

N°d'ordre : 996

50376
1992
260

50376
1992
260

A RETOURNER

présentée à

**L'UNIVERSITE DES SCIENCES ET TECHNOLOGIES
DE LILLE**

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITE

en

**PRODUCTIQUE,
AUTOMATIQUE ET INFORMATIQUE INDUSTRIELLE**

par



Abdoul Karim Armand TOGUYENI

Ingénieur IDN



**SURVEILLANCE ET DIAGNOSTIC EN LIGNE DANS
LES ATELIERS FLEXIBLES DE L'INDUSTRIE
MANUFACTURIERE**

Soutenue le 12 novembre 1992 devant la Commission d'Examen

Membres du jury :

M. VALETTE

Rapporteur, President

P. MILLOT

Rapporteur

M. STAROSWIECKI

Rapporteur

D. CORBEEL

Examineur

E. CRAYE

Examineur

J.C. GENTINA

Examineur, Directeur de Thèse

X. DATIN

Invité

Directeurs de travail E. CHAYE

Thèse préparée dans le cadre de LAI, COPEPSA, BOULE CENTRALE DE LILLE

A ma famille, à mes amis ...

AVANT-PROPOS

Le travail présenté dans ce mémoire a été effectué au Laboratoire d'Informatique Industrielle de Lille (L.A.I.L.) sous la direction scientifique de Monsieur GENTINA, Professeur à Centrale Lille, avec la collaboration de Monsieur CRAVE, Maître de Conférences à Centrale Lille. Je voudrais leur témoigner toute ma reconnaissance pour l'aide tant scientifique que morale qu'ils ont su m'apporter tout au long de ce travail.

Je tiens également à remercier :

Monsieur CORBEEL, Maître de Conférences à Centrale Lille, spécialiste des problèmes en Intelligence Artificielle,

Monsieur MILLOT, Professeur à l'Université de Valenciennes (UVHC), spécialiste des problèmes de supervision,

Monsieur STAROSWIECKI, Professeur à l'Université de Lille (USTL), spécialiste des problèmes de surveillances des systèmes complexes,

Monsieur VALETTE, Directeur de Recherche CNRS au LAAS de Toulouse, animateur du GT2 (commande et surveillance) du pôle SED du GR automatique.

De l'honneur qu'ils me font en acceptant de juger ce travail.

Je suis également flatté de la présence à ce Jury de Monsieur DATUN, Responsable de la Division Recherche et Développement de la Société Télémécanique, et je tiens tout particulièrement à l'en remercier.

Qu'il me soit permis ici de rendre hommage à tous les membres du L.A.I.L., pour l'aide précieuse qu'ils m'ont apportée tant sur le plan scientifique que sur le plan humain. Permettez moi de remercier individuellement :

- Monsieur BOUREY, Maître de Conférences à Centrale Lille , qui m'a aidé dans ma connaissance des API et des réseaux locaux,*
- Monsieur YIM, Maître de Conférences à Centrale Lille, pour l'aide qu'il m'a apporté dans la formalisation du positionnement des capteurs pour le diagnostic en ligne,*
- S. ELKHATTABI et A. FARAH mes collègues de l'équipe surveillance.*

Je voudrais également remercier Monsieur DEPARCY, et l'ensemble du département de Mécanique de Centrale Lille, pour l'aide précieuse qu'ils m'ont apportée sur la connaissance des machines de production utilisées dans l'industrie manufacturière.

Je remercie également Madame DUPLOUICH et ses collaboratrices du Centre de Documentation d'EC Lille, pour l'aide précieuse qu'elles m'ont apporté dans mes recherches bibliographiques. Enfin, je remercie très sincèrement Monsieur VANGREVENINGE et Monsieur GERMOND qui ont assuré la reprographie de ce mémoire.

SOMMAIRE

ABREVIATIONS	Page 11
INTRODUCTION GENERALE	Page 15
CHAPITRE I	Page 19
CHAPITRE II	Page 115
CHAPITRE III	Page 159
CHAPITRE IV	Page 245
CONCLUSION GENERALE	Page 305
ANNEXES	Page 309
BIBLIOGRAPHIE	Page 327
TABLE DES MATIERES	Page 343
TABLE DES FIGURES	Page 357

ABREVIATIONS

AA : Arrêt des Avances
AED : Arrêt dans un Etat Déterminé
AFC : Arrêt en Fin de Cycle
AMDE : Analyse des Modes de Défaillance et de leurs Effets
API : Automate Programmable Industriel
ATE Automatic Test Equipment
AU : Arrêt d'Urgence
CASPAIM : Conception Assistée de Système de Production Automatisée en Industrie Manufacturière
EPO : Élément de Partie Opérative
GF : Graphe Fonctionnel
MAC : Méthode de l'Arbre de Cause
MCPR : La Méthode des Combinaisons des Pannes Résumées
MDCC : Méthode du Diagramme Causes-Conséquences
MDS : Méthode du Diagramme de Succès
MDT : Mean Down Time (durée moyenne d'indisponibilité)
MEE : Méthode de l'Espace des Etats
MTBF : Mean Time Between Failure (temps moyen de bon fonctionnement)
MTTF : Mean Time To Failure (temps moyen avant la première défaillance)
MOCN : Machine Outil à Commande Numérique
OBSD : ensemble des noeuds Directement OBServables
OCE : Objet Commandable Élémentaire
OSI : Open System Interconnection
PC : Partie Commande
PO : Partie Opérative
RdP : Réseaux de Petri
RdPI : Réseaux de Petri Interprétés
RdPO : Réseaux de Petri à Objets
RdPSAC : Réseaux de Petri Structurés Adaptatifs Colorés
SA : Système Automatisé
SED : Système à Evénements Discrets
SFP : Système Flexible de Production
SFPM : Système Flexible de Production Manufacturière
STC : Signature Temporelle Causale

INTRODUCTION GENERALE

L'automatisation de la production a permis des gains de productivité considérable dans le cadre de l'Industrie manufacturière. Ils ont été rendus possibles par l'intérêt conjoint porté d'une part à la conception du système de commande et d'autre part à la prise en compte d'une politique de surveillance et de maintenance de plus en plus fine. La complexité des systèmes automatisés s'est fortement accrue au cours des dernières années par la flexibilité progressivement conférée aux Systèmes Flexibles de Production Manufacturière (SFPM).

Au début de l'automatisation, la maintenance a souvent été assurée par les opérateurs de supervision. Pour tenir compte de la complexité des systèmes actuels, il a fallu créer de véritables équipes de maintenance. Aujourd'hui, la maîtrise de la maintenance est devenue un des facteurs essentiel du gain en productivité [BRU90] et de la poursuite de l'automatisation. Pour être efficace, la maintenance doit être rapide. L'introduction au sein de la commande d'une fonction de surveillance assurant l'identification et le confinement rapide des défaillances, permet aujourd'hui de tendre vers la réalisation de ces objectifs.

Au Laboratoire d'Automatique et d'Informatique Industrielle de LILLE (L.A.I.L.), nous proposons d'intégrer la conception de la surveillance à la méthodologie de conception des SFPM. Les travaux présentés dans ce mémoire, font partie intégrante du projet CASPAIM (Conception Assistée de Système de Production Automatisée en Industrie Manufacturière). Ils se situent notamment en amont des travaux présentés par S. Bois [BOI91] dans le cadre de la gestion des modes de marches des moyens de production d'un atelier.

Les résultats présentés dans ce mémoire, concernent la surveillance du procédé. Nous avons choisi dans ce sens une approche de surveillance séparée [SAH87] basée sur l'utilisation d'outils d'intelligence artificielle.

La présentation du mémoire est divisée en quatre chapitres.

Le premier chapitre concerne **"La surveillance en ligne : Principe et Structuration"**. Il sert d'introduction et aboutit à une proposition de structuration de la surveillance en ligne des SFPM en trois fonctions : la fonction détection, la fonction diagnostic et la fonction recouvrement. Dans ce chapitre, nous proposons également une méthode de conception de la fonction détection. La méthode proposée s'appuie sur la définition d'un modèle temporel générique pour chaque opération. L'utilisation des Réseaux de Petri temporels à Objets [CAR90] est ici proposée en tant qu'un outil de spécification du mécanisme de détection.

Au cours du deuxième chapitre nous abordons les questions relatives au **diagnostic en ligne**. Notre objectif ici est d'aboutir à l'**expression des besoins** pour cette forme de diagnostic. Les limites des systèmes experts classiques utilisés comme outils pour le diagnostic en ligne y sont envisagées. Deux principales contraintes sont évoquées : la **contrainte temporelle** et la nécessité de **complétude**. Nous aboutissons à l'idée qu'un raisonnement qualitatif basé sur une modélisation fonctionnelle du procédé est un bon moyen de conception du diagnostic en ligne.

Les points de vue **Modélisation et Raisonnement** dans le contexte du diagnostic sont présentés au chapitre III. La première partie de ce chapitre, est consacrée à la présentation d'une méthode de modélisation fonctionnelle du procédé. Cette modélisation concerne trois aspects respectivement : structurel, fonctionnel et comportemental [SEM86]. Nous obtenons ainsi deux modèles : un modèle fonctionnel et un modèle structurel. Le modèle fonctionnel précise comment les fonctions principales¹ du système sont créées à partir des fonctions initiatrices². Le modèle structurel explicite les composants utilisés pour la réalisation des fonctions.

Au niveau du diagnostic, nous distinguons trois traitements de base :

- la **localisation** ou **diagnostic local** qui a pour but de déterminer les sous-ensembles défectueux,
- l'**identification** ou **diagnostic global** qui a pour but de déterminer l'origine fonctionnelle d'une défaillance,
- le **pronostic** qui a pour but de propager dans le modèle les prédictions sur les conséquences d'une défaillance, afin d'anticiper et donc d'éviter les effets des défaillances.

La localisation et l'identification sont respectivement basées sur le raisonnement temporel et le raisonnement hypothétique.

Le chapitre IV concerne la "**Spécification fonctionnelle de la commande d'un atelier flexible, dans le cadre de l'intégration des mécanismes de recouvrement des défaillances du procédé**". Il résume une démarche plus prospective dans laquelle nous illustrons l'idée d'une conception conjointe de la commande et de la surveillance. La décomposition en couches fonctionnelles permettra ici de concilier un objectif de cohérence dans la conception de systèmes de commande et de surveillance, réellement indépendants, au sens de la structuration et de la modularité.

¹ Ce sont des fonctions perceptibles par l'environnement du système. On peut les identifier à des services.

² Ce sont les fonctions les plus élémentaires du modèle fonctionnel.

CHAPITRE I

LA SURVEILLANCE EN LIGNE : PRINCIPE ET STRUCTURATION

SOMMAIRE

I - Positionnement et contexte du problème,

II - Choix d'une politique de surveillance adaptée à la maintenance des SFP,

III - Structuration de la surveillance dans le contexte des ateliers flexibles,

IV - Détection.

INTRODUCTION

La surveillance n'est pas un concept nouveau. Il est apparu très tôt avec le développement de l'aéronautique et du nucléaire. Que représente ce concept pour le profane ? Lorsqu'on est dans une voiture, on peut observer un certain nombre d'indicateurs sur le tableau de bord : indicateur du niveau d'essence, indicateur du niveau d'huile, indicateur de la température du moteur etc ... Ces indicateurs servent à alerter le conducteur d'un fonctionnement anormal de son véhicule (cas de l'indicateur de température), ou de la dégradation d'une de ses fonctions principales (cas des indicateurs de niveau). Le tableau de bord d'un avion repose sur le même principe avec des d'indicateurs plus nombreux et plus variés, dont la fonction n'est pas toujours directement liée à la **surveillance du système physique** : c'est le cas d'un altimètre. Ce type de capteur est utilisé pour **surveiller le processus** que représente le vol de l'avion.

Dans ce contexte, l'objectif de la surveillance est d'assurer la **sécurité**. Avec le développement de l'informatique "bon marché" à partir des années 80, ce concept a été largement repris par le monde industriel sous le terme de **supervision**. Dans les **Systèmes à Événements Discrets (SED)**, l'intégration de la surveillance aux systèmes de conduite a pour but d'**augmenter la productivité** par une meilleure **maîtrise de la disponibilité des moyens de production**.

Dans ce chapitre, nous allons introduire le concept de surveillance en ligne [SAH87], [SAH92]. Il est divisé en quatre parties :

*** Positionnement et contexte du problème.**

Le but de cette partie est de situer la surveillance d'une part par rapport à l'ensemble de la méthodologie CASPAIM, et d'autre part par rapport à des concepts comme la supervision et la maintenance.

*** La surveillance en ligne.**

Dans cette partie nous allons définir ce qu'est la surveillance en ligne. Cela nécessitera une étude des approches classiques de la surveillance, notamment de celles utilisées pour les systèmes continus.

* Structuration de la surveillance dans le contexte des ateliers flexibles.

Il s'agira ici de définir les principales fonctions de la surveillance dans le contexte des ateliers flexibles ; c'est à dire l'expression des besoins pour la construction de chaque fonction.

* Détection.

A partir de la structuration précédente, nous développerons directement la fonction détection.

I. POSITIONNEMENT ET CONTEXTE DU PROBLEME.

I.1 Introduction

Notre objectif dans cette partie est de positionner la surveillance. En effet, dans la littérature, on la voit souvent assimilée à des concepts comme la supervision, la maintenance ou la sûreté de fonctionnement. Il est évident qu'elle a un lien avec tout ces concepts. Nous voulons clarifier ce lien. Avant cette étude, nous allons commencer par situer la surveillance et donc nos travaux par rapport aux autres développements de la méthodologie CASPAIM.

I.2 Place de la surveillance dans la méthodologie CASPAIM.

I.2.1 Un peu d'histoire.

La méthodologie CASPAIM s'inscrit dans l'ensemble des travaux visant à assister la conception des Systèmes Flexibles de Production (SFP) dans l'industrie manufacturière. L'idée de base de cette méthodologie réside dans une approche produit. C'est à dire que la question fondamentale posée au concepteur est la suivante : quels moyens de production doit-on faire coopérer pour obtenir les produits spécifiés par le cahier de charges ?

Au L.A.I.L.¹, les premiers travaux dans ce domaine, datent du début des années 80. Ils portaient essentiellement sur le développement de méthodes de conception de la commande de Systèmes Automatisés (SA). Ils ont conduit à l'adoption des RdPS (Réseaux de Petri Structurés) [COR79], [COR80], [BOU88], comme outils de spécification de la commande. L'intégration d'autres extensions des RdP comme les arcs adaptatifs [COR84] et la coloration [PET80], [CAS87] ont permis d'enrichir cet outil. Les différents travaux entrepris ont conduit

¹ Laboratoire d'Automatique et d'Informatique industrielle de Lille

au développement d'outils informatiques pour la conception assistée de la commande d'un SFP.

Le premier de ces outils concerne la génération d'un modèle agrégé de la commande à partir d'une spécification du cahier de charges sous forme de règles de production [KAP88]. Le développement de cet outil a été basé sur l'exploitation de techniques d'intelligence artificielle.

Le deuxième outil a pour point d'entrée le modèle agrégé ou **prégraphe**. C'est un outil qui a pour but le développement et la structuration du graphe de commande [BOU88]. Il permet par ailleurs :

- la répartition des systèmes de commande par rapport à l'architecture du procédé,
- et la détection des indéterminismes du graphe de commande.

Parallèlement aux travaux sur la conception de la commande, ont été développés des travaux pour sa validation et sa mise en œuvre sur site.

Au niveau validation, les premiers travaux ont porté sur la conception d'un simulateur de **RdPSAC** (Réseaux de Petri Structurés Adaptatifs Colorés) [CAS87] permettant de valider le graphe développé. Par la suite, d'autres travaux ont permis l'utilisation des **réseaux de Petri temporisés à capacités** pour la pré-validation du prégraphe [HEI87], [SEY89].

Les travaux pour la mise en œuvre pratique de la commande, concernaient le transfert et la répartition du graphe de commande sur des API ou Automates Programmables Industriels [CRA89]. Des règles de passage des RdPSAC au grafcet ont été établies [TOG88], [CRA89].

Plus récemment, des travaux ont porté sur l'intégration de la gestion des modes de marches et d'arrêts des moyens de production à l'ensemble de la commande [BOI91/b]. Nous reviendrons sur ces travaux dans le chapitre sur le recouvrement (chapitre IV).

L'ensemble des travaux précités, s'intègrent à une première phase de la méthodologie CASPAIM ; phase baptisée **CASPAIM-1**.

En effet, depuis le début des années 90, cette méthodologie s'est enrichie pour faire face au développement de moyens de production de plus en plus complexes (Stockeurs Rotatifs, Robots Portiques), et à l'introduction de techniques de gestion des produits de plus en plus sophistiquées (gestion par lots, notion de dates dues).

L'évolution de la méthodologie s'explique également par l'introduction d'une nouvelle extension des RdP : les **RdP à objets** ou **RdPO** [SIB88], [SIB90]. L'intérêt principal de cet outil est qu'il permet de passer d'une gestion des produits par classe (contexte d'utilisation de la coloration) à une individualisation de chaque objet

Les travaux actuels s'inscrivent donc dans une nouvelle phase appelée **CASPAIM-2**.

I.2.2 Présentation des principaux travaux de CASPAIM II.

I.2.2.1 Elaboration de la commande de coordination [CRU91].

I.2.2.1.1 Principe

Une des idées essentielles de CASPAIM-2 réside dans la séparation au niveau de la conception de la commande, de ce qui relève du produit et de ce qui relève des moyens de production. D. Cruette a illustré cette nécessité à partir de l'exemple du transport de pièces par un ascenseur.

Nous supposons que nous avons un système de production réparti sur quatre étages desservis par un ascenseur. L'ascenseur a une capacité maximale de 10 pièces. Nous disposons par ailleurs de 2 types de pièces :

- les pièces1 qui doivent être transférées de l'étage1 à l'étage3,
- les pièces2 qui doivent être transférées de l'étage2 à l'étage4.

Avec CASPAIM-1, la ressource ascenseur est gérée comme un tout (ressource critique). Cela impose une stratégie unique de pilotage de l'ascenseur. Il ne peut exécuter qu'un seul des deux types de transferts à la fois. Il est aisé de se rendre compte qu'une telle stratégie n'est pas optimale. Si à un moment donné, on veut transférer X pièces1 et Y pièces2 avec $X+Y \leq 10$, on est obligé de le faire en deux transferts. Pourtant la capacité de stockage de la ressource permet :

- 1 - d'aller charger les pièces1 à l'étage1,
- 2 - d'aller charger les pièces2 à l'étage2,
- 3 - de décharger les pièces1 à l'étage3,
- 4 - de décharger les pièces2 à l'étage4.

Pour résoudre ce problème D. Cruette a eu l'idée d'insérer entre la **Partie Commande (PC)** et le procédé, un module appelé **Interface**, dont le but est de gérer effectivement la commande des moyens de production. La PC s'occupe du contrôle des différentes opérations que doivent subir les produits circulant dans l'atelier. Avec la notion d'Interface, la structure d'un SFP est définie par la figure I.1.

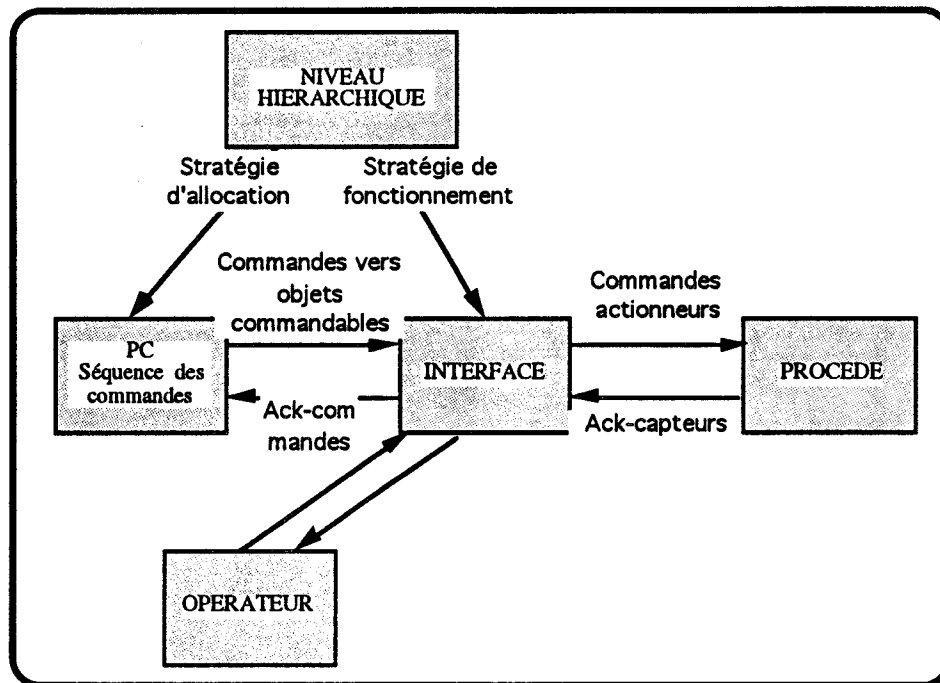


Figure I.1 : Schéma de principe d'un SFP d'après [CRU91].;

I.2.2.1.2 Les étapes de la méthodologie CASPAIM-2

La conception de la commande se déroule ainsi en parallèle selon deux axes (Fig. I.2) :

* L'axe produit.

C'est l'axe essentiel pour la génération du graphe de commande. Il comprend trois étapes :

- la spécification fonctionnelle des produits,
- la génération assistée des gammes opératoires,
- la spécification des protocoles d'accès

* L'axe procédé.

Il permet la conception de l'Interface. Les principales étapes sur cet axe sont :

- la spécification et l'analyse des moyens de production,
- la spécification des stratégies de fonctionnement.

Ensuite l'ensemble des fonctionnalités du SFP sont intégrées pour une validation fine par simulation, avant l'implantation sur site.

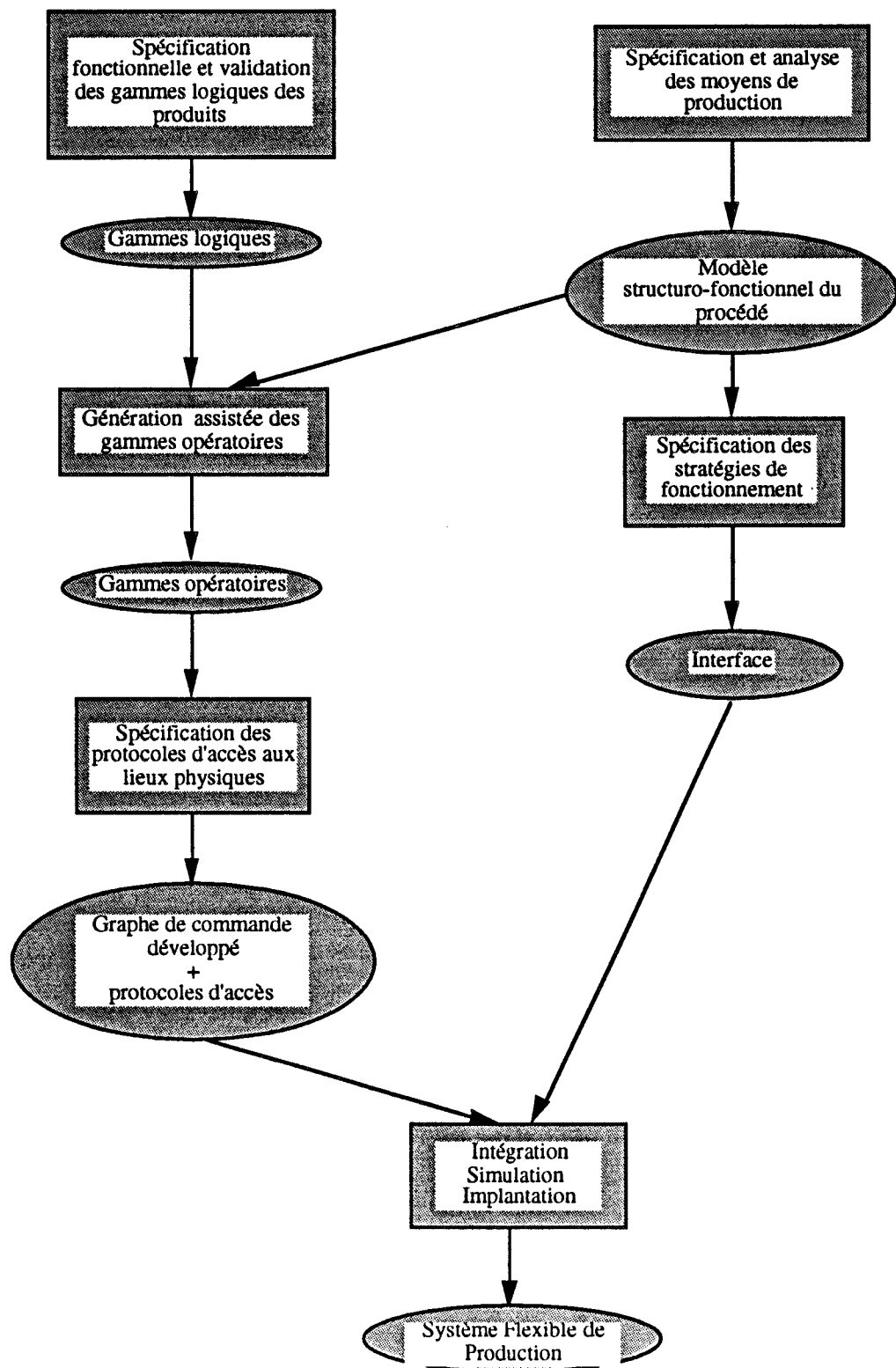


Figure I.2 : Synoptique de CASPAIM-2 d'après [CRU91]

I.2.2.1.3 Gammes logiques et gammes opératoires.

Dans une approche produit, il est avant tout nécessaire de définir les différentes opérations que doivent subir les produits pour passer de l'état brut à l'état fini. C'est ce qu'exprime la notion de gamme logique [BOU84], [KAP88].

Définition I.1 : La gamme logique [CRU91].

La gamme logique d'un produit fini P décrit la séquence et la synchronisation des fonctions qui sont appliquées à chaque produit qui entre dans la composition du produit fini P. Le modèle utilisé est le RdP où une place modélise l'état d'avancement d'un produit, et une transition une fonction qui modifie l'état d'avancement d'un produit, ou crée de nouveaux produits à partir de produits existant.

Remarque :

Il est important de noter que le terme "opération" serait plus approprié que le mot "fonction" employé dans la définition précédente. Au chapitre III, nous définirons ce qu'est une fonction.

Certaines opérations peuvent être réalisées dans un ordre indifférent. La combinatoire induite par cette flexibilité doit être traduite au niveau de la gamme : c'est la flexibilité de gamme.

Exemple I.1 :

Supposons que le produit P fini doit être Tourné (T+), Fraisé (F+) et Taraudé (Ta+). Supposons par ailleurs que le séquençement des trois opérations (t+, f+, ta+) soit indifférent. On obtient alors la gamme logique suivante :

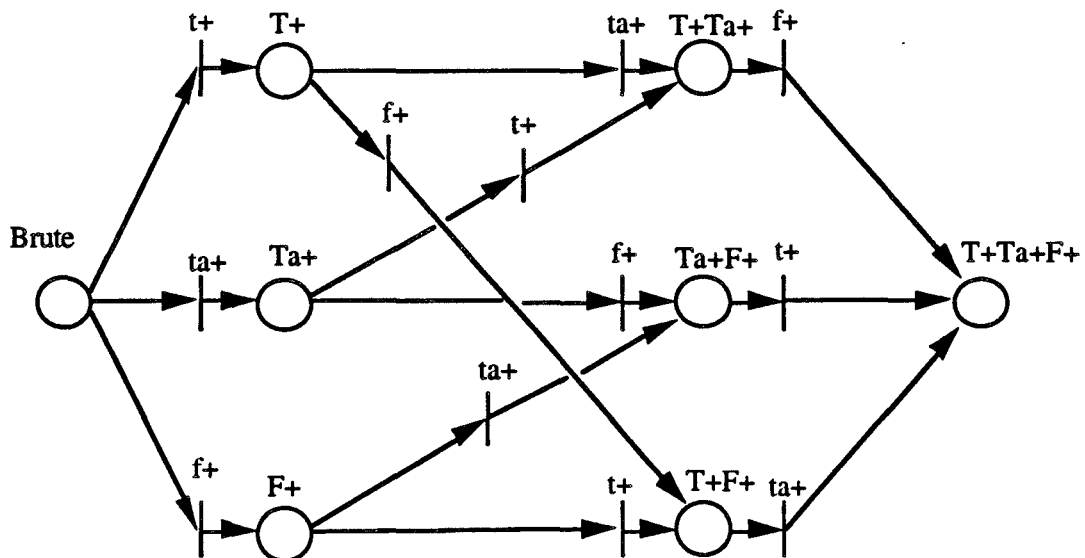


Figure I.3 : Gamme logique d'usinage avec flexibilité de gamme.

Après s'être intéressé aux étapes de la transformation des différents produits, la question est : comment les mettre en œuvre dans une architecture flexible ? Cela nécessite la prise en compte des moyens de production : c'est ce que traduit la gamme opératoire.

Définition I.2 : La gamme opératoire [CRU91].

Une gamme opératoire décrit la succession des états d'avancement d'une pièce, ainsi que la succession des lieux opératoires par lesquels elle transite. Elle associe ainsi les moyens physiques (ressources de production) aux fonctions de la gamme logique visant à l'élaboration effective des produits.

La gamme opératoire est également décrite par un RdP dans lequel les places modélisent l'état d'avancement d'une pièce sur un lieu donné, et les transitions modélisent de manière exclusive une opération de transformation ou un changement de lieu. Dans la démarche de D. Cruette, la notion de lieu est primordiale. En effet, une des hypothèses de base est la nécessité de gérer les emplacements de stockage des pièces au niveau de la PC. Ainsi pour la construction d'une gamme opératoire, on s'appuie sur la connaissance de l'ensemble des emplacements de stockage et de leur relation d'accessibilité [AMA92].

Exemple I.2 :

Supposons que nous voulions réaliser la production de pièces de type P dont la gamme logique est donnée par la figure I.3. Le système de production (Fig. I.4) comprend :

- Deux machines d'usinage multi-fonctions M1 et M2. Chaque machine possède les opérations tourner, fraiser et tarauder. M1 est munie d'un tampon d'entrée des pièces et d'un tampon de sortie. M2 est munie de deux tampons d'entrée-sortie.
- Un convoyeur à bande permettant le transfert des pièces entre la cellule considérée et le reste de l'atelier. Sur le convoyeur on dispose de deux dériviations D1 et D2 pour alimenter en pièces le poste P1 et le poste P2. P1 et P2 servent à la palettisation et la dépalettisation des pièces. Les dériviations sont actionnées par des vérins pneumatiques accrochant par le bas les palettes sur lesquelles reposent les pièces. Sur le convoyeur, un poste d'entrée permet le stockage des pièces brutes en attente.
- Un robot, R1, est utilisé pour le chargement et le déchargement des deux machines.

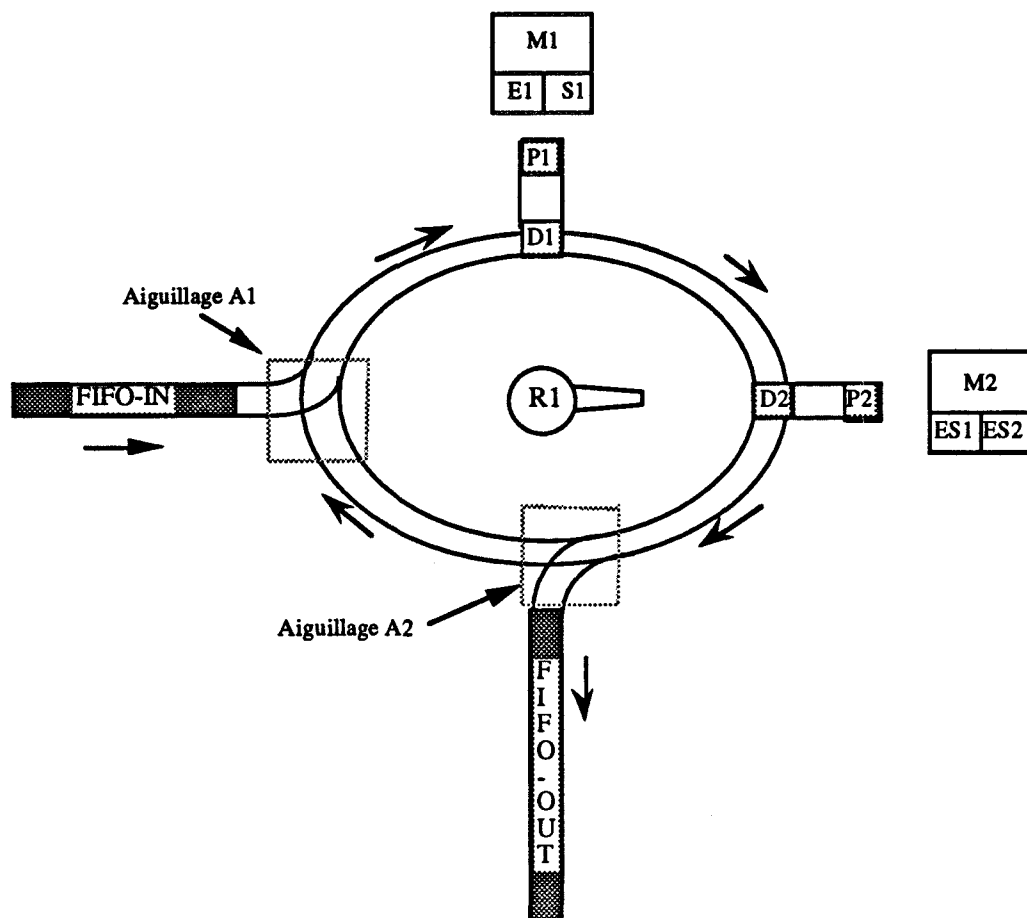


Figure I.4 : Cellule de transformation.

La flexibilité matérielle de cette cellule peut être justifiée par plusieurs raisons. D'abord dans le cadre du recouvrement des défaillances, pour permettre la gestion des modes de marche dégradés. Supposons par exemple que l'une des fonctions d'usinage soit défaillante sur l'une des deux machines. On pourra pour les pièces qui lui sont destinées, exécuter les opérations possibles et ensuite transférer la pièce sur la seconde machine pour réaliser la dernière opération. C'est un point sur lequel nous reviendrons au chapitre IV.

La flexibilité peut également se justifier pour des raisons de qualité des pièces. En effet, la précision des deux machines peut être différente pour certaines opérations. Pour un lot de pièces donné exigeant une qualité particulière, la contrainte de la qualité, associée à la contrainte temporelle exigerait alors de commencer l'usinage de certaines pièces sur l'une des machines, et de le finir sur l'autre.

A partir du procédé précédent, on établit la gamme opératoire de la figure I.5.

Pour terminer la conception de la PC, il reste à contraindre les gammes opératoires en spécifiant les protocoles d'accès aux ressources.

I.2.2.2 Conception de l'interface [CRU91], [HEB92].

Le rôle de l'interface est de déterminer parmi les requêtes de la PC celles qui sont réellement applicables au procédé. Ce tri s'explique principalement par deux raisons :

* Notion de requêtes aveugles.

La PC sollicite les services des ressources du procédé sans tenir compte de leur état réel. Par exemple, une ressource peut être active car déjà sollicitée pour un autre produit. Une ressource peut également être défaillante. Aussi, l'Interface doit commencer par contrôler si la requête émise est conforme à l'état du procédé. Pour cela, l'Interface a besoin d'exploiter un modèle dynamique du procédé. La notion de modèle dynamique utilisée par D. Cruette est identique à celle de S. Amar [AMA90] que nous présenterons au chapitre II (cf. Chapitre II § IV.2.3.1).

* Stratégie de fonctionnement.

Comme illustré par l'exemple de l'ascenseur une même ressource peut être sollicitée par plusieurs requêtes concurrentes. Dans quel ordre les réaliser ? Dans ce contexte, on a besoin de définir des stratégies de fonctionnement de la ressource. Ces stratégies vont dépendre de l'état de la ressource et également des requêtes en compétition. Elles sont paramétrées par le niveau hiérarchique.

Par exemple, dans le cas de l'ascenseur, la stratégie suivante peut être envisageable :

Si l'ascenseur est positionné à l'étage1 et que X (le nombre d'emplacements alloués aux pièces1) est supérieur ou égal à 7, alors aller directement à l'étage3. On voit que les paramètres fixant la stratégie ici seront la position de l'ascenseur, les valeurs relatives de X et Y et également des priorités de production.

I.2.3 Conclusion

La présentation des principaux travaux de la méthodologie CASPAIM, montre que dans un premier temps nous nous sommes exclusivement intéressés à la conception de la commande du mode normal de fonctionnement d'un atelier. La conception et la gestion des modes dégradés nécessitent de connaître, de détecter et d'identifier les dysfonctionnements des différents acteurs d'un SFP. C'est la fonction dédiée à la surveillance qui classiquement est vue comme une sous-fonction du niveau hiérarchique (Fig. I.6).

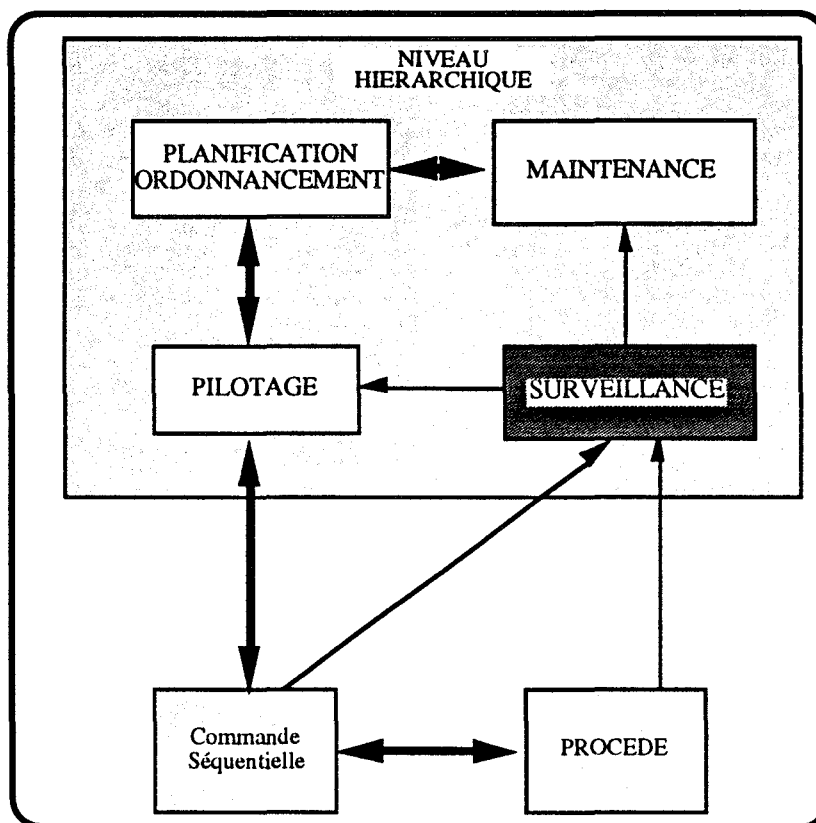


Figure I.6 : Structuration du niveau hiérarchique pour l'intégration de la surveillance.

D'après E. Niel [NIE92], "il est important de tenir compte de la surveillance le plus tôt possible, dans le processus de conception, afin de minimiser les risques et les coûts". En effet,

l'expérience acquise lors de la conception des premiers SA a montré que l'intégration de fonctions de sûreté dès la spécification induit un coût d'environ 10% du coût total. Par contre l'ajout de fonctions de sûreté à posteriori, représente approximativement 70% du coût total d'une installation. D'où le choix dans la méthodologie CASPAIM, de mener en parallèle la conception de la commande, du procédé et de la surveillance. Nous verrons notamment dans la suite de ce chapitre, que la conception du système de surveillance hérite des études préliminaires de sûreté de fonctionnement nécessaires au choix de la structure matérielle de l'atelier. Nous verrons également au chapitre IV, comment les choix dus à la surveillance vont influencer sur la conception de la commande.

I.3 Surveillance et supervision.

Avec le développement des micro-ordinateurs, on a vu fleurir un certain nombre de logiciels qualifiés de "systèmes de supervision industriels". Aussi, dans cette partie nous proposons de clarifier les concepts de supervision et de surveillance.

I.3.1 Systèmes de supervision des processus continus.

D'après [MIL88], la supervision des processus continus permet d'assurer deux fonctions essentielles : **la conduite et la surveillance**. En fait, dans les approches classiques, ces deux fonctions ne sont pas dissociées. La fonction de base est la conduite. Cette activité, trop "complexe" pour être assurée par un calculateur, est confiée à un opérateur humain.

La conduite consiste à partir de la connaissance des valeurs des variables caractéristiques de l'état du SA, à déterminer les consignes à appliquer pour se rapprocher du fonctionnement optimal. La conduite induit donc des prises de décision par l'opérateur. Cette activité de conduite oblige l'opérateur à connaître à tout moment l'état du système. De ce fait lorsqu'il y a un dysfonctionnement, il peut décider des actions à mener pour y remédier. En fait, dans cette situation son but est d'empêcher le système de s'écarter de son point de fonctionnement optimal. C'est la raison pour laquelle, la surveillance est une activité implicite et indissociable de la conduite.

Pour faciliter la supervision, on a eu l'idée d'associer des alarmes aux principales variables du SA, afin d'en détecter la dérive et de focaliser l'attention de l'opérateur sur les

paramètres importants intervenant dans la prise de décision. Rasmussen a proposé le modèle de la figure I.7 qui décrit les étapes de la procédure de décision d'un opérateur humain.

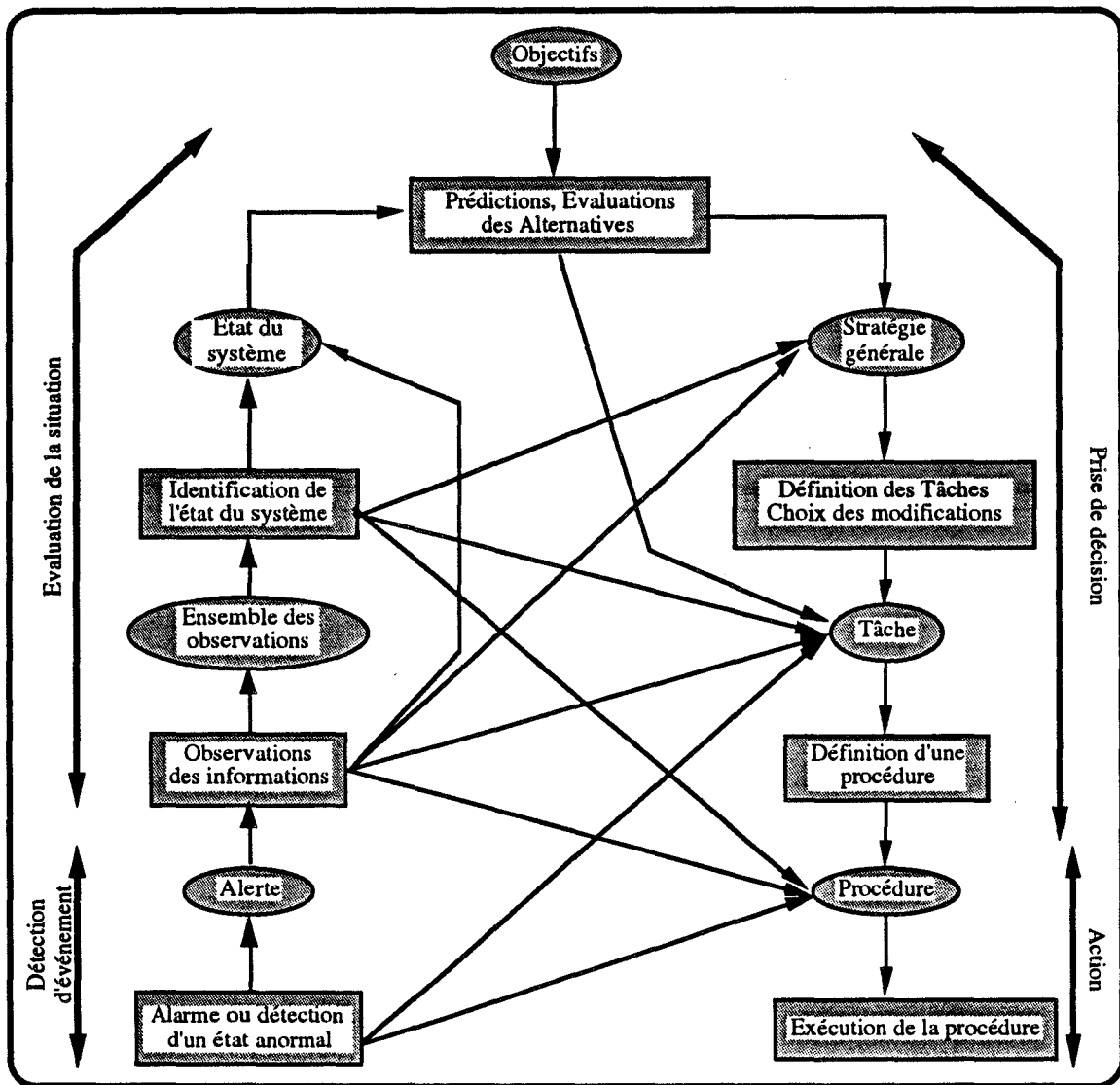


Figure I.7 : Modèle de résolution de problème adapté de Rasmussen [MIL88].

Malgré l'apprentissage de règles de prise de décision systématique permettant de sauter des étapes dans des situations classiques de supervision, on a pu constater un grand nombre d'accidents. Ainsi, un accident comme celui de la centrale nucléaire de Three Miles Island [VIL88] est dû à une erreur d'appréciation des opérateurs.

L'analyse du comportement des opérateurs humains a permis de montrer que ces erreurs incombent peu à des problèmes de compétence. Elles sont souvent dues à une surcharge en

alarmes, amenant les opérateurs à s'occuper parfois de problèmes mineurs pendant qu'apparaissent des défaillances majeures. Les problèmes sont également dus à la lassitude des opérateurs devant la génération d'un grand nombre de fausses alarmes. Du coup, leur vigilance se relâche, entraînant un temps de réaction supérieur.

Face à ces constats, les recherches ont été orientées vers une coopération homme-calculateur pour améliorer la supervision.

I.3.2 Coopération homme-calculateur [MIL88].

Il s'agit en fait d'assister certaines tâches de l'opérateur humain par le calculateur. On met alors à la disposition de l'opérateur humain des fonctions d'aide à la décision. Cette coopération peut se faire selon deux approches : l'approche verticale et l'approche horizontale.

I.3.2.1 Coopération verticale

Dans ce type de coopération toutes les décisions sont prises par l'opérateur. Toutefois, ce dernier dispose d'un certain nombre d'outils d'aide à la décision auxquels il peut faire appel en cas de besoin. Ces outils sont en général des outils de **détection**, de **classification des alarmes**, de **diagnostic**, et de **reprise** après dysfonctionnements. La classification des alarmes consiste en fait en un filtrage et une hiérarchisation. Cela est surtout valable en cas d'occurrence de défaillances dépendantes de cause commune. La classification des alarmes est un des outils les plus appréciés des opérateurs. P. Millot citant Rijnsdorf, indique que seulement 10 % des alarmes sont exploités par les opérateurs dans les systèmes de supervision classiques [MIL88].

Le principal reproche que nous ferons à la coopération verticale est le fait qu'elle ne soit pas systématique.

I.3.2.2 Coopération horizontale

La coopération horizontale consiste à partager les mêmes fonctions entre l'opérateur et le calculateur. Il ne s'agit donc pas de mettre en place une technique de redondance permettant un contrôle mutuel entre ces deux agents de la supervision. En fait, l'idée est de décharger l'opérateur de certaines tâches de surveillance, dans un **contexte de surcharge**. Il y a alors **répartition dynamique** de l'activité de surveillance (notamment les fonctions de décision)

entre l'opérateur et le calculateur qui agissent en parallèle sur des entités indépendantes du procédé.

Comment est évaluée une situation de surcharge ? Sur quels critères s'effectue la répartition ?

Il existe deux formes de répartition :

- la répartition explicite,
- la répartition implicite.

La répartition explicite a pour agent ... l'opérateur. Bien entendu, cela peut constituer une charge supplémentaire pour lui. D'où l'idée de confier au calculateur le soin d'évaluer en temps réel la charge de supervision, et d'en effectuer la répartition. C'est l'objet de la répartition implicite.

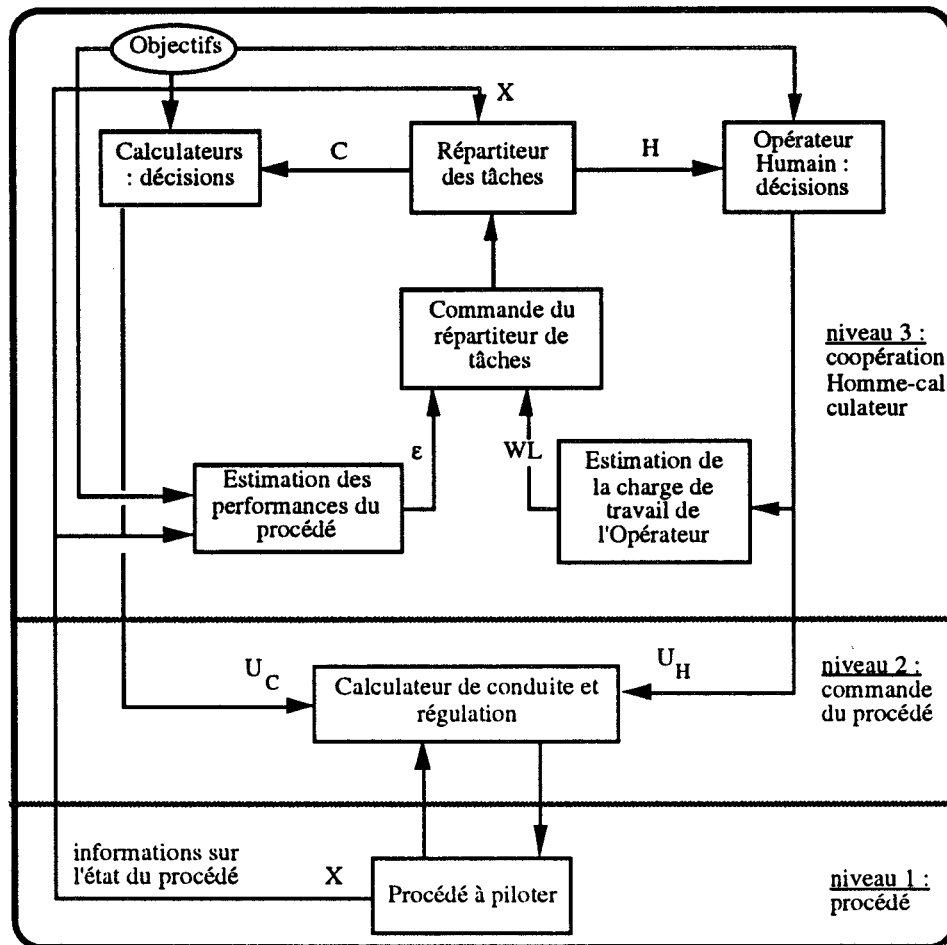


Figure I.8 : Répartition dynamique des tâches entre opérateur et calculateur d'après P. MILLOT [MIL88]

I.3.3 Conclusion

Cette étude montre que la supervision ne doit plus être perçue comme un simple système de visualisation de l'état du procédé. Elle doit être conçue comme un ensemble de **fonctions intelligentes** permettant d'assister l'opérateur humain dans sa fonction de conduite-surveillance.

Dans le cadre de l'automatisation des procédés discrets de l'industrie manufacturière, il y a bien longtemps que les fonctions de conduite (ou contrôle) et de surveillance ont été séparées. Comme nous l'avons souligné dans la partie précédente (§ I.1), il existe des méthodologies (CASPAIM en est un exemple d'illustration), permettant l'automatisation d'une grande partie de la fonction de conduite. Nous pensons qu'il est possible de proposer une démarche analogue pour la fonction surveillance. Notre approche de la surveillance ne se situe plus en terme de coopération entre opérateur et calculateur. L'idée est plutôt de remplacer l'opérateur par un, voir plusieurs calculateurs hiérarchiquement répartis. De prime abord, cela peut sembler une gageure. Nous avons pourtant la conviction que cette solution s'impose comme une nécessité. En effet, d'un point de vue conduite et surveillance, les SED sont plus simple à appréhender que les systèmes continus. Les mécanismes de contrôle-surveillance y sont plus dissociés. Par contre, dans les SED, la contrainte temporelle est plus forte et la qualité de fabrication est un impératif. Une opération de perçage n'a pas la même échelle de temps que la montée en température d'un four. Les vitesses d'avance sur une M.O.C.N.² sont souvent très importantes. Une défaillance classique des tours à axe horizontal, et la panne du système de régulation de l'axe Z (axe de broche). Dans ce contexte, l'arrêt des avances doit être instantané afin d'éviter que l'outil aille s'encaster dans la broche.

Les deux contraintes précédentes (temporelle et qualité) expliquent la nécessité de se passer autant que possible de l'intervention d'un opérateur. Pour nous résumer, disons que dans les SED, la surveillance est au delà du problème d'une "simple" classification d'alarmes. Elle doit permettre une **réaction temps réel** sur le SA au moindre signe de défaillance.

² Machine Outil à Commande Numérique

I.4 Surveillance et maintenance.

I.4.1 Introduction.

Au sens strict du terme, la maintenance regroupe l'ensemble des actions permettant de maintenir ou rétablir un système (ou un processus) dans un état donné, en lui restituant les caractéristiques de fonctionnement spécifiées.

Cette définition montre que le concept de maintenance est lié à la notion de défaillance. Quels sont alors les liens entre la maintenance et la surveillance ?

I.4.2 La sûreté de fonctionnement.

I.4.2.1 Introduction

La sûreté de fonctionnement a pour objectif d'établir le degré de confiance que l'on peut attribuer à un système dans le cadre de la mission qu'il doit assurer. Ce degré de confiance est donné par les concepts suivants :

- fiabilité,
- disponibilité,
- maintenabilité,
- sécurité.

Dans l'annexe A, nous donnons les définitions formelles et mathématiques de ces concepts, ainsi que celles des principaux paramètres associés. Rappelons ici, une hypothèse de base de nos travaux : les systèmes auxquels nous nous intéressons sont des **systèmes réparables et cohérents**.

Comme la sûreