'instant t, "elle tire au sort" une latence d'erreur l, selon une loi de probabilité L_L inconnue, indépendante de t.
Dans le modèle à taux d'erreur distribué, cette loi était caractérisée par sa densité de probabilité :

$$p_L(l) = \sum_{\mu_i} p_\mu(\mu_i)\, e^{-\mu_i l}$$

Ici, cette loi est quelconque. Les conditions sur l'environnement du système qui permettent de faire cette supposition, peuvent être résumées sous la forme de l'hypothèse suivante H_6 :

H6) Il existe une période d'observation , petite devant T dans le contexte A, et devant $1/\lambda$ dans le contexte B, telle qu'observé périodiquement tous les Δ, le vecteur d'entrée du système apparaisse comme aléatoire et équidistribué, chaque observation étant indépendante des précédentes.

Cette hypothèse signifie que les entrées ainsi observées sont analogues à celle que génèrerait un testeur aléatoire, ce qui permet d'appliquer la théorie de la latence d'erreur [SHE76]. L'adéquation de cette hypothèse avec la réalité de l'environnement du système doit être discutée ; cette discussion sera esquissée en conclusion de ce chapitre.
Pour appliquer l'hypothèse 6 dans le cas de pannes multiples, il convient d'étendre le principe pessimiste établi en V.4.2. pour le modèle à taux d'erreur unique, de la façon suivante :

Principe pessimiste

Lorsqu'une panne apparaît dans une unité à l'instant t et "tire au sort" une latence 1, si une (ou plusieurs) autre(s) panne(s) apparaît dans l'autre (ou les autres) unité(s), entre les instants t et t+1, ces pannes se manifestent de façon cohérente à l'instant t+1.

V.6.2. Application à la stratégie duplex - Contexte A

Pour qu'une situation d'erreur se produise pendant la mission, il faut

- une première panne dans l'une des unités à une date t_1, $0 \leq t_1 \leq T$
- une latence 1 de cette panne telle que : $t_1 + 1 \leq T$ (sinon cette panne ne se manifestera pas au cours de la mission)
- une panne dans la deuxième unité à la date t_2 telle que

$$t_1 \leq t_2 \leq t_1+1$$

Soit u(x) la fonction échelon :

u(x) = si x < 0 alors 0 sinon 1

Pour toute réalisation des trois variables aléatoires t_1, t_2 et 1, la fonction $V(t_1,t_2,1)$ vaut 1 si la mission échoue et 0 sinon :

$(V(t_1,t_2,1) = U(T-t_1-1)\ U(t_2-t_1)\ U(t_1+1-t_2)$

L'insécurité du système s'écrit donc :

$$IS_4(T) = E_{t_1,t_2,1}(V(T,t_1,t_2,1))$$

Par linéarité, nous obtenons :

$$IS_4(T) = E_1(E_{t_1,t_2}(V(T,t_1,t_2,1))) = E_1(W(T,1))$$

avec $$W(T,1) = \int_0^{T-1} 2\ \lambda\ e^{-2\lambda t_1} \left\{ \int^{t_1+1} \lambda e^{-\lambda(t_2-t_1)}\ dt_2 \right\}\ dt_1 . U(T-1)$$

$$W(T,1) = (1 - e^{-2\lambda(T,1)})\ (1 - e^{-\lambda 1})\ U(T-1)$$

Pour T petit devant $1/\lambda$ nous obtenons :

$$W(T,1) = 2\lambda^2\ 1(T-1)\ U(T-1)$$

Remarquons que la fonction $Z(T,1) = 1(T-1)$ est concave (figure 13)

Elle admet un maximum unique en $\bar{I}(T) = \frac{T}{2}$

et $Z(T,\bar{I}(T)) = \frac{T^2}{4}$

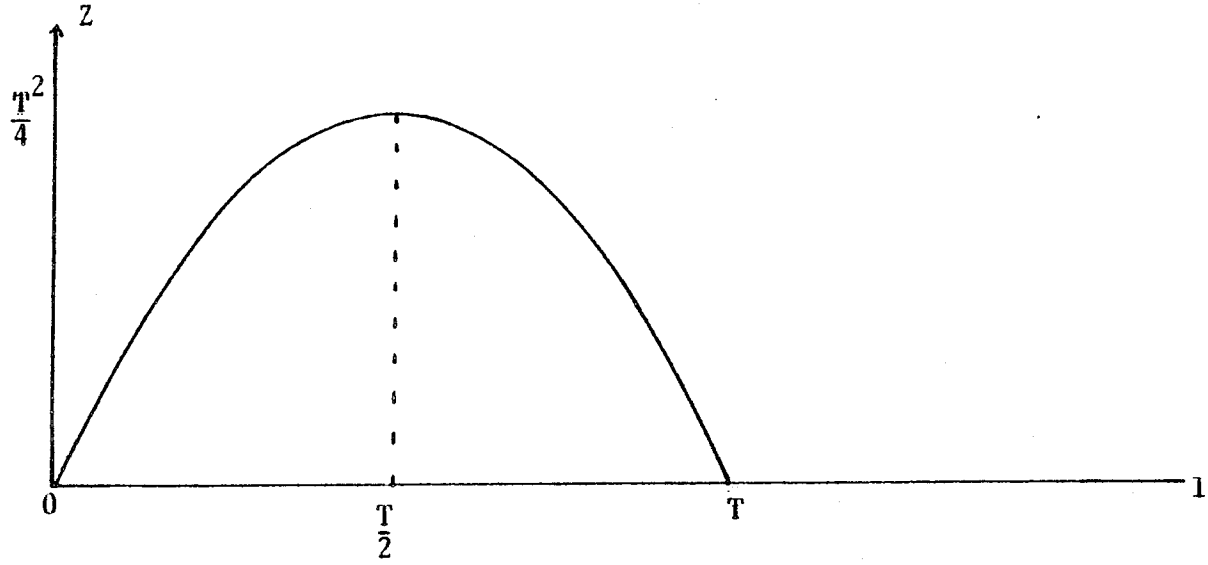


Figure 13

Deux approches sont évidemment possibles :

- approche de pire cas

Elle consiste à prendre :

$$IS_4(T) \leq 2\ \lambda^2\ \frac{T^2}{4} = \frac{\lambda^2 T^2}{2}$$

Remarquons que cette évaluation est moins favorable que le pire cas des modèles précédents. Cela tient à ce que l'hypothèse de latence distribuée est plus faible que celles de taux d'erreur unique ou distribué.

. Hypothèse numérique sur la latence moyenne

Nous pouvons écrire :

$$E_1(W(T,1)) = E_1(Z(T,1)/1 \leq T).Prob(1 \leq T)$$

(le signe / est pris au sens des probabilités conditionnelles).

Le terme Prob(1 ≤ T) n'est pas connu mais peut être majoré par 1 puisqu'il s'agit d'une probabilité.

La fonction Z étant concave, nous avons :

$$E_1(Z(T,1)/1 \leq T) \leq Z(T,E(1/1 \leq T))$$

Si nous faisons l'hypothèse que la moyenne des latences, considérée sur l'ensemble des latences inférieures à T, vaut 1_o, nous obtenons :

$$IS_4(T) = 2\ \lambda^2\ Z(T,1_o)$$

Si de plus 1_o est petit devant T :

$$IS_4(T) = 2\ \lambda^2\ T\ 1_o\ .$$

Nous retrouvons le résultat énoncé en V.5.2.2. , approche c ; cependant, ce résultat est obtenu à partir de l'hypothèse 6 qui est plus faible que l'hypothèse 5. De plus ici, 1_o n'est pas la latence moyenne en général, mais la latence moyenne sur l'ensemble des latences inférieures à T : c'est un facteur qui pourrait s'avérer avantageux si l'on cherchait à estimer 1_o par une expérience statistique : chaque essai de panne aurait alors une durée inférieure à T.

V.6.3. Stratégie 4MRSP

Si nous analysons comment une situation d'échec peut se produire, nous distinguons deux possibilités :

a) une panne arrive dans une unité à une date t_1 $\quad 0 \leq t_1 \leq T$

. cette panne se manifeste avec une latence 1 telle que $t_1 + 1 \leq T$

. une panne arrive dans une deuxième unité à une date t_2 telle que

$t_1 \leq t_2 \leq t_1 + 1$

. les deux pannes se manifestent de manière cohérente (principe pessimiste) à l'instant $t_1 + 1$

b) une première panne arrive et se manifeste, provoquant la reconfiguration en TMR

. le TMR échoue par le processus décrit en A

Nous avons vu que le terme correspondant au cas b) peut être majoré par $4(\lambda T)^3$ (voir V.4.3.).

Le terme correspondant au cas a) est semblable à celui trouvé pour le duplex, à un coefficient près (choix d'une unité parmi 4, puis choix d'une unité parmi 3) : $12\ \lambda^2.1.(T-1).U(T-1)$

d'où $IR_{4MRSP4} = 6 \underset{1}{E} (W(T,1)) + 4(\lambda T)^3$

Les mêmes analyses que pour le duplex s'appliquent donc.

V.6.4. Stratégies 4H et 5H

De même que ci-dessus, l'analyse des situations d'échec permet d'écrire

$$IR_{4H4} = 3 \underset{1}{E} (W(T,1)) + 4(\lambda T)^3$$

$$IR_{5H4} = 3 \underset{1}{E} (W(T,1)) + 0(\lambda T)^3$$

Là encore, les mêmes analyses que pour le duplex s'appliquent.

V.6.5. Analyse de la stratégie 5MRSP

Cette analyse, qui était complexe en utilisant le modèle à taux d'erreur distribué, s'avère dans ce modèle, relativement simple.

De même que pour la stratégie 4MRSP, nous nous bornerons à évaluer la probabilité que trois pannes fassent échouer la mission : les autres cas sont pris en compte dans un terme de type $5(\lambda T)^4$.

Introduisons les dates des trois premières pannes :

t_1 de densité de probabilité $5\ \lambda\ e^{-5\lambda t_1}$

t_2 de densité de probabilité $4\ \lambda\ e^{-4\lambda(t_2-t_1)}$, $(t_2 \geq t_1)$

t_3 de densité de probabilité $3\ \lambda\ e^{-3\lambda(t_3-t_2)}$, $(t_3 \geq t_2)$

Deux cas seulement de triple panne conduisant à l'échec peuvent se produire :

a) Le système reste en 5MR et la mission échoue : il y a eu 3 pannes latentes qui se sont manifestées ensemble ; ce cas correspond à :

$$t_1 + l_1 \leq T \quad ; \quad t_2 \leq t_3 \leq t_1 + l_1$$

b) Le système se reconfigure en 4MR après une panne manifestée. Deux pannes latentes se manifestent simultanément, font ensuite échouer le 4MR ; ce cas correspond à $t_1 + l_1 < t_3 \qquad t_2 + l_2 \leq T \qquad t_3 \leq t_2 + l_2$

l_1 et l_2 étant deux latences indépendantes équidistribuées.

Il vient :

$$IR_{5MRSP4} = \underset{l_1,l_2}{E}\ (V'(T,l_1,l_2)) + 5(\lambda T)^4$$

où

- $V'(T,l_1,l_2) = W_1(T,l_1,l_2) + W_2(T,l_1,l_2)$

- W_1 correspond au premier cas, W_2 au second

$$- W_1(T,l_1,l_2) = \int_0^{T-l_1} 5\lambda e^{-5\lambda t_1} \int_{t_1}^{t_1+l_1} 4\lambda e^{-4\lambda(t_2-t_1)} \int_{t_2}^{t_1+l_1} 3\lambda e^{-3\lambda(t_3-t_2)}\, dt_3\, dt_2\, dt_1\, U(T-l)$$

$$- W_2(T,l_1,l_2) = \int_0^{T-l_1} 5\lambda e^{-\lambda t_1} \int_{t_1}^{T-l_2} 4\lambda e^{-4\lambda(t_2-t_1)} \int_{Max(t_2,t_1+l_1)}^{t_2+l_2} 3\lambda e^{-3\lambda(t_3-t_2)}\, dt_3\, dt_2\, dt_1\, U(T-l_3)\, T(T-l_2)$$

Pour λT petit devant 1, nous obtenons :

$$W_1(T,l_1,l_2) = 60\,\lambda^3\,\frac{l_1^2}{2}\,(T-l_1)\,U(T-l_1)$$

Nous simplifions le calcul de W_2 en supprimant la contrainte $t_3 > t_1+l_1$, ce qui constitue bien une majoration ; il vient alors :

$$W_2(T,l_1,l_2) = 60\,\lambda^3\,\frac{l_2}{2}\,(T-l_2)^2\,U(T-l_2)$$

En employant la majoration usuelle $Prob(l \le T) = 1$ nous obtenons

$$\underset{l_1 l_2}{E}\,(V'(T,l_1,l_2)) = 60\,\lambda^3\,\{\underset{l}{E}\,[\frac{l^2}{2}\,.\,(T-l)/_{l\le T}] + \underset{l}{E}\,[\frac{l}{2}\,.\,(T-l)^2/_{l\le T}]\}$$

Soit :

$$\underset{l_1 l_2}{E}\,[V'(T,l_1,l_2)] = 60\,\lambda^3\,\underset{l}{E}\,[\frac{lT}{2}\,(T-l)/_{l\le T}]$$

d'où :

$$IR_{5MPSP4}(T) = 60\,\lambda^3\,\underset{l}{E}\,[\frac{lT}{2}\,(T-l)/_{l\le T}] + 5(\lambda T)^4$$

- L'approche de pire cas peut être appliquée :

le pire cas est obtenu pour $l = \frac{T}{2}$ et donne :

$$IR_{5MRSP4}\,(T) = \frac{60}{8}\,\lambda^3\,T^3 + 5\,\lambda^4\,T^4 = \frac{15}{2}\,\lambda^3\,T^3 + 5\,\lambda^4\,T^4$$

Ce résultat est plus défavorable que le pire cas obtenu en V.4.6. par le modèle à taux d'erreur unique ($0{,}05316 \times 60\,\lambda^3\,T^3$).

- L'approche "par concavité" est aussi applicable :

en effet, la fonction $Z(T,1) = \frac{1T(T-1)}{2}$ est concave en 1

Nous pouvons alors appliquer la majoration :

$$IR_{5MRSP4}(T) \leq 60\ \lambda^3\ Z(t, \underset{1}{E}(1/_{1\leq T}) + 5\ \lambda^4\ T^4$$

Si nous notons 1_o la latence moyenne sur l'ensemble des latences inférieures à T : $1_o = \underset{1}{E}\ (1/_{1\leq T})$

$$IR_{5MRSP4}\ (T) = 30\ \lambda^3\ 1_o\ T(T-1_o) + 5\ \lambda^4\ T^4$$

Si de plus nous pouvons faire l'hypothèse que 1_o est petit devant T, il vient :

$$IR_{5MRSP4}\ (T) \simeq 30\ \lambda^3\ T^2\ 1_o + 5\ \lambda^4\ T^4$$

Cette formule est analogue à la formule obtenue en V.4.6. :

$$30\ \frac{\lambda^3 T^2}{\mu_o} + 5(\lambda T)^4$$

V.6.6. Stratégie - duplex - Contexte B

Nous pouvons construire le modèle semi-markovien suivant (figure 14)

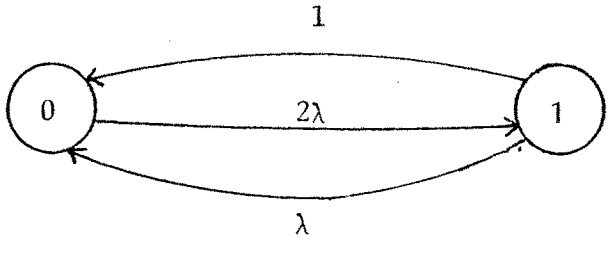


Figure 14 : Modèle stratégie - duplex - contexte B

Ce modèle est semi-markovien [LAP75] car 1 n'est pas un taux constant, mais un temps fixe. Ce modèle décrit les évolutions suivantes :

après une première panne (transition 0 1), soit cette panne se manifeste au bout du temps 1 et il y a arrêt et réparation, soit une deuxième panne se produit avant 1, ce qui conduit à une situation d'insécurité.

Si T(1) est le taux des évènements correspondant à la transition (1,0) étiquetée par en régime stationnaire, le taux d'insécurité du duplex s'écrit :

$$TIS_4 = \underset{1}{E}\,(T(1))$$

Comme le taux de la transition (0,1) en régime stationnaire est $2\lambda P_0(\infty)$, nous avons :

$$T(1) = 2\lambda.P_0(\infty) \cdot \int_0^1 \lambda\, e^{-\lambda t}\, dt = 2.\lambda.(1-e^{-\lambda 1}).P_0(\infty)$$

En majorant par $P_0(\infty) \leq 1$:

$$T(1) = 2\,\lambda\,(1 - e^{-\lambda 1})$$

Remarquons tout d'abord que pour l'approche de pire cas, la majoration $P_0(\infty) = 1$ est trop forte ; une étude plus précise s'imposerait pour appliquer cette approche.

La fonction T(1) étant concave en 1, en notant $1_o = \underset{1}{E}(1)$, nous pouvons majorer TIS_4 par :

$$TIS_4 = T(\underset{1}{E}\,(1)) = 2\,\lambda\,(1 - e^{-\lambda 1_o})$$

Si de plus nous pouvons garantir une valeur de 1_o faible devant $1/\lambda$, nous obtenons :

$$TIS_4 = 2\,\lambda^2\,1_o$$

Cette formule est analogue à celles obtenues en V.4.8. et V.5.4. par les modèles à taux d'erreur unique et à taux d'erreur distribué.

V.6.7. Conclusion sur le modèle à latence distribué

Il apparaît que le passage de l'hypothèse de taux d'erreur distribués à celle plus faible de latences d'erreur distribuées permet de conserver les résultats obtenus en V.5. :

- les pires cas sont légèrement plus défavorables : ceci tient au caractère plus faible de l'hypothèse,
- les calculs moyens obtenus par concavité sont identiques,
- les calculs sont moins systématiques car les modèles construits ne sont pas Markoviens. Cependant, ces calculs s'avèrent parfois plus simples : dans le contexte A, nous obtenons des expressions polynomiales en fonction des latences inconnues, ce qui facilite la recherche des pires cas ; ainsi, il a été possible d'obtenir des expressions analytiques de ces pires cas ; de même, le modèle de la stratégie 5MRSP a pu être traité simplement, alors qu'il était difficile à traiter avec l'hypothèse de taux d'erreur distribués.

V.7. CONCLUSION : DISCUSSION SUR LES MODELES ETUDIES

Ce chapitre nous a permis d'introduire une approche de l'évaluation "rigoureuse" de sécurité et de fiabilité des stratégies à décision externe. Outre deux hypothèses de base, différentes hypothèses ont été proposées :

. l'hypothèse H3 (manifestation immédiate des pannes) donne les résultats classiques trop optimistes.

. la suppression de toute hypothèse sur la latence des pannes étant trop pessimiste, nous avons envisagé successivement :

l'hypothèse H4 : une latence distribuée selon une loi à taux d'erreur unique,

l'hypothèse H5 : une latence distribuée selon une loi inconnue,

l'hypothèse H6 : une latence distribuée selon une loi inconnue.

Ces trois hypothèses peuvent être traitées par pire cas ou par hypothèse numérique portant respectivement sur le taux d'erreur, l'inverse de la latence moyenne et la latence moyenne, l'étude s'appuyant dans ces deux derniers cas sur une propriété de concavité des fonctions d'évaluation. Ces trois hypothèses donnant des résultats semblables, il est logique de ne retenir que la dernière, qui est la plus faible et qui aura donc le plus de chances d'apparaître réaliste aux autorités de certification.

Cependant, il faut remarquer que ces hypothèses ont en commun de considérer une latence d'erreur à distribution indépendante du temps : l'environnement doit être, à une homothétie temporelle près, analogue à celui que délivrerait un testeur aléatoire. Ce cas de figure est assez rare :

par exemple, le programme d'atterrissage d'un pilote automatique d'avion n'est sollicité qu'une fois par vol ; de même un système de commande d'aiguillage de chemins de fer rencontre des configurations périodiques de circulation des trains de période 24 heures (ou plus longue, certaines voies d'une gare peuvent n'être utilisées que très rarement). Une solution pour pallier à cette difficulté, consiste à adjoindre au système des tests en lignes dont le rôle est d'émuler les configurations d'environnement se produisant naturellement trop rarement et dont les résultats sont votés à l'extérieur comme des sorties ordinaires : dans ce cas, il convient de démontrer que ces tests sont bien susceptibles d'avoir l'effet désiré, c'est-à-dire d'émuler un environnement aléatoire.

L'étude présentée conduit à isoler trois solutions offertes au concepteur pour démontrer que son système a les propriétés de fiabilité ou de sécurité requises :

- utiliser le modèle pessimiste,

- si le résultat n'est pas satisfaisant, traiter par pire cas le modèle à latence distribuée ; il faut alors "démontrer" l'indépendance de cette latence par rapport au temps.
- si le résultat n'est encore pas satisfaisant (ce qui est probable, l'approche de pire cas n'améliorant que très peu les résultats du modèle pessimiste), proposer une hypothèse numérique sur la latence moyenne ; il faut alors montrer théoriquement ou par des expériences appropriées que la valeur proposée est réaliste.

Cependant, l'approche "rigoureuse" de l'évaluation a été appliquée ici à des systèmes très simples (stratégies à décision externe) ; s'il est évident que le problème de la validation ne peut être traité correctement que sur des systèmes bien structurés (cf. chapitre II), permettant de partitionner le problème de l'évaluation du système en plusieurs évaluations de sous-systèmes simples, il n'en demeure pas moins que bon nombre de stratégies s'avèrent plus complexes que celles étudiées dans ce chapitre : ces stratégies concernaient des unités dont la seule fonction est d'émettre des résultats vers un autre sous-système chargé du vote ; l'analyse de l'effet des pannes se complique lorsque les unités étudiées sont chargées d'effectuer aussi des votes, des reconfigurations, ou d'émettre des résultats vers plusieurs sous-systèmes différents...

Les deux chapitres suivants ont pour objet de montrer comment, sur deux exemples de complexité croissante, l'approche "rigoureuse" peut être appliquée à des systèmes complexes.

C H A P I T R E V I

EVALUATION D'UN SYSTEME DUPLEX

A IMPLANTATION INTERNE

VI.1. PRESENTATION DU PROBLEME

Considérons le système duplex à implantation logicielle interne représenté dans la figure 1 :

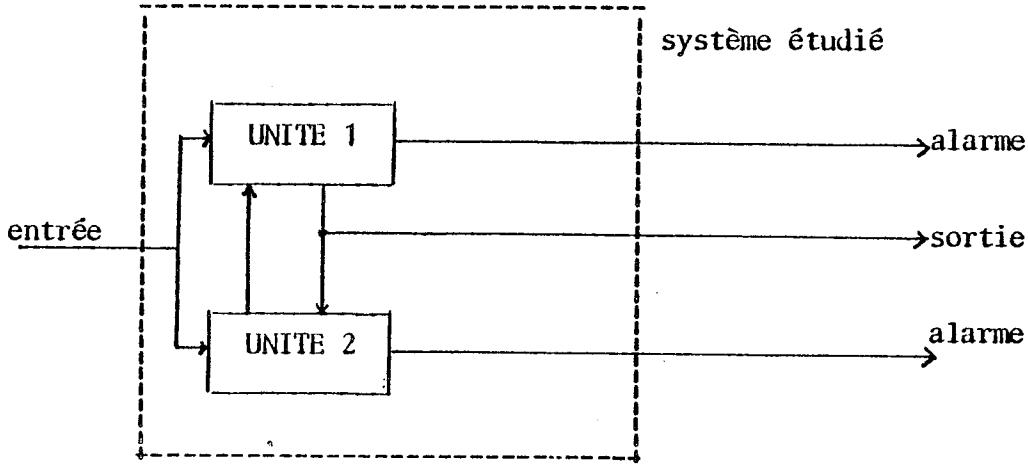


Figure : duplex à implantation interne

Le système est constitué de deux calculateurs 1 et 2 qui reçoivent les mêmes entrées et calculent la même fonction. Ils échangent périodiquement leurs résultats mais seuls les résultats de l'unité 1 sont émis vers le monde extérieur. De plus, chaque calculateur dispose d'une sortie "alarme" qu'il met à '1' s'il constate un désaccord entre son résultat et le résultat reçu de son partenaire. Le monde externe ne prend en compte la sortie que si aucune alarme n'est émise. Ainsi, nous pouvons définir la réussite de la mission ainsi : le système est sûr si, ou la sortie est correcte, ou une alarme est émise. La sécurité de mission T sera la probabilité que le système reste sûr de l'instant 0 à l'instant T.

Il est clair que le fait de confier aux unités le rôle de détection, diversifie les cas d'échec de la mission par rapport au duplex externe déjà étudié : cette diversification rend plus délicate la construction du modèle d'évaluation.

VI.2. CONSTRUCTION DU MODELE

VI.2.1. Approches possibles

Il s'agit de décrire les réactions du système (émission des résultats, émission des alarmes) aux évènements de pannes ; pour résoudre ce problème, deux approches sont possibles :

.l'approche descendante ("top-down") dite d'arbre de défaillance ("fault tree") consiste à essayer de construire, à partir des situations d'échec, l'arbre des causes de cet échec jusqu'à atteindre toutes les pannes élémentaires pouvant mener à cet échec ; cette approche semble peu utilisée en informatique. Notons la tentative de J. LOSQ [LOSQ78] pour l'appliquer de façon systématique à un calculateur tolérant les pannes.

.l'approche montante ("bottom up") dite F.M.E.A. (Failure Modes and Effects Analysis) consiste à recenser les modes de pannes des composants du système et à dériver leurs effets sur le comportement du système ; elle est couramment utilisée pour les analyses de fiabilité de systèmes électriques, mécaniques ou à base d'électronique peu intégrée, qui comportent peu de modes de pannes de sorte que leurs effets peuvent être facilement dérivés par une analyse simple ou une simulation. Elle peut aussi s'appliquer à des systèmes plus complexes, à condition que leur comportement externe puisse se résumer en l'expression d'un petit nombre de fonctions et que la stratégie de tolérance aux pannes s'exprime en termes de correctes ou incorrectes exécutions de ces fonctions. L'évaluation de COPRA est basée sur une telle approche (BRO77). Dans la suite, nous utiliserons cette approche ; plus particulièrement, la construction du modèle sera réalisée après deux étapes :

. description des fonctions des unités et recensement de leurs modes de pannes,

. affectation, pour chaque mode de panne, d'un taux de pannes et d'une description d'un processus de manifestation de la panne considérée ; ces descriptions seront basées, dans ce chapitre, sur le modèle à latence distribuée développé en fin de chapitre précédent. Une évaluation du duplex à implantation interne, basée sur le modèle à taux d'erreur unique peut être trouvée dans [CAS81b]

VI.2.2. Description des modes de pannes

Chaque unité a deux fonctions : calcul des résultats et émission de l'alarme. La fonction "calcul" peut prendre deux états, représentés par la variable E_c :

$E_c = 0$: aucune panne n'altère la fonction

$E_c = 1$: une panne fait délivrer un résultat faux pour certaines configurations de l'état interne et des entrées.

La fonction "alarme" a deux modes de pannes possibles : "pas d'alarme" et "fausse alarme", décrits par les variables E_{na} et E_{fa} :

$E_{na} = 0$: une alarme est émise quand cela est nécessaire (mais elle peut aussi être émise quand cela n'est pas nécessaire).

$E_{na} = 1$: dans certains cas, alors que l'alarme devrait être émise, elle ne l'est pas.

$E_{fa} = 0$: aucune fausse alarme n'est émise.

$E_{fa} = 1$: dans certains états de l'unité, une fausse alarme est émise.

Ainsi, l'état d'un calculateur peut être décrit par un triplet (E_c, E_{na}, E_{fa}) qui peut prendre 8 valeurs.

En faisant l'hypothèse que une panne ne "répare" pas, c'est-à-dire qu'un évènement de panne ne peut pas faire passer une variable représentant un mode de panne de 1 à 0, l'évolution entre les états de pannes permet de définir un treillis booléen sur l'ensemble des 8 états (figure 2).

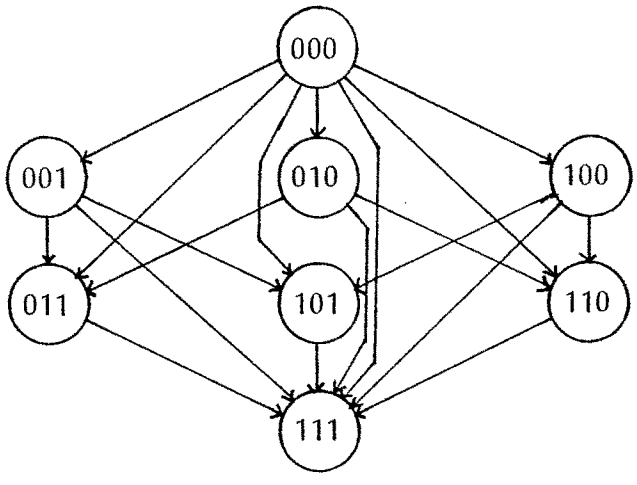


Figure 2 : Etats de panne d'une unité

La lecture de cette figure permet de mesurer la complexité de la modélisation : d'une part, chaque unité comptant 8 états, le duplex comportera 64 états plus les deux états puits arrêt et alarme, atteints après manifestation des pannes ; d'autre part, il convient d'associer un taux à chaque transition entre états de pannes ; nous pouvons dénombrer 19 transitions dans le treillis de la figure 2 : les 19 taux sont a priori inconnus et même si certaines relations peuvent les lier (exemple : le taux de panne d'une unité est égal à la somme des taux des transitions partant de l'état 000), l'analyse devra être faite sur de nombreux paramètres inconnus ; de même, il faut affecter des latences à chaque mode de pannes et introduire ainsi d'autres paramètres inconnus.
Ces remarques nous conduisent à chercher à simplifier le modèle ; pour rester fidèle à notre démarche, toute simplification ne sera faite qu'à partir d'une analyse de pire cas ou en posant une hypothèse dûment explicitée et justifiée.

VI.2.3. Construction du modèle simplifié

La première simplification consiste à supprimer l'état de fausse alarme : puisque toute manifestation d'une panne de type "fausse alarme" conduit à l'arrêt, c'est bien une analyse pessimiste que de considérer que ces pannes n'arrivent jamais : ainsi, le taux de pannes global d'une unité n'est réparti que sur des évènements qui eux sont dangereux. L'ensemble des états de pannes d'une unité se réduit alors à quatre états représentés par le doublet (E_c, E_{na}) (figure 3).

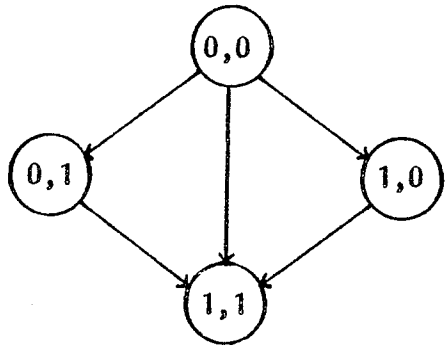
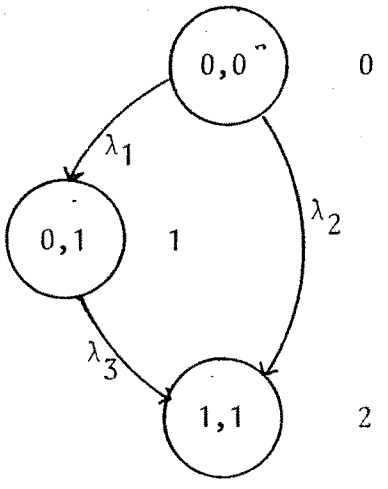


Figure 3 : première réduction du graphe d'état de pannes d'une unité.

Considérons maintenant les états (1,0) et (1,1) : dans les deux cas, l'unité est susceptible d'émettre un mauvais résultat ; mais dans l'état (1,0), elle reste capable de donner l'alarme : cet état a moins de chance de conduire à l'échec que l'état (1,1) (l'échec ne peut se produire que sur double panne concordante) : de manière pessimiste, nous supprimons cet état (1,0) en l'assimilant à l'état (1,1). Nous obtenons alors le graphe de la figure 4 dans lequel nous renommons les états (0,0), (0,1) et (1,1) par 0, 1, 2.



Etat 0 : "unité saine"

Etat 1 : "pas d'alarme à tort"

Etat 2 : "pas d'alarme à tort et mauvais calcul"

Figure 4 : graphe réduit d'état de pannes d'une unité.

Il convient maintenant de préciser la définition des taux de transition dans ce graphe.

Le taux de pannes global d'une unité, qui est un paramètre connu λ, concerne l'ensemble des pannes pouvant arriver dans une unité saine (état 0) ; cet ensemble des pannes est réparti entre les modes correspondant aux états 1 et 2 avec les taux λ_1 et λ_2 ; nous pouvons écrire :

$$\lambda_1 + \lambda_2 = \lambda$$

Le rapport entre ces paramètres est a priori inconnu. Le paramètre α tel que $\lambda_2 = \alpha\lambda$ et $\lambda_1 = (1 - \alpha)\lambda$ est choisi comme paramètre représentant cette inconnue.

Pour définir λ_3, nous faisons l'hypothèse que le processus d'occurrence d'une nouvelle panne dans une unité déjà en panne est du même type que le processus d'occurrence de panne dans une unité saine de sorte que nous pouvons majorer

λ_3 par λ ; remarquons que, si λ_2 n'est pas trop petit, le paramètre λ_3 intervient peu dans l'analyse (il conduit l'unité dans le même état que par la transition λ_2 mais après deux pannes) et qu'on peut dans ce cas se contenter de l'évaluer grossièrement.

Ainsi, la stratégie peut être représentée par le graphe de la figure 5 dans lequel les états sont constitués par les doublets des états des unités 1 et 2, et où les arcs indicés par "m" représentent la manifestation des pannes (les latences tirées au sort seront précisées lors du calcul du modèle). Remarquons qu'il n'y a évidemment pas symétrie par rapport aux unités puisque seule l'unité 1 émet le résultat et peut ainsi commettre une erreur. Remarquons enfin que les pannes de type 1 ne se manifestent que lors de la manifestation d'une panne de type 2 (c'est lorsqu'on a besoin de l'alarme qu'elle peut s'avérer défaillante).

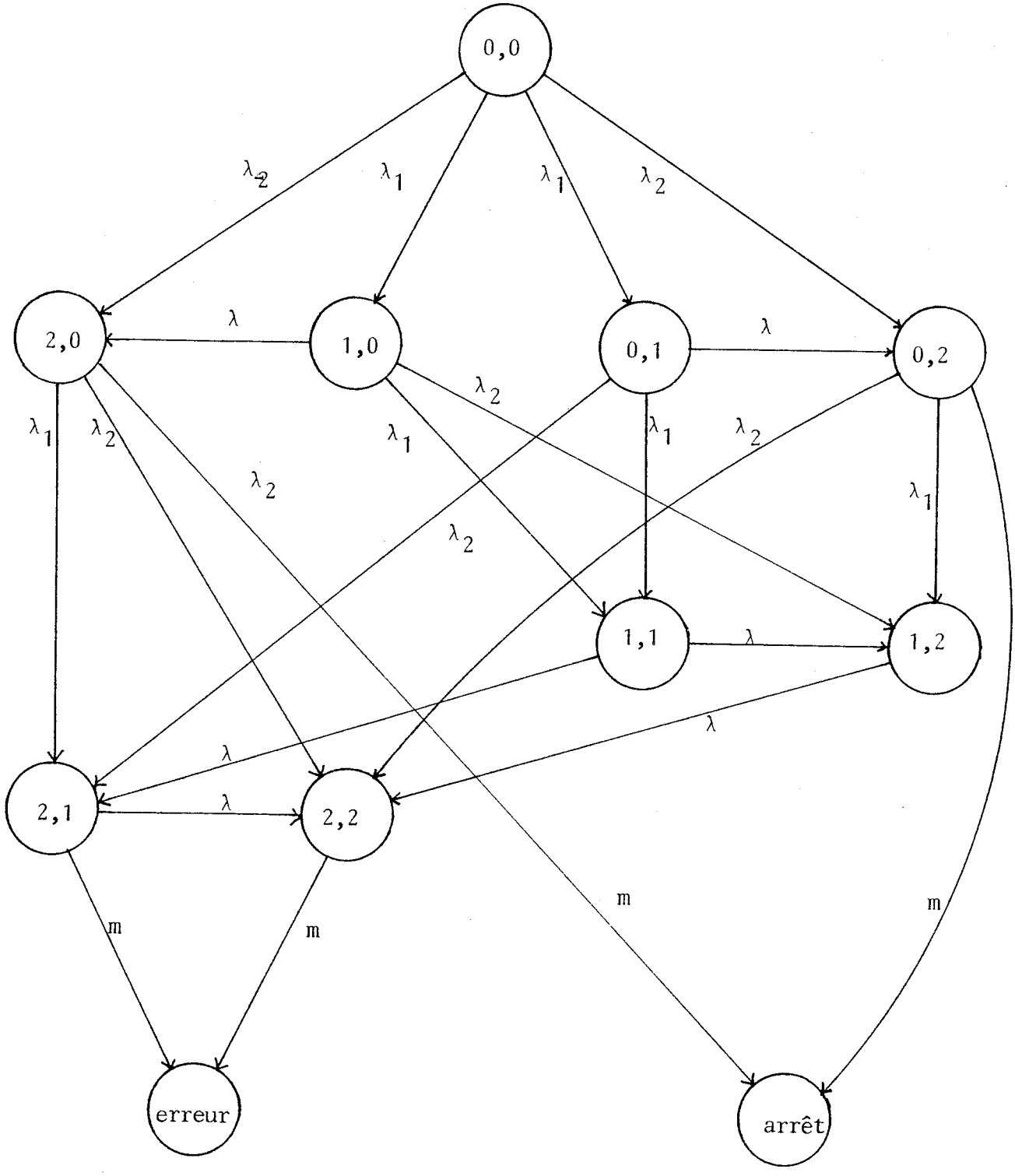


Figure 5 : Modèle de la stratégie duplex interne

VI.3. CALCUL DU MODELE

Parmi les chemins, dans le graphe de la figure , qui conduisent à l'état d'erreur, nous nous intéresserons aux plus courts qui vont constituer le facteur représentatif de l'insécurité (les chemins plus longs constituant des termes d'ordre supérieur en λ, négligeables devant les premiers).
Nous dénombrons 4 chemins de "longueur 2" :

1) $(0,0) \xrightarrow{\lambda_2} (2,0) \xrightarrow{\lambda_1} (2,1) \xrightarrow{m}$ erreur

2) $(0,0) \xrightarrow{\lambda_2} (2,0) \xrightarrow{\lambda_2} (2,2) \xrightarrow{m}$ erreur

3) $(0,0) \xrightarrow{\lambda_1} (0,1) \xrightarrow{\lambda_2} (2,1) \xrightarrow{m}$ erreur

4) $(0,0) \xrightarrow{\lambda_2} (0,2) \xrightarrow{\lambda_2} (2,2) \xrightarrow{m}$ erreur

Ces chemins correspondent aux cas suivants :

1) panne de type 2 dans l'unité 1 à t_1, tirant au sort une latence 1 et panne de type 1 dans l'unité 2 à t_2, avec :

   $t_1 \leq t_2 \leq t_1+1 \leq T$

2) panne de type 2 dans l'unité 1 à t_1, tirant au sort une latence 1 et panne de type 2 dans l'unité 2 à t_2, avec :

   $t_1 \leq t_2 \leq t_1+1 \leq T$

3) panne de type 1 dans l'unité 2 à t_1 et panne de type 2 dans l'unité 1 à t_2 tirant au sort la latence 1 avec :

   $t_1 \leq t_2$ et $t_2 + 1 \leq T$

4) panne de type 2 dans l'unité 2 à t_1 tirant au sort la latence 1' et panne de type 2 dans l'unité 1 à t_2 tirant au sort la latence 1 avec

$$t_1 \leq t_2 \leq t_1+1' \quad \text{et} \quad t_2 + 1 \leq T$$

(il importe peu que t_1+1' soit inférieur à T puisqu'une erreur de calcul de l'unité 2 n'apparaît pas en sortie).

Pour λT petit devant 1, les termes du calcul correspondant à ces chemins sont donnés par :

1) $$A_1 = \int_o^{T-1} \lambda_2 \int_{t_1}^{t_1+1} \lambda_1 \; dt_2.dt_1.\; U(T-1)$$

2) $$A_2 = \int_o^{T-1} \lambda_2 \int_{t_1}^{t_1+1} \lambda_2 \; dt_2.dt_1.\; U(T-1)$$

3) $$A_3 = \int_o^{T-1} \lambda_1 \int_{t_1}^{T-1} \lambda_2 \, dt_2 \, dt_1 \;\; U(T-1)$$

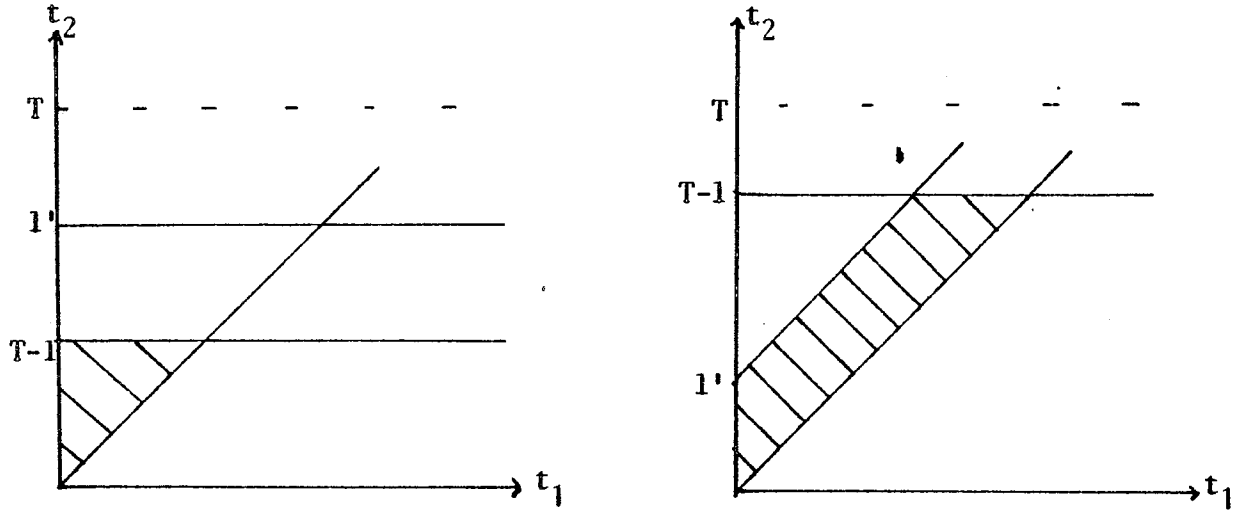
4) $$A_4 = \int_o^{T} \lambda_2 \int_{t_1}^{\min(t_1+1',T-1)} \lambda_2 \,.\, dt_2 \,.\, dt_1 \;\; U(T-1)$$

Le calcul des trois premiers termes donne :

$$A_1 + A_2 = \lambda_2.\lambda.1.(T-1).U(T-1)$$

$$A_3 = \lambda_2.\lambda_1.\frac{(T-1)^2}{2} \;.\; U(T-1)$$

Le terme $A_4\ \lambda_2^2$ représente l'aire hachurée dans l'une des figures 6a et 6b.



a : cas $1' \geq T-1$ b : cas $1' \leq T-1$

Figure 6 : représentation du terme $\lambda_2^2\ A_4$

d'où l'expression de ce terme :

si $T-1 \leq 1'$ $\quad A_4 = \lambda_2^2 \cdot \frac{(T-1)^2}{2} \cdot U(T-1)$

si $1' \leq T-1$ $\quad A_4 = \lambda_2^2 \cdot [\frac{(T-1)^2}{2} - \frac{(T-1-1')^2}{2}] = \lambda_2^2\ 1' \cdot \frac{(2T - 21 - 1')}{2}$

Essayons de majorer A_4 :

1° majoration

Une première majoration est obtenue en supposant $1' \geq T$ (nous considérons de manière pessimiste que la panne de l'unité 2 ne se manifeste pas) : nous sommes ainsi dans le premier cas ($T-1 \leq 1'$) :

$$A_4 = \lambda_2^2\ \frac{(T-1)^2}{2}\ U(T-1)$$

Et la somme des quatre termes A_1, A_2, A_3, A_4 donne :

$$S = \lambda_2.\lambda. \left[1.(T-1) + \frac{(T-1)^2}{2}\right] . U(T-1)$$

Clairement, le pire cas pour $\alpha = \frac{\lambda_2}{\lambda}$ est $\alpha = 1$; l'insécurité est alors donnée par :

$$IS(T) = \underset{1}{E} (S(T,1)) = \underset{1}{E} (\lambda^2 \frac{(T-1)(T+1)}{2} /_{1\leq T}).Prob(1\leq T)$$

L'expression $\lambda^{\frac{2(T-1)(T+1)}{2}}$ est concave en 1

En posant $1_o = E(1/_{1\leq T})$ (c'est-à-dire 1_o est la moyenne des latences inférieures à T) et en appliquant la propriété de Jensen, nous obtenons la majoration :

$$IS = \lambda^{\frac{2(T+1_o)(T-1_o)}{2}} \; Prob(1\leq T)$$

Le pire cas $1_o = 0$ donne $IS = \frac{\lambda^2 T^2}{2} Prob(1\leq T)$

Ce résultat n'est guère satisfaisant (nous obtenons la moitié du résultat donné par le modèle pessimiste), ce qui nous conduit à chercher une majoration plus fine de A_4.

Deuxième majoration

Nous prendrons :

si $1' \geq T$ $\quad A_4 = \lambda_2^2 \frac{(T-\lambda_2)^2}{2} U(T-1)$

si $1' \leq T$, dans les deux cas, A_4 est majoré par $\lambda^2 1'(T-1) U(T-1)$

Posons $X = \text{Prob}(1 \leq T)$ et calculons IS :

$$IS = \lambda^2 \{ E_1 [\alpha.1.(T-1) + \alpha(1-\alpha)\frac{(T-1)^2}{2} /_{1\leq T}] \cdot X$$

$$+ E_{1,1'} [\alpha^2.1'.(T-1)/_{1'\leq T, 1\leq T}] \cdot X^2$$

$$+ E_1 [\alpha^2.\frac{(T-1)^2}{2} /_{1\leq T}] \cdot X \cdot (1-X) \}$$

En décomposant le premier terme suivant X et (1-X) (c'est-à-dire en introduisant 1' dans ce terme :

$$IS(T) = \lambda^2 \{ E_{1,1'} [\alpha 1(T-1) + \alpha(1-\alpha)\frac{(T-1)^2}{2} /_{1'\leq T, 1\leq T}] X^2$$

$$+ E_1 [\alpha 1(T-1) + \alpha (1-\alpha)\frac{(T-1)^2}{2} /_{1\leq T}] X (1-X)$$

$$+ E_{1,1'} [\alpha^2 1' (T-1)/_{1'\leq T, 1\leq T}] \cdot X^2$$

$$+ E_1 [\alpha^2 \frac{(T-1)^2}{2} /_{1\leq T}] . X . (1-X) \}$$

d'où

$$IS(T) = \lambda^2 \{ E_{1,1'} [\alpha \frac{(T-1)(T+1)}{2} - \alpha^2 \frac{(T-1)^2}{2} + \alpha^2 1'(T-1) /_{1\leq T, 1'\leq T}] X^2$$

$$+ E_1 [\alpha \frac{(T-1)(T+1)}{2} /_{1\leq T}] \cdot X (1-X)$$

Les fonctions dont on cherche à calculer l'espérance sont concaves en 1 (elles sont polynomiales de degré 2, le coefficient de 1^2 étant négatif).

De même la fonction du premier terme est linéaire en 1', donc concave en 1'. Nous pouvons donc majorer l'insécurité en introduisant $l_o = E(l/_{1 \leq T})$

Il vient :

$$IS(T) = \lambda^2 \{[\alpha \frac{T^2 - l_o^2}{2} + \alpha^2 (T-l_o)(l_o - \frac{T-l_o}{2})] X^2 + \alpha . \frac{T^2-l_o^2}{2} . X.(1-X)\}$$

soit :

$$IS(T) = \lambda^2 [\alpha . X . \frac{T^2-l_o^2}{2} + \alpha^2 X^2 (T-l_o) \frac{(3l_o - T)}{2}]$$

αX est un paramètre inconnu ($0 \leq \alpha X \leq 1$) ; nous cherchons un pire cas sur ce paramètre.
L'étude de la dérivée en αX fait apparaître deux cas :

- si $l_o \geq \frac{T}{7}$, IS est croissante en αX sur l'intervalle [0,1]

  et le pire cas est donc donné par $\alpha X = 1$

  et : $IS(T) = 2 \lambda^2 l_o (T-l_o)$

- si $l_o \leq \frac{T}{7}$, IS admet un maximum en $\hat{\alpha X} = \frac{T+l_o}{2(T-3l_o)}$

Ce maximum est $IS(T) = \lambda^2 \frac{(T-l_o)(T+l_o)^2}{8(T-3l_o)}$

Ainsi, nous constatons que :

. lorsque l_o tend vers 0, IS tend vers $\frac{\lambda^2 T^2}{8}$: le duplex interne est donc moins favorable que le duplex externe : quand la latence tend vers 0 (c'est-à-dire qu'on s'approche du modèle optimiste), la sécurité ne tend pas vers 1.

. à l'autre borne, $l_o = T$, l'insécurité en revanche est nulle (les pannes n'ont pas le temps de se manifester).

. enfin, le pire cas absolu est obtenu dans la deuxième tranche des valeurs de l_o (pour $l_o > \frac{T}{7}$, c'est-à-dire avec $\alpha X = 1$) pour $l_o = \frac{T}{2}$.

Ce pire cas absolu donne :

$$IS(T) = \frac{\lambda^2 T^2}{2}$$

Ainsi, nous constatons que pour $l_o \geq \frac{T}{7}$ le résultat obtenu est le même que pour le duplex externe ; en particulier, le pire cas obtenu pour $l_o = \frac{T}{2}$ est identique.

VI.4. CONCLUSION

La modélisation de la stratégie duplex interne s'est avérée plus compliquée que pour les stratégies à implantation externe : cette complexité est introduite par l'existence de plusieurs modes de pannes dans une unité.
Ce problème de complexité a été levé en effectuant des simplifications de pire cas, ce qui a permis de conserver la philosophie guidant nos évaluations et de limiter les hypothèses soutenant ce modèle au jeu suivant :

- hypothèses H_o, H_1, H_2, H_5 déjà discutées
- hypothèse concernant le taux λ_3 d'occurrence de panne dans une unité déjà en panne de type 1 : ce taux est supposé être de l'ordre du taux de pannes de l'unité.

Les calculs effectués donnent une expression de l'insécurité qui dépend, outre de λ et de T, de trois paramètres : l_o, moyenne des latences inférieures à T ; α représentant la proportion des pannes de type 2 : $\alpha = \frac{\lambda_2}{\lambda}$ et $X = Prob(l \leq T)$ probabilité que la latence soit inférieure à T.

Suivant la connaissance que l'on peut avoir de la latence, on prendra l'expression de pire cas de l'insécurité sur ces trois paramètres où on fixera une valeur de l_o et on obtiendra, pour cette valeur, une expression de pire cas de l'insécurité sur α et X ; pour certaines valeurs de l_o, l'analyse fait apparaître un pire cas non trivial.

L'analyse de fiabilité de la stratégie originale décrite au chapitre 3 sera menée, dans le prochain chapitre, de manière identique à l'analyse effectuée ici.

CHAPITRE VII

EVALUATION DU SYSTEME CARL

VII.1. INTRODUCTION

L'objet de ce chapitre est de proposer une évaluation de la fiabilité du bi-calculateur décrit au chapitre III. Après les modifications apportées, ce système peut être partitionné en deux sous-ensembles :

. sous-ensemble capteur : une stratégie tripliquée (par exemple TMR) est appliquée à deux capteurs Cp1 et Cp2 ; la gestion de cette stratégie est assurée à l'extérieur du sous-ensemble par les calculateurs.

. sous-ensemble calculateur : une stratégie originale (la stratégie décrite en VII.2.4.).

Pour que le système fonctionne, il faut que tous les sous-ensembles fonctionnent et la fiabilité de mission du système décrit :

$$R_{SYSTEME}(T) = R_{TMR\ Cp1}(T) \cdot R_{TMR\ Cp2}(T.) \cdot R_{Calculateurs}(T)$$

où $R_{TMR\ Cp1}$ et $R_{TMR\ Cp2}$ sont les fiabilités des stratégies TMR externes appliquées aux unités Cp1 et Cp2.

et $R_{Calculateurs}$ est la fiabilité de la stratégie appliquée aux calculateurs.

La stratégie TMR externe peut être évaluée très simplement : la pire latence de panne est la latence nulle, car l'existence d'une latence ne fait que retarder l'échec de la stratégie : cette stratégie ne fournit qu'un masquage mais pas de détection et reconfiguration ; l'effet de deux pannes simultanées est donc identique à l'effet de deux pannes successives : la stratégie échoue donc après deux pannes ; pour λT petit devant 1, nous obtenons :

$$IF_{TMR} \simeq 3\ \lambda^2\ T^2$$

Il nous suffit donc d'étudier la "stratégie calculateurs".
Nous situerons tout d'abord le problème en donnant des bornes à la fiabilité que nous cherchons à obtenir, bornes données par le modèle optimiste et le modèle pessimiste, avant de donner le modèle à latence distribuée et les calculs qui s'y rapportent.

VII.2. MODELES EXTREMES

VII.2.1. Modèle optimiste ([PUL79], [CAS81a])

Si nous supposons que la latence des pannes est nulle, il est clair qu'il faut qu'il y ait au moins une panne dans chaque duplex pour que la stratégie échoue ; si R(T) est la fiabilité d'un calculateur, le taux de panne d'un calculateur, nous obtenons :

$$IF(T) = (1 - R(T)^2)^2 = (1 - e^{-2\lambda T})^2$$

pour λT petit devant 1 :

$$IF(T) = 4\ \lambda^2\ T^2$$

VII.2.2. Modèle pessimiste

Si nous ne faisons pas d'hypothèse sur la latence, il nous faut considérer que les pannes ne se manifestent que lorsqu'elles deviennent critiques.
Ceci nous conduit à estimer que l'occurrence de pannes dans deux quelconques des quatre unités font échouer la mission (en faisant échouer le duplex lorsque les deux unités font partie de la même paire, ou comme précédemment, lorsqu'elles appartiennent à des pannes différentes). Nous obtenons :

$$IF(T) = (1 - R)^4 + 4(1 - R)^3\ R + 6(1 - R)^2\ R^2 \simeq 6(1 - R)^2\ R^2 \simeq 6\ \lambda^2\ T^2$$

Nous obtenons ainsi une première "fourchette" pour l'infiabilité :

$$4\ \lambda^2\ T^2 \leq IF(T) \leq 6\ \lambda^2\ T^2$$

VII.3. CONSTRUCTION DU MODELE A LATENCE DISTRIBUEE

VII.3.1. Modes de pannes

Chaque calculateur a trois fonctions : fonction de calcul de résultats, fonction d'émission d'alarme (vers l'autre paire) et fonction de reconfiguration (réagir à la réception d'un signal d'alarme de l'autre paire pour continuer en bi-simplex) : il s'agit de se dérouter vers de nouveaux programmes). Comme précédemment, nous pouvons faire l'inventaire des modes de pannes :

- la fonction "calcul" comporte deux états représentés par la variable E_c

  $E_c = 0$: aucune panne n'altère la fonction calcul

  $E_c = 1$: une panne fait délivrer un résultat faux pour certaines reconfigurations de l'état interne et des entrées.

- la fonction "alarme" peut être dans deux modes de pannes, "pas d'alarme" et "fausse alarme", repérés par les variables E_{na} et E_{fa} :

  $E_{na} = 0$: une alarme est émise quand cela est nécessaire

  $E_{na} = 1$: dans certains cas, alors que l'alarme devrait être émise, elle ne l'est pas.

  $E_{fa} = 0$: aucune fausse alarme n'est émise

  $E_{fa} = 1$: dans certains états de l'unité, une fausse alarme est émise,

  de même, la fonction reconfiguration comporte deux modes de pannes, repérés par les variables E_{nr} et E_{fr}.

  $E_{nr} = 0$: quand cela est nécessaire, une reconfiguration est effectuée

  $E_{nr} = 1$: dans certains cas, alors qu'une reconfiguration devrait être effectuée, elle ne l'est pas.

  $E_{fr} = 0$: aucune fausse reconfiguration (c'est-à-dire une reconfiguration qui n'a pas été sollicitée) n'est effectuée.

E_{fr} = 1 : dans certains états, une fausse reconfiguration est effectuée (alors que cela n'a pas été demandé, l'unité change éventuellement de programme, et ne fait plus les comparaisons du duplex).

En première analyse, chaque unité a 2^5 = 32 états possibles, ce qui donnerait un modèle à 32^4 états, et qui justifie donc la recherche de simplifications.

VII.3.2. Construction du modèle simplifié

Une analyse de pire cas permet de réduire le nombre d'états d'une unité :

- la panne de fausse reconfiguration a exactement le même effet que la panne de calcul : quand cette panne se manifeste, le calculateur soit exécute le programme de l'autre duplex, soit continue le même programme mais en simplex, sans échanger les valeurs à comparer avec son partenaire : la comparaison va échouer, cette panne apparaissant comme un déséquencement du calculateur, c'est-à-dire comme une panne de calcul : la variable s_{fr} peut donc être supprimée.

- de même, les pannes de fausse alarme ont au pire le même effet que les pannes de calcul : elles ne peuvent pas, en cas de double panne, conduire à l'échec d'un duplex mais dans le cas d'une manifestation d'une panne simple, elles conduisent à une reconfiguration : représenter ces pannes de fausses alarmes comme des pannes de calcul est donc bien un choix pessimiste (double panne dans le duplex toujours catastrophique).

- à ce stade des simplifications, l'état d'une unité n'est plus représenté que par un triplet (s_c, s_{na}, s_{nr}) pouvant prendre 8 valeurs.

Il est clair que l'état (1,1,1) (calcul incorrect, pas d'alarme et pas de reconfiguration), rend l'unité incapable de faire évoluer l'état du système, et constitue un pire cas pour tout état du type $(1, \phi, \phi)$ (ϕ = 0 ou 1 : calcul incorrect et tout état des fonctions d'alarme et de reconfiguration) : tous ces états peuvent donc être inclus dans l'état (1,1,1).
De même l'état (0,1,1) est un pire cas pour les états (0,1,0) et (0,0,1), qui lui seront inclus.

Nous obtenons ainsi une représentation d'unité à trois états (figure 1) où les états ont la signification suivante :

Etat 0 : unité saine

Etat 1 : alors que cela serait nécessaire, il n'y a pas d'alarme ou de reconfiguration.

Etat 2 : idem, mais de plus le calculateur est en "panne de calcul".

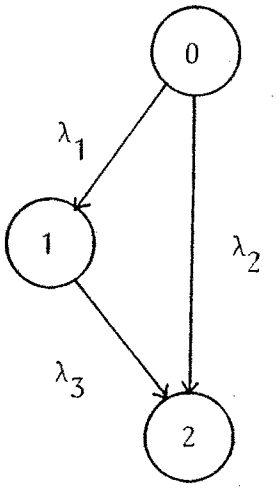


Figure 1 : Modèle simplifié des états de pannes d'une unité

Les taux de transitions sont définis à partir des mêmes remarques que dans le chapitre précédent :

$$\lambda_1 + \lambda_2 = \lambda \quad ; \quad \lambda_3 \simeq \lambda$$

Le modèle correspondant à la stratégie comporte 3^4 états plus les états d'arrêts et d'erreurs, soit 83 états. Il ne sera pas construit complètement puisque nous nous intéresserons uniquement aux plus courts chemins menant à l'état d'erreur : ces chemins correspondent à deux classes de cas :

- l'échec de l'un des duplex conduisant les unités de celui-ci à émettre de mauvais résultats dans donner l'alarme : il suffit donc de recenser deux fois les quatre chemins 1 , 2 , 3 et 4 décrits au chapitre VI.

- l'échec de la stratégie due à la succession de deux pannes, chacune dans un duplex différent. Les plus courts chemins conduisant à un tel échec, sont représentés dans la figure 2 où les états de pannes sont donnés par des couples (a,b) tels que : a = état de panne de la première unité tombant en panne ; b : état de panne de la deuxième unité tombant en panne dans l'autre duplex.

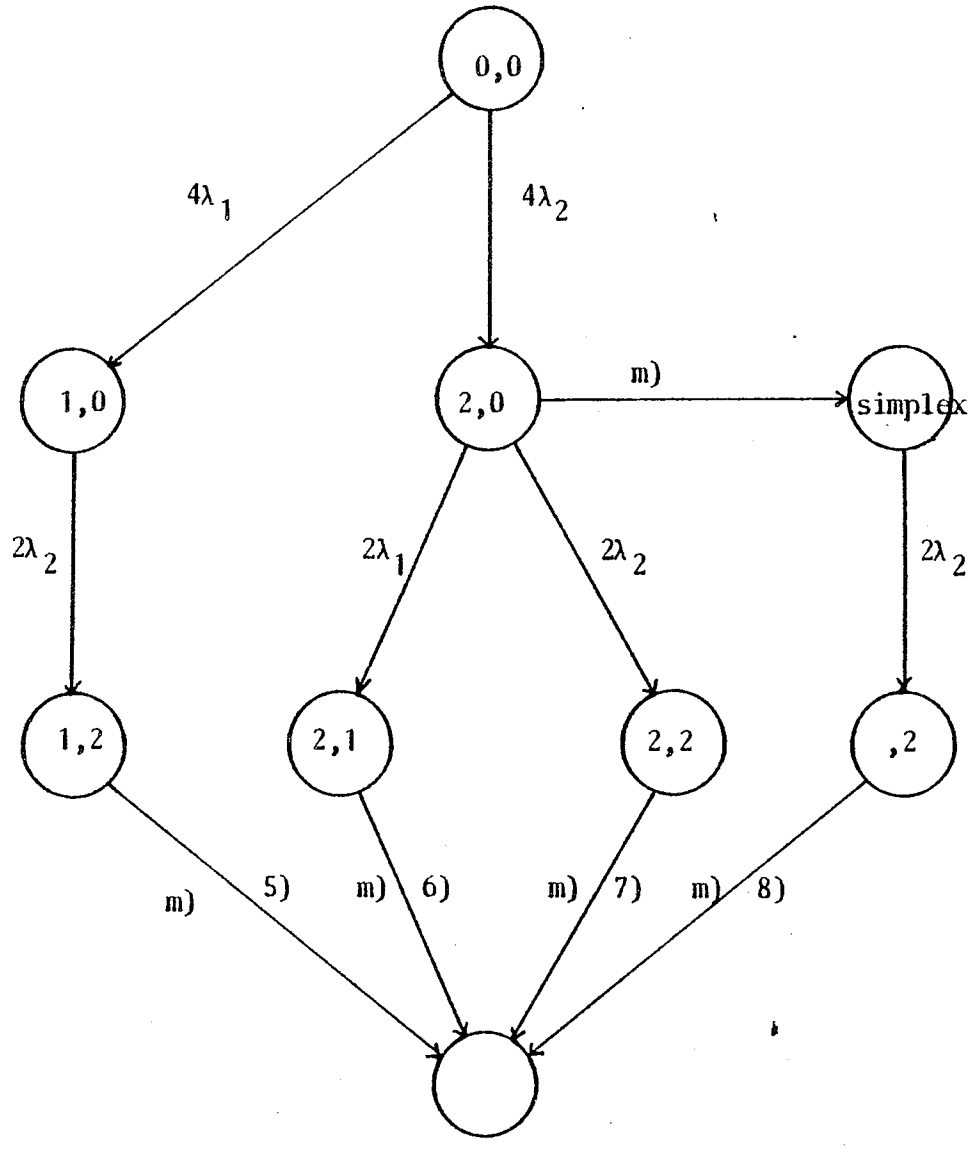


Figure 2 : Représentation des plus courtes successions de pannes de paires différentes menant à l'erreur.

VII.4. CALCUL DE LA FIABILITE

Les chemins 5 , 6 , 7 , 8 décrits dans la figure, correspondent aux cas suivants :

5) panne de type 1 à t_1 ; panne de type 2, dans une unité de l'autre paire, à t_2, se manifestant après une latence 1 tirée au sort avec :

$0 \leq t_1 \leq t_2 \leq t_2 + 1 \leq T$

(la panne de type 2 provoque lors de sa manifestation l'émission d'une alarme : mais dans l'autre paire, une unité est en panne de type 1, incapable de faire la reconfiguration).

6) panne de type 2 à t_1 tirant au sort une latence 1 et panne de type 1 à t_2 avec :

$0 \leq t_1 \leq t_2 \leq t_1 + 1 \leq T$

(situation similaire mais avec un ordre d'arrivée des pannes différent).

7) panne de type 2 à t_1 tirant au sort une latence 1 et panne de type 2 à t_2 tirant au sort une latence 1' avec :

$0 \leq t_1 \leq t_2 \leq t_1 + 1$

et $t_1 + 1 \leq T$ chemin 7.1

ou $t_2 + 1' \leq T$ chemin 7.2

(pannes complètes successives de deux unités manifestées dans l'ordre d'apparition ou l'ordre inverse).

8) panne de type 2 à t_1 tirant au sort une latence 1 et panne de type 2 à t_2 tirant au sort une latence 1' avec :

$0 \leq t_1 \leq t_1 + 1 \leq t_2 \leq t_2 + 1' \leq T$

(panne manifestée provoquant la reconfiguration puis panne dans un simplex).

Pour λT petit devant 1, les termes du calcul correspondant aux chemins 5 , 6 et 8 sont donnés par :

5) : $A_5 = \int_o^{T-1} 4\ \lambda_1 \int_{t_1}^{T-1} 2\ \lambda_2\ dt_2 \cdot dt_1 \cdot U(T-1)$

6) : $A_6 = \int_o^{T-1} 4\ \lambda_2 \int_{t_1}^{t_1+1} 2\ \lambda_1 \cdot dt_2 \cdot dt_1 \cdot U(T-1)$

8) : $A_8 = \int_o^{T-1-1'} 4\ \lambda_2 \int_{t_1+1}^{T-1'} 2\ \lambda_2 \cdot dt_2 \cdot dt_1 \cdot U(T-1)$

Le calcul de ces termes donne :

$$A_5 = 4 \cdot \lambda_1 \cdot \lambda_2 \cdot (T-1)^2 \cdot U(T-1)$$

$$A_6 = 8 \cdot \lambda_1 \cdot \lambda_2 \cdot 1 \cdot (T-1) \cdot U(T-1)$$

$$A_8 = 4 \cdot \lambda_2^2 \cdot (T-1-1')^2 \cdot U(T-1)$$

Le calcul du terme A_7 correspondant au chemin 7) est plus complexe et sera abordé comme suit :

les chemins 7.2 et 8 correspondent à la même succession de pannes (deux pannes de type 2) mais diffèrent par les dates de manifestation :

7.2) : $0 \leq t_1 \leq t_2 \leq t_1 + 1$ et $t_2 + 1' \leq T$

8) : $0 \leq t_1 \leq t_1 + 1 \leq t_2 \leq t_2 + 1' \leq T$

Les conditions exprimées sont exclusives et leur union, qui correspond à "8 ou 7.2", donne :

$$0 \leq t_1 \leq t_2 \leq t_2 + 1' \leq T$$

à cette condition correspond le terme :

$$A_8 + A_{7.2} = \int_0^{T-1'} 4\,\lambda_2 \int_{t_1}^{T-1'} 2\,\lambda_2\, dt_2\, dt_1\, U(T-1')$$

Soit $A_8 + A_{7.2} = 4\,\lambda_2^2\,(T-1)^2$ (en notant 1 au lieu de 1')

Le terme $A_{7.1}$ correspondant au chemin 7.1 correspond aux conditions :

$$0 \leq t_1 \leq t_2 \leq t_1 + 1 \leq T$$

d'où

$$A_{7.1} = \int_0^{T-1} 4\,\lambda_2 \int_{t_1}^{t_1+1} 2\,\lambda_2 . dt_2 . dt_1 . U(T-1)$$

$$A_{7.1} = 8\,\lambda_2^2 \,.\, 1.(T-1)\,.\,U(T-1)$$

La somme des termes est donc :

$$S = A_5 + A_6 + A_{7.1} + (A_{7.2} + A_8) = 8\,\lambda_2\,\lambda\,[\frac{(T-1)^2}{2}] + 1(T-1) = 8\,\lambda_2\,\lambda[\frac{T^2-1^2}{2}]\,.U(T-1$$

L'infiabilité de la stratégie est donnée par :

$$IF(T) = IF_1(T) + IF_2(T)$$

où IF_1 est l'infiabilité due à l'échec d'un des duplex

et IF_2 est l'infiabilité due à des pannes dans les deux paires

(c'est-à-dire correspondant aux chemins 5 , 6 , 7 , 8)

Nous avons :

$$IF_2(T) = \underset{1}{E}\,(8\,\lambda_2\,\lambda[\frac{T^2-1^2}{2}]/_{T\leq 1})\ .\ Prob(1\leq T)$$

La fonction considérée étant concave en 1, si nous adoptons les mêmes notations que précédemment ($1_o = E(1/_{T\geq 1})$, $\alpha = \frac{\lambda_2}{\lambda}$, $X = Prob(1\leq T)$)

Nous obtenons la majoration :

$$IF_2(T) = \lambda^2.4.\alpha.X.(T^2-1_o^2)$$

Le terme $IF_1(T)$ est obtenu dans le chapitre précédent par : $IF_1(T) = 2.IS(T)$

Soit :

$$IF_1(T) = \lambda^2[\alpha.X.(T^2-1_o^2) + \alpha^2.X^2(T-1_o)(31_o-T)]$$

Nous obtenons alors l'infiabilité de la stratégie calculateur :

$$\boxed{IF(T) = \lambda^2\,[5\,\alpha\,X(T^2-1_o)^2 + \alpha^2\,X^2\,(T-1_o)(31_o-T)]}$$

αX est un paramètre inconnu ($0 \leq \alpha X \leq 1$) pour lequel nous cherchons un pire cas ; l'analyse de la dérivée en αX de IF fait apparaître que IF est croissante pour αX variant de 0 à 1.

Le pire cas est donc obtenu pour $\alpha X = 1$:

$$IF(T) = \lambda^2[5(T^2-1_o^2) + (T-1_o)(31_o-T)]$$

$$IF(T) = {}^{2}\{(T^2-1_o)[(5T+51_o) + (31_o - T)]\} =$$

$$\boxed{IF(T) = 4.\lambda^2.(T-1_o).(T+21_o)}$$

Si nous analysons la dépendance de l'infiabilité par rapport à 1_o, nous obtenons :

- si $1_o = T$, $IF(T) = 0$ (les pannes ne se manifestent pas pendant la mission)
- si $1_o = 0$, $IF(T) = 4\,\lambda^2\,T^2$ (ce qui correspond bien au modèle optimiste)

- le pire cas est obtenu pour $l_o = \frac{T}{4}$, et est :

$$IF(T) = 4 \left(\frac{3T}{4}\right)^2 \cdot \frac{3T}{2} = \underline{4.5\ \lambda^2\ T^2}$$

VII.5. CONCLUSION

L'analyse de fiabilité de la stratégie appliquée aux calculateurs a pu être menée après simplification, par analyse de pire cas, du modèle. Une analyse de pire cas sur les facteurs mal connus ($\alpha = \frac{\lambda 2}{\lambda}$ et Prob(l≤T)) permet d'exprimer l'infiabilité en fonction de λ, T et de la moyenne de latence inférieure à T

Le résultat le plus fort de ce chapitre semble être que l'analyse de pire cas faite sur cette latence donne un résultat très proche du résultat obtenu par analyse optimiste : nous savions que : $4\ \lambda^2\ T^2 \leq IF(T) \leq 6\ \lambda^2\ T^2$

Le pire cas absolu donne $IF(T) = 4.5\ \lambda^2\ T^2$; ainsi pouvons-nous "garantir" que (sous les hypothèses de bases H_o, H_1, H_2, H_5) l'infiabilité est très proche du résultat optimiste et ce quelle que soit la distribution de latence. Ainsi, cette stratégie présente la caractéristique d'avoir des résultats par analyse de pire cas qui s'avèrent acceptables. Dans [CAS81a], nous avons défini les stratégies présentant cette caractéristique comme étant "robuste La recherche et l'usage de stratégies robustes apportent à notre sens un élément de réponse au problème de la validation des systèmes tolérant les pannes : elles permettent en quelque sorte de "garantir" un chiffre de sûreté de fonctionnement acceptable.

En utilisant les taux de pannes des unités donnés dans [PUL79] ($\lambda_{Cal}=2.5\ 10^{-5}$/H, $\lambda_{Capt}=2.8\ 10^{-5}$/H), l'infiabilité du système pour une mission de 10 Heures est : $1-[(1-4.5\lambda^2_{Cal}T^2)(1-3\lambda^2_{Capt}T^2)^2]$ soit : $7.5\ 10^{-7}$.

Enfin, nous conclurons cette partie en rappelant que la méthode proposée a permis d'évaluer, de manière "rigoureuse" - c'est-à-dire en faisant clairement apparaître le jeu d'hypothèses sur lequel est basée l'évolution, en cherchant à justifier ces hypothèses et en veillant à réduire autant que possible ce jeu - un large éventail de stratégies. Notamment, le modèle à latence distribuée, pour lequel on a décrit les conditions qui le rendent applicable, permet soit d'obtenir une évaluation de pire cas sur toute latence, soit, s'il est possible de fixer une moyenne des latences l_o, une évaluation en fonction de l_o. Le tableau de la figure 5 résume les résultats obtenus avec ce modèle, pour les stratégies étudiées, dans le contexte A.

Enfin, cette approche nous a permis de détailler et de prendre en compte les phénomènes qui mettent en échec les stratégies étudiées et relèvent de la non-couverture :

- les pannes transitoires et les pannes de mode commun ont été exclues par hypothèse (hypothèses H0,H1,H2) ;
- les phénomènes de manifestation simultanée d'erreur et les phénomènes d'occurence de pannes sur les mécanismes de détection ont été traités par pire cas ;
- le phénomène de latence d'erreur fait l'objet d'hypothèses numériques "crédibles".

Cette approche présente, à notre avis, deux avantages par rapport à l'approche de couverture [BOU71] :

. le caractère complexe du facteur de couverture rend celui-ci très difficile à évaluer [HOP80] ;

. l'application de ce modèle au duplex donne un taux d'insécurité

TIS = $2\lambda(1-c)$;

ainsi, pour $\lambda = 10^{-5}/h$, si l'on veut atteindre un taux d'insécurité de $10^{-9}/h$, il faut $1-c = 0.5\ 10^{-4}$. Si l'on veut conforter une telle hypothèse par l'expérience, il faudra étudier un nombre considérable (de l'ordre de 10^4) de passivation de pannes : il semble donc difficile de rendre cette hypothèse numérique crédible.

STRATEGIE		MODELE "OPTIMISTE" $l_o = 0$	DEPENDANCE EN l_o	Pire cas en l_o	
				l_o	Résultat
4H externe	IF(T)	$4(\lambda T)^3$	$6\lambda^2.l_o.(T-l_o)+4(\lambda T)^3$	$\frac{T}{2}$	$3\frac{\lambda^2T^2}{2} + 4\lambda^3T^3$
5H externe	IF(T)	$5(\lambda T)^4$	$6\lambda^2.l_o.(T-l_o)$ +	$\frac{T}{2}$	$3\frac{\lambda^2T^2}{2}$ +
4MRSP externe	IF(T)	$4(\lambda T)^3$	$12\lambda^2.l_o.(T-l_o)+4(\lambda T)^3$	$\frac{T}{2}$	$3\lambda^2T^2 + 4\lambda^3T^3$
5MRSP externe	IF(T)	$5(\lambda T)^4$	$30\lambda^3.l_o.T.(T-l_o)+5(\lambda T)^4$	$\frac{T}{2}$	$\frac{5}{2}\lambda^3T^3 + 5\lambda^4T^4$
Duplex externe	IS(T)	0	$2\lambda^2 l_o (T-l_o)$	$\frac{T}{2}$	$\frac{2T^2}{2}$
Duplex interne	IS(T)	$\frac{\lambda^2T^2}{8}$	si $l_o \leq \frac{T}{7}$, $\frac{\lambda^2(T-l_o)(T+l_o)^2}{8(T-3l_o)}$ si $\frac{T}{7} \leq l_o$, $2\lambda^2.l_o.(T-l_o)$	$\frac{T}{2}$	$\frac{\lambda^2T^2}{2}$
Stratégie Calculateur	IF(T)	$4\lambda^2T^2$	$4\lambda^2(T-l_o)(T+2l_o)$	$\frac{T}{4}$	$4,5\ .\ \lambda^2T^2$

Figure 3 : Résultats d'évaluations par le modèle à latence distribuée

CHAPITRE VIII

ECRITURE DU LOGICIEL DE TOLERANCE AUX PANNES DU SYSTEME CARL

VIII.1. INTRODUCTION

Le choix d'une implantation logicielle des mécanismes de tolérance aux pannes est souvent mis en cause par le raisonnement suivant :
ce choix consiste à remplacer des dispositifs matériels, qui peuvent tomber en panne, par des programmes ; mais, alors que les dispositifs matériels ont des taux de pannes bien connus, pris en compte dans les analyses de fiabilité, comment prouver que les programmes les remplaçant sont entièrement corrects ? Il est en effet reconnu que la production de logiciels corrects et la preuve de logiciels, sont des problèmes difficiles [HOA69][FLO67][LIV78].

Le but de ce chapitre est de répondre, dans le cas du système étudié, à la question posée. Nous n'y répondrons pas par une étude approfondie des problèmes de production et de preuve de logiciel. Ce chapitre constitue plutôt une étude de faisabilité destinée à montrer la simplicité du logiciel considéré ; cette simplicité nous permettra de répondre favorablement à l'utilisation de mécanismes logiciels, d'une part parce qu'ils sont trivialement prouvables, d'autre part parce qu'ils représentent de toutes façons une faible proportion du volume total d'un logiciel d'application dont la validation pose en revanche sans doute problème.

La recherche de simplicité a guidé les choix effectués dans la conception de la machine ; elle guidera encore la construction du logiciel de tolérance aux pannes. Cette construction sera menée en dérivant, par étapes prouvables, les spécifications du logiciel jusqu'à l'écriture des programmes.
En raison de la simplicité du problème, nous n'alourdirons pas l'exposé de la démarche par un formalisme dans les notations et dans les preuves de chaque étape (une démarche possible serait d'exprimer les spécifications du programme en terme de post-condition et pré-condition et de faire des décompositions successives du programme, en montrant que chaque décomposition permet de vérifier les spécifications de la partie décomposée) [HOA69].

Nous nous contenterons donc d'exprimer les spécifications de ces programmes puis de proposer une décomposition simple en langage algorithmique.

VIII.2. SPECIFICATIONS DU LOGICIEL DE TOLERANCE AUX PANNES

Le logiciel de tolérance aux pannes doit permettre au système de réagir aux évènements de panne conformément à la description donnée par la modélisation de la stratégie de tolérance aux pannes (chapitre VII) ; ainsi, ses spécifications initiales sont données par :

- une panne se manifeste à l'extérieur de l'unité avec une certaine latence,
- une unité saine doit :
  - . en l'absence de manifestation de panne, poursuivre sa tâche
  - . en présence d'une manifestation de panne, envoyer une alarme
  - . à la réception d'une alarme provenant de l'autre duplex, faire une reconfiguration.

D'autre part, il semble souhaitable que ce logiciel présente une caractéristique de "transparence" vis-à-vis du programme d'application :

- l'écriture du programme d'application doit pouvoir être faite indépendamment des considérations de tolérance aux pannes (le spécialiste de l'application n'est pas nécessairement spécialiste de tolérance aux pannes).
- les programmes de tolérance aux pannes ne doivent pas interférer sur les programmes d'applications (modifier des variables...).

VIII.3. HYPOTHESES DE BASE

Il convient tout d'abord d'exprimer comment les phénomènes physiques (pannes) sont observables à travers des variables observables par programme. Ces précisions seront fournies par les hypothèses 1 et 2.

Hypothèse 1 : une panne se manifeste à l'extérieur lors de la production d'une sortie suivant l'un des modes suivants :

- l'unité produit une sortie erronée (différente de la sortie correcte)
- l'unité ne produit pas de sortie avant un délai maximum.

Remarquons ensuite que le modèle donné au chapitre VII ne prévoit que les réactions du système à des pannes définitives ; ceci suppose donc que les pannes transitoires soient "filtrées" par le logiciel de tolérance aux pannes. Rappelons que l'application est supposée être réalisée par un programme cyclique et sans mémoire.

Notre définition d'une panne transitoire est précisée par l'hypothèse 2.

Hypothèse 2 : une panne est dite transitoire s'il n'y a pas manifestation de panne pendant deux cycles consécutifs ; si deux cycles consécutifs sont affectés par des manifestations de pannes, l'unité est déclarée en panne définitive.

Cette hypothèse appelle plusieurs remarques :

- le but de cette hypothèse est de proposer un critère de distinction entre pannes transitoires et définitives ;

des insuffisances de ce critère peuvent être mises à jour :

par exemple, on peut imaginer qu'une panne définitive ne soit manifestée que sur des cycles espacés (soit parce qu'une caractéristique matérielle se détériore lentement, ou que certaines actions ne sont pas faites à tous les cycles) ;

le critère choisi fera conclure fréquemment à l'existence d'une panne transitoire (un système de comptage mémorisant l'histoire des occurrences de panne offrirait un critère plus efficace sur ce point). Le critère idéal devrait assurer qu'on ne décrète pas définitive une panne transitoire et qu'on ne décrète pas transitoire une panne définitive : notons que l'évaluation de fiabilité proposée en VII est faite pour toute latence ; ainsi, si une panne définitive n'est pas immédiatement décrétée définitive (en étant déclarée transitoire), cette panne est considérée comme latente et la phénomène est pris en compte dans l'analyse. Inversement, une panne sera déclarée définitive à tort si deux pannes transitoires différentes se manifestent pendant deux cycles consécutifs : un tel évènement a une faible probabilité d'occurrence. (En toute rigueur, il conviendra d'ajouter le taux d'occurrence de cet évènement aux taux des unités dans l'analyse de fiabilité) ; l'application de l'hypothèse 2 à chaque variable (une panne est définitive si elle affecte la même variable pendant deux cycles consécutifs), permettrait de réduire encore cette probabilité, mais présenterait l'inconvénient suivant : une panne définitive peut se manifester sur deux cycles, sur des variables différentes (panne de l'unité arithmétique et logique par exemple).

La description de la stratégie impose qu'en cas de panne transitoire, le système poursuive sa tâche (au cycle suivant). Le cablage réalisé impose, pour que cette poursuite soit possible, que les unités soient resynchronisées en début de cycle par une horloge externe, et que l'on tolère que des mauvais résultats soient produits pendant un cycle : la resynchronisation doit permettre de débloquer le système dans le cas d'un déséquencement d'une unité provoquant la non production d'une variable.
Une solution plus stricte consisterait à ne pas utiliser d'horloge externe, et à ne pas laisser propager de mauvais résultats pendant un cycle ;

cela nécessiterait un dispositif "d'alarme transitoire" donnant l'ordre à toutes les unités de se resynchroniser pour démarrer un cycle et alertant l'extérieur sur l'invalidité possible des sorties. Cependant, une telle solution nécessiterait une autre analyse de fiabilité introduisant les nouvelles fonctions "alarme transitoire" et "resynchronisation" et les modes de pannes s'y rattachant. Notons enfin que l'introduction d'une phase de synchronisation alourdit, comme nous le verrons plus loin, le logiciel de tolérance aux pannes et sa preuve.

Il reste à préciser les relations entre programmes d'application et de tolérance aux pannes. Ces relations devront permettre de respecter le principe de transparence énoncé plus haut.

Appels au logiciel de tolérance aux pannes

L'hypothèse 1 induit la nécessité d'un appel à des vérifications pour toute émission d'une sortie. Il conviendra donc de remplacer (éventuellement de manière automatique) toutes les instructions de sortie par un appel au logiciel de tolérance aux pannes.

Notons que, pour diminuer la latence de détection, ce remplacement pourrait aussi être appliqué à d'autres instructions : instructions d'entrée (se "mettre d'accord" sur les variables reçues), ou certaines instructions d'affectations.

D'autre part, le logiciel de tolérance aux pannes est appelé sur réception d'une interruption-alarme.

Retour du logiciel de tolérance aux pannes

L'exécution du logiciel de tolérance aux pannes se termine par l'une des actions suivantes :

- poursuivre la tâche,
- envoyer un alarme,
- faire une reconfiguration.

Poursuivre la tâche consiste en un simple retour au programme d'application. Envoyer une alarme consiste à émettre sur un port de sortie spécifique du microprocesseur, la valeur 1 : il s'agit de l'exécution d'un programme ALARME consistant simplement en l'adressage et l'affectation de ce port. Faire une reconfiguration sera une action différente suivant le rôle de l'unité exécutant le programme : les unités 1 et 2 doivent poursuivre en simplex leur tâche (respectivement tâche 1 et tâche 2) alors que les unités 1' et 2' doivent reprendre la tâche de l'autre paire (respectivement tâche 2 et tâche 1). Pour ces unités 1' et 2', la reconfiguration consistera donc en un déroutement vers un nouveau programme, inscrit dans leur mémoire. Ce déroutement devra être effectué également à chaque réception ultérieure de l'horloge externe. Dans le cas des unités 1 et 2, nous éviterons cette solution de déroutement vers un nouveau programme : dans la mesure où il faut poursuivre la même tâche, ce nouveau programme serait similaire au précédent, et ne diffèrerait que par les points suivants :

- les ressources pour les échanges avec l'autre tâche ne sont plus les mêmes (les interfaces "intercouples" sont remplacées par les interfaces "internes au couple").
- les instructions de sorties ne conduisent plus à des vérifications et ne devraient donc pas appeler le logiciel de tolérance aux pannes.

En effet, le programme précédent peut être facilement utilisé en implantant les dispositifs suivants :

- lorsque le programme de tolérance aux pannes est appelé par le programme d'application, une vérification du mode de fonctionnement (bi-duplex ou bi-simplex) permet de décider d'un retour immédiat au programme ou de faire les comparaisons nécessaires . La variable mode de fonctionnement est modifiée en début de reconfiguration.

- dans le programme d'application, les références aux ports d'échanges sont remplacés par des références à une table contenant l'adresse de ces ports ; l'action de reconfiguration consistera donc en l'exécution d'un programme modifiant la variable mode de fonctionnement et les adresses des ports d'échanges et en un retour au programme d'application.

Les figures 1 et 2 présentent les relations logiciel d'application - logiciel de tolérance aux pannes dans le cas des unités 1, 2 d'une part et 1', 2' d'autre part.

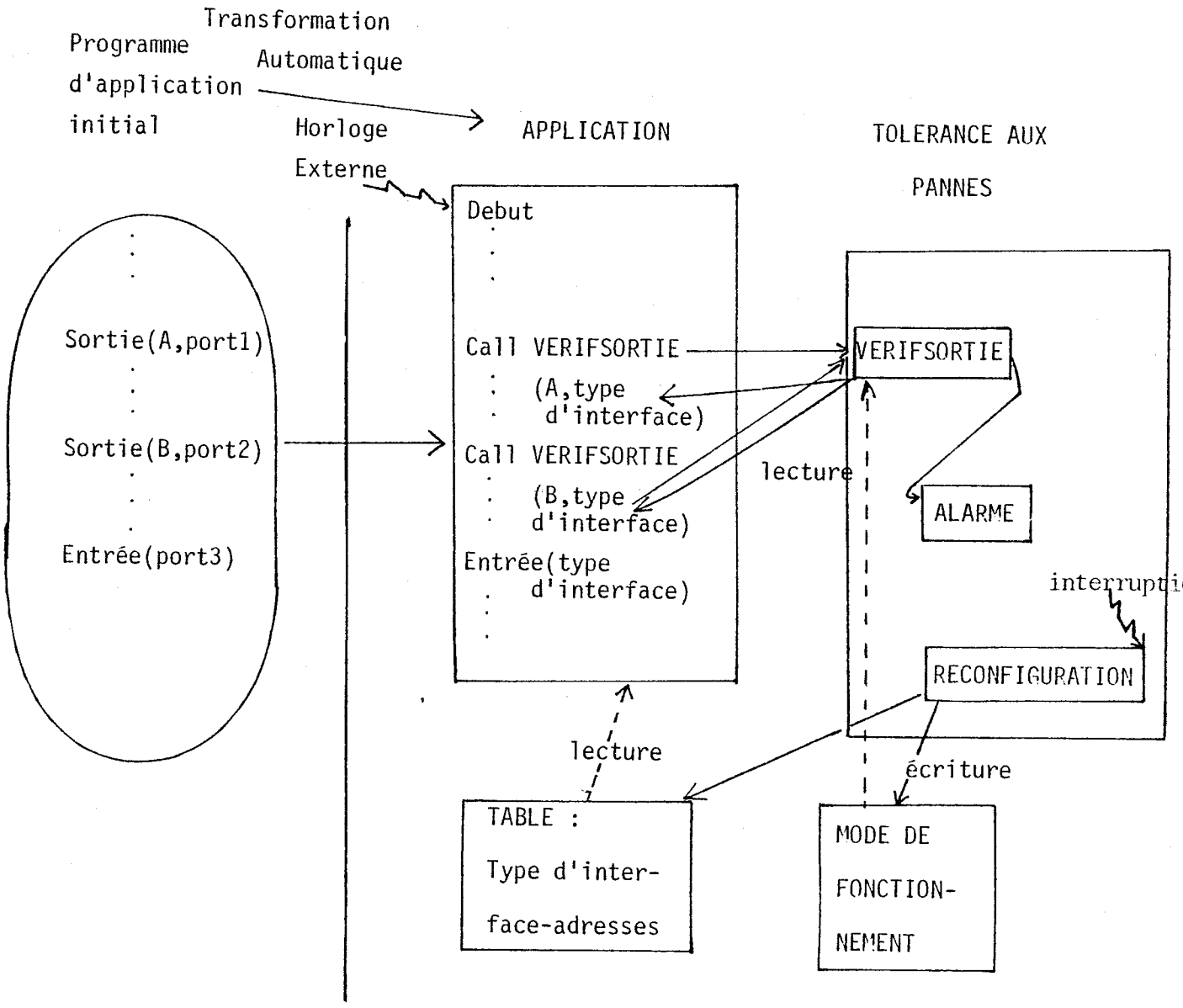


FIGURE 1 : Application et tolérance aux pannes :

Cas des unités 1 et 2.

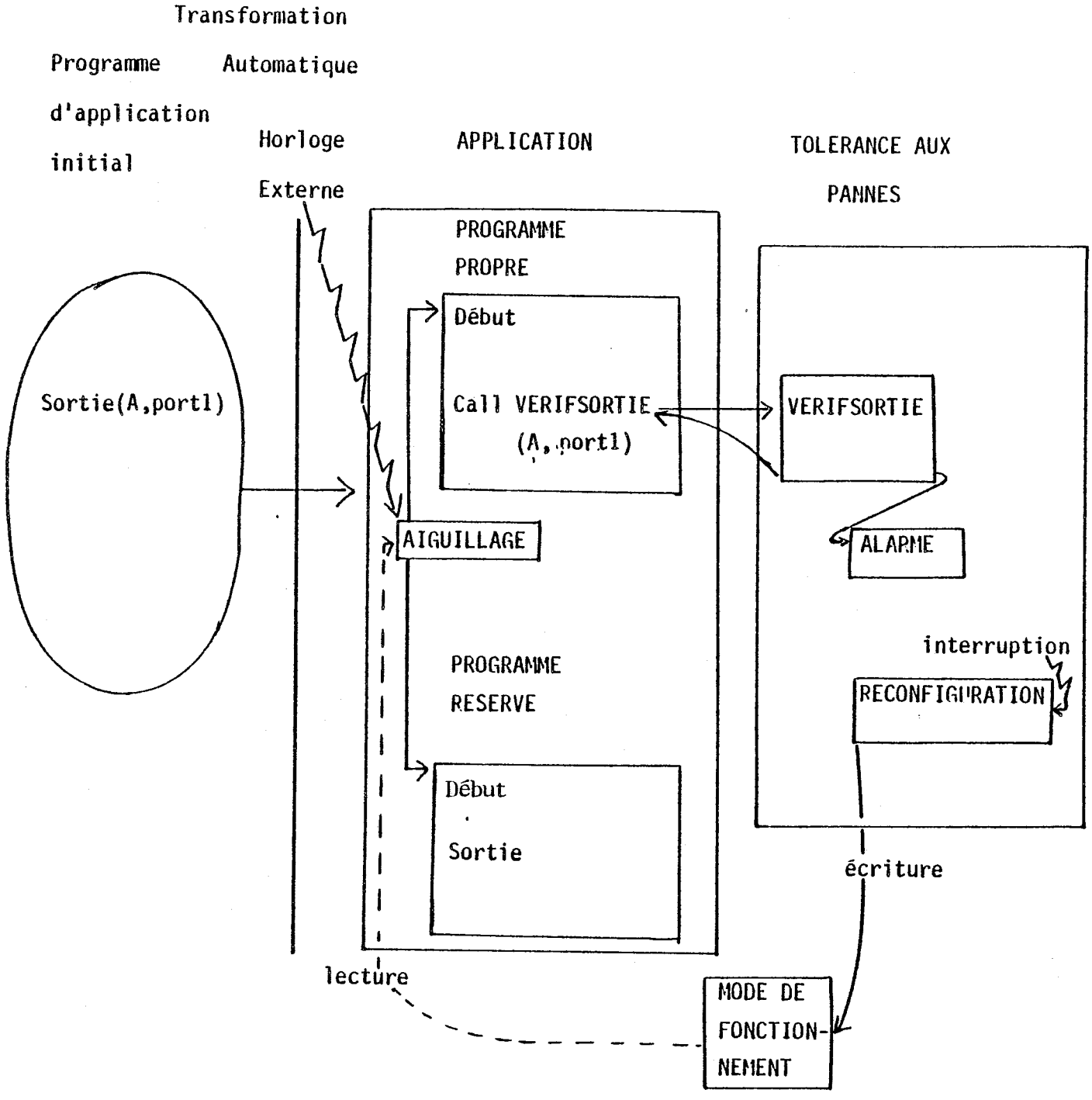


FIGURE 2 : Application et tolérance aux pannes :

Cas des unités 1' et 2'.

Ainsi, il nous faut écrire deux programmes correspondant aux deux entrées du logiciel de tolérance aux pannes : VERIFSORTIE et RECONFIGURATION (le programme ALARME consiste en une simple affectation PORTALARME := 1) RECONFIGURATION consiste également en une simple affectation de la variable MODE DE FONCTIONNEMENT et de la table d'adresse des interfaces.

VIII.4. DECOMPOSITION

Une première écriture du programme VERIFSORTIE est déduite directement des spécifications :

VERIFSORTIE Détecter une panne
Si panne alors alarme

- Première étape prise en compte des pannes transitoires

Cette prise en compte se fait en appliquant l'hypothèse 2 ; pour cela nous introduisons deux variables booléennes cycle prec et cycle

- cycle prec qui vaut "faux" s'il y a eu un "désaccord" au cours du cycle précédent et vrai sinon.
- cycle vaut faux s'il y a eu un désaccord dans le cycle en cours.

La variable cycle sera mise à jour dans "détecter une panne" qui s'écrit :

```
détecter un désaccord
si désaccord alors
    si cycleprec = faux alors alarme
    sinon cycle := faux
```

Cette écriture suppose une mémorisation des désaccords survenant au cycle précédent dans cycleprec ; la mise à jour de cycleprec se fera à chaque changement de cycle :

```
cycleprec := cycle
cycle := vrai
```

Ainsi, la prise en compte des pannes transitoires, évènements qui n'apparaissent pas dans les spécifications initiales, oblige à revenir à la définition des relations entre logiciel de tolérance aux pannes et logiciel d'application : pour toutes les unités, le signal d'horloge externe provoque l'activation d'un programme DEBUT relevant du logiciel de tolérance aux pannes qui aura la charge d'initialiser les variables cycleprec et cycle et de lancer le programme adéquat (dans le cas des unités 1' et 2', "l'aiguillage" fait partie du module DEBUT).

- Deuxième étape : décomposition de "détecter un désaccord"

Cette décomposition s'appuie sur l'hypothèse 1 et sur l'examen du cablage de la machine : il s'agit d'échanger les valeurs à sortir à travers les interfaces internes au duplex et de les comparer ; s'il y a accord, l'unité 1 (ou 2) peut émettre sa sortie à travers son interface de sortie (ou d'échange intertâche) ; la tolérance aux pannes de cet interface impose que cette émission soit vérifiée : le cablage prévoit qu'elle le soit par relecture de cette émission par l'unité 1' (respectivement 2'). Ainsi, les programmes des unités 1 ou 2 d'une part et 1' ou 2' d'autre part, sont aussi différenciés à ce niveau :

Unité 1 (ou 2) :

DETECTER UN DESACCORD

Echange

Diagnostic
Emission vers l'extérieur

Unité 1' (ou 2') :

DETECTER UN DESACCORD

Echange

Diagnostic

Relecture

Diagnostic Relecture

Echange devra faire l'émission de la valeur calculée et l'attente de la valeur du partenaire. Relecture consiste simplement en l'attente de la valeur du partenaire sur un autre interface. L'attente de la valeur doit fournir soit la valeur reçue du partenaire, soit l'indication que cette valeur n'a pas été reçue dans les délais (cf. hypothèse 1).

ECHANGE

Portsortie := val

I := 0

Tantque Portentrée = nil et I ≠ T faire I := I+1

Valreçue := portentrée ; cø : en ce point, sauf incident, le partenaire ayant émis sa valeur est en train de lire la valeur émise sur Portsortie.

I := 0

Tant que I ≠ Tempo faire I := I+1

Portsortie := nil

Remarquons que l'écriture de la boucle d'attente suppose portentrée = nil au début de l'exécution d'échange ; dans le cas d'une exécution normale, ceci est assuré par la temporisation et la mise à nil des interfaces de sorties (portsortie) en fin d'échange (après que l'on soit sûr que la valeur émise a été prise en compte).

Mais à la suite d'un cycle altéré par une panne transitoire, les ports sont susceptibles de contenir n'importe quelle valeur. Il convient donc de réinitialiser ces ports à nil au début du cycle (dans "DEBUT").
L'écriture d'une boucle pour gérer la "synchronisation" entre les unités constitue sans doute la seule difficulté de validation ; il est cependant facile de se "convaincre" de la validité de ce mécanisme en décrivant les fonctionnements des deux unités à l'aide d'outils simples (chronogramme, réseaux de Pétri...) [BER82].

L'écriture de relecture est un peu différente : il s'agit d'espionner l'émission d'une valeur attendue VAL, soit dans un échange intercouples, soit dans la sortie vers l'extérieur ; dans tous les cas, nous n'avons pas à faire l'hypothèse sur la forme de ces échanges (le protocole d'échange prévoit-il une valeur nil...) ; nous nous contenterons donc d'attendre un certain temps la valeur Val.

RELECTURE

I := 0

Tantque Portrelecture ≠ VAL et I ≠ T' faire I := I+1

DIAGNOSTIC

désaccord := (valreçue ≠ val) ; commentaire : si valreçue = nil, c'est-à-dire "temps dépassé", désaccord := vrai.

VIII.5. RECAPITULATION

Nous donnons ici une récapitulation du logiciel de tolérance aux pannes dans le cas des unités 1' ou 2' (a priori cas les plus complexes).
Le logiciel est composé de quatre modules : DEBUT, VERIFSORTIE, ALARME, RECONFIGURATION, INITIALISATION.

VIII.5.1. Module INITIALISATION

. Appels : initialisation de la machine (réception du signal "RESET")

. Sortie : explicite vers le début du programme normal d'application

. Ecriture

```
MODE DE FONCTIONNEMENT := NORMAL
cycle := cycleprec := vrai
Portsortie := nil
```

VIII.5.2. Module DEBUT

. Appels : sur réception du signal d'horloge externe

. Sorties : explicites vers un début de programme d'application

. Ecriture

```
Si MODE DE FONCTIONNEMENT = Normal  alors
  début
    Portsortie := nil ;
    cycleprec := cycle ;
    cycle := vrai ;
    Allera PROGNORMAL
  fin
Sinon  Allera PROGRESERVE
```

VIII.5.3. Module VERIFSORTIE (VAL)

. Appels : instruction sortie(VAL) du prog d'application

. Sorties : explicite vers le module alarme

ou retour normal au point d'appel

. Ecriture

(ECHANGE) : Portsortie := val

I := 0

Tantque Portentrée = nil et I ≠ T faire I := I + 1

valreçue := Portentrée

I := 0 : Tantque I ≠ Tempo faire I = I + 1

Portsortie := nil

(DIAGNOSTIC) : si Valreçue ≠ val alors

si cycleprec = faux alors Allera ALARME

sinon cycle := faux

(RELECTURE) : I := 0

Tantque Portrelecture ≠ val et I ≠ T' faire I := I + 1

Valreçue := Portrelecture

(DIAGNOSTIC) : Si Valreçue ≠ val alors

si cycleprec = faux alors Allera ALARME

sinon cycle := faux

VIII.5.4. Module ALARME

. Appels : de VERIFSORTIE

. Sorties : pas de sortie explicite ! La sortie de ce module correspond à la "mort" de l'unité (éventuellement à un mode diagnostic pour la maintenance)

. Ecriture : PORTALARME := 1

VIII.5.5. Module RECONFIGURATION

. Appel : sur réception du signal d'interruption

. Sortie : attente du signal d'horloge

. Ecriture : MODE DE FONCTIONNEMENT := dégradé

ATTENTE

VIII.6. CONCLUSION

Ce chapitre a montré la simplicité des mécanismes à mettre en oeuvre : exceptées trois boucles (très simples), le logiciel consiste en une suite d'affectations et de "conditionnelles" de sortie que la preuve en est triviale.

Cependant, à ce stade, il reste à réaliser le passage du langage évolué au langage machine. Dans le cadre de cette étude, nous ne disposions pas d'outil automatique pour ce passage ; qu'il soit manuel ou automatique, ce passage nécessite une réflexion sur le cablage de la machine afin d'éviter d'engendrer des erreurs de conception ; citons deux exemples :

- l'accès aux ports d'entrée sortie de 8 bits se fait en réalité en deux accès de 4 bits. Ainsi, l'accès aux entrées n'est pas indivisible, de sorte qu'il peut arriver que l'on détecte l'arrivée d'une entrée (port ≠ nil) mais que cette entrée soit incomplète ; il est donc nécessaire, après temporisation, de relire ce port ; ainsi, chaque lecture du port traduite par une suite d'instructions machines et sa traduction, était dépendante du contexte d'utilisation du microcalculateur.
- l'accès à la mémoire externe au microcalculateur 8748 nécessite la sortie sur un port des poids forts de l'adresse, sortie qui, décodée, permet de sélectionner le bon boitier. Ainsi, chaque accès mémoire externe consiste

en l'affectation d'un port puis l'adressage de la variable adressée ; dans ce cas encore, cette traduction en langage machine était dépendante du cablage réalisé.
Il est cependant apparu que cette traduction, bien que laborieuse, pouvait être menée de manière systématique et sans doute être "quasi-prouvée".

Nous concluerons en remarquant que le logiciel de tolérance aux pannes constitue, de toutes façons, une infime partie du logiciel total du système : dans la mesure où, en toute rigueur, le problème de la validité de tout le logiciel doit être posé, il ne saurait être question de rejeter l'utilisation de mécanismes logiciels de tolérance aux pannes en posant le problème de sa validité.

CHAPITRE IX

CONCLUSION

Le problème de la validation est, à notre avis, un problème clé de la haute sûreté de fonctionnement : s'il semble que l'on sache concevoir des systèmes très sûrs, il est en revanche difficile de démontrer qu'ils le sont.

Ce travail nous permet d'apporter des éléments de solution à ce problème :

- la validation ne nous semble accessible que pour des systèmes simples, structurés et pouvant être clairement partitionnés ; nous avons évoqué des critères caractérisant de tels systèmes et indiqué une démarche pour les concevoir.
- les évaluations prédictives de sûreté de fonctionnement doivent prendre place dans la validation en étant basées sur un jeu réduit d'hypothèses et en étant résolument non optimistes ; la démarche proposée, qui permet de respecter ces principes, a été appliquée à des cas de compléxités diverses dans le cadre de ce travail. Nous noterons que cette approche a été appliquée à un autre système, qui n'a pas été présenté ici, qui consiste en un calculateur à deux niveaux de partition ; les résultats obtenus ont permis de guider la conception des mécanismes annexes de test en ligne (fixer les objectifs d'efficacité de ces tests) ; cette approche nous semble donc très adaptée à une utilisation "interactive" avec le processus de conception [CAS82c].

Notons enfin que ce problème de la validation comporte d'autres aspects importants, qui font l'objet de nombreuses études, et que nous avons seulement évoqués ici : évitement, tolérance et mesure des fautes de conception (notamment logicielles : cet aspect est improprement appelé "fiabilité du logiciel"), plans d'essai (injection et simulation de fautes et de pannes, mesures statistiques).

BIBLIOGRAPHIE

AND75 G.A. ANDERSON & E.D. JENSEN : "Computer Interconnection Structures : Taxonomy, Characteristice and Examples", ACM Computing Surveys, Vol 7, n°4,dec 1975.

AND81 T. ANDERSON & P.A. LEE : "Fault Tolerance - Principles and Practise", Prentice Hall Editeurs, 1981.

AVI71 A. AVIZIENIS : "Arithmetic Error Codes",IEEE Transactions on Computer, Vol C-20, n°11, Nov71.

AVI82 A. AVIZIENIS : "The Four Universe Information System Model for the Study of Fault Tolerance", Proceedings of the 1982 International Symposium on Fault Tolerant Computing, IEEE n°82CH1760-8.

BEL77 C. BELLON : "Etude de la dégradation progressive dans les systèmes répartis, Thèse de 3èmecycle,sept 77, INP Grenoble.

BEO78 C. BEOUNES & F. CEREJA : "Design Methodology for Secure Microcomputers : Application to the Implementation of the Control of a Turbo-Jet Enginé", Proceedings of the 1978 International Symposium on Fault Tolerant Computing, IEEE n°78CH1286-4C.

BER82 G. BERTHELOT & al : "Petri Net Modelling and Reliability of Distributed Algorithms", Application and Theory of Petri Nets, Springer Verlag, 1982.

BRO78 F BROWAEYS & O. MURON : "Analytical Modelling of the Reliability of a Fault Tolerant Computer C.O.P.R.A.", Measuring, Modelling and Evaluating Computer Systems, H. BEILNER & E. GELENBE Eds, North Holland 77.

BOU71 W.G. BOURICIUS & al : "Reliability Modeling for Fault Tolerant Computer, IEEE Transactions on Computers, Vol C-20, n)11, Nov71.

CAS81a P. CASPI & J. PULOU : "A Method for Improving the Reliability of Functionally Distributed Networks", Jouranl of Digital Systems, Vol IV, Issue 4, 1981.

CAS81b P. CASPI & al : "Validation Methods for Fail Safe or Fault Tolerant Computers", Digest of FTSD Conference, Brno 1981.

CAS81c P. CASPI & E. PILAUD : "Etude Critique d'un Systeme de Sécurité Ferroviaire à Base de Microprocesseurs", Rapport de Contrat SNCF, Sept 81.

CER78 CERT DERA : "Etude Critique de Faisabilité d'architecture Multiprocesseurs en Avionique", Rapport Interne 78.

CHA76 K.M. CHANDY & C.V. RAMAMOORTHY : "Rollback and Recovery Strategies for Computer Programs", IEEE Transaction on Computers, Vol C-21, N°6 Juin76.

COU76 B. COURTOIS : "Etude d'un Calculateur Tolérant les Pannes : ses Fiabilité, sécurité, performance et Coût",Thèse de Docteur Ingénieur, Déc76, INP Grenoble.

CRO78 Y. CROUZET : "Conception de Circuits à Large Echelle d'Intégration Totalement Autotestables", Thèse de Docteur Ingénieur, Nov78,INP Toulouse

DAL73 W. DALY & al : "A Fault Tolerant digital Clocking", Proceedings of the 1973 International Symposium on Fault Tolerant Computing.

DIA74 M. DIAZ : "Conception de Systèmes totalement auto-testables et à pannes nondangereuses", Thèse d'Etat, Juin 74, Université P. SABATIER, Toulouse.

FLO67 R. W. FLOYD : "Assigning Meanings to Programs", Proceedings of Symposium in Applied Mathematics, 1967

FTC12 Proceedings of the 1982 International Symposium on Fault Tolerance Computing, Panel Session on Fundamental Concepts of Fault Tolerance

HOA69 C.A.R. HOARE : "An Axiomatic Basis of Computer Programming", Communication of ACM,12,10,1969.

HOP78 A.L. HOPKINS & al : "FTMP- A Highly Reliable Fault Tolerant Multi-Processor for Aircraft, Proceedings of the IEEE, Vol66, n°10, Oct78.

HOP80 A.L. HOPKINS : "Fault Tolerant System Design : Broad Brush and Fine Print", Computer, Vol13, Number3,mars 1980.

JEN63 P.A. JENSEN : "Quadded Nor Logic", IEEE Transactions on Reliability, Vol R15, n°3, Sept63.

KAU77 A. KAUFMANN & al : "Mathematical Models for the Study of the Reliability of Systems", Academic Press, 1977.

KOP82 H. KOPETZ : "The Failure-Fault (FF) Model", Proceedings of the 1982 International Symposium on Fault Tolerant Computing.

LAP75 J.C. LAPRIE : "Prévision de la Sûreté de Fonctionnement et Architecture de Structure Numériques Temps réel Réparables", Thèse d'Etat, Juin 75, Toulouse.

LAP79 J.C. LAPRIE & al : "La sûreté de Fonctionnement : Besoins et Solutions", S.E.E., Congrès de Toulouse, Oct; 1979.

LAP82 J.C. LAPRIE & A. COSTES : "Dependability : A Unifying Concept for Reliable Computing", Proceedings of the 1982 International Symposium on Fault Tolerant Computing.

LEW79 D.W. LEWIS : "A Fault Tolerant Clock Using Standby Sparing" Proceedings of the 1979 International Symposium on Fault Tolerant Computing.

LIV78 C. LIVERCY : "Théorie des Programmes", Dunod Informatique, 1978.

LOS76 J. LOSQ : "A Highly Efficient Redundancy Scheme : Self Purging Redundancy", IEEE Transactions on Computers, Vol C-25, n°6, Juin76.

LOS78 J. LOSQ : "Enumeration of the Critical Fault Patterns in Fault Tolerant Computer Systems", Proceedings of the 1978 International Symposium on Fault Tolerant Computing.

MAC79 S.R. MAC CONNEL & al : "The Measurement and Analysis of Transient Errors in Digital Computer Systems", Proceedings of the 1979 International Symposium on Fault Tolerant Computing

MAT71 F.P. MATHUR : "On Reliability Modeling and Analysis of Ultra-Reliable Fault Tolerant Digital Systems", IEEE Transactions on Computers VolC20, n°11, Nov71.

MER76 C. MERAUD & al : "Automatic Rollback Techniques of the COPRA Computer", Proceedings of the 1976 International Symposium on Fault Tolerant Computing

MIN67 H. MINE & Y. KOGA : "Basic P-operties and a Construction Method for Fail Safe Logical Systems", IEEE Transactions on Electronic Computers, Vol EC15, n°3, Juin67

MOA81 M. MOALLA : "Spécifications et Conception Sûre d'Automatismes Discrets Complexes Basées sur l'utilisation du Grafcet et des Réseaux de Petri", Thèse d'Etat, 1981, INP Grenoble.

MOR76 J. MOREIRA DE SOUSA & al : "A Research Oriented Microcomputer with Built in Auto Diagnostics", Proceedings of the 1976 International Symposium on Fault Tolerant Computing.

PET72 W. PETERSON : "Error Correcting Codes", MIT Press, Cambridge, 1972

PIL82 D. PILAUD : "Méthode de Conception Descendante de Systèmes Temps Réel", Thèse de 3ème Cycle, Nov1982, INP Grenoble

PUL79 J. PULOU : "Expression de la synchronisation dans les Systèmes Informatiques et Conception d'Architectures Tolérant les Pannes", Thèse de Docteur Ingénieur, Sept79, INP Grenoble

ROB79 C. ROBACH : "Test et Testabilité des Systèmes Informatiques", Thèse d'Etat, Juin 1979, INP Grenoble

ROH73 J.A. ROHR : "Starex Self Repair Routines : Software Recovery in the JPL Computer", Proceedings of the 1973 International Symposium on Fault Tolerant Computing.

SHE75 J.J. SHEDLETSKY & E.J. MC CLUSKEY : "The Error Latency of a Fault in a Combinational Digital Circuit", Proceedings of the 1975 International Symposium on Fault Tolerant Computing.

SHE76 J.J. SHEDLETSKY & E.J. MC CLUSKEY : "The Error Latency of a Fault in a Sequential Digital Circuit", IEEE Transactions on Computers Vol C25, n°6, Juin 1976.

SIE78 D.P. SIEWIOREK & al : "A Case of Study of C.mmp, Cm and CVMP : Part I- Experiences with Fault Tolerance in Multiprocessor Systems", Proceedings of the IEEE, Vol66, n°10, Oct 1978

STI77 J.J. STIFFLER : "Coding for Random Access Memory", Proceedings of the 1977 International Symposium on Fault Tolerant Computing.

TRY J.G. TRYON : "Quadded Logic", Redundancy Techniques for Computing Systems, Wilcox & Mann eds, Spartan Book.

WAK76 J.F. WAKERLY : "Microcomputer Reliability Improvement Using Triple Modular Redundancy", Proceedings of the IEEE, Vol64, n°6, Juin76.

WEI80 C.B. WEINSTOCK : "Sift : System Design and Implementation", Proceedings of the 1980 International Symposium on Fault Tolerance Computing.

WEN78 J.H. WENSLEY & al : "Sift : Design and Analysis of a Fault Tolerant Computer for Aircraft Control", Proceedings of the IEEE, Vol66, n°11, Oct78.

YOU74 J.W. YOUNG : "A First Order Approximation to the Optimum Checkpoint Interval", Communications of the ACM, Vol17, n°9, Sept74

A U T O R I S A T I O N D E S O U T E N A N C E

VU les dispositions de l'article 3 de l'arrêté du 16 avril 1974,

VU les rapports de présentation de

. Madame SAUCIER, Professeur

. Monsieur LAPRIE, Maître de recherche

Monsieur Eric PILAUD

est autorisé à présenter une thèse en soutenance pour l'obtention du diplôme de DOCTEUR-INGENIEUR, spécialité "Génie informatique".

Fait à Grenoble, le 12 novembre 1982

Le Président de l'I.N.P.-G.

D. BLOCH
Président
de l'Institut National Polytechnique
de Grenoble

P.O. le Vice-Président,

RESUME :

Un système de description des calculateurs à haute sûreté de fonctionnement est proposé et appliqué à des calculateurs existants. Cette description nous conduit à proposer une démarche de conception devant faciliter la validation.
Une méthode d'évaluation de la sûreté de fonctionnement, destinée à fournir des évaluations pour la certification, est ensuite étudiée. Elle s'appuie sur une démarche non optimiste et permet de prendre en compte certains paramètres difficilement quantifiables (latence d'erreur par exemple) les deux aspects de cette étude, conception et évaluation, sont appliqués à un calculateur tolérant les pannes développé dans le cadre de ce travail (CARL)

MOTS CLES :

SURETE DE FONCTIONNEMENT, TOLERANCE AUX PANNES, FIABILITE, SECURITE.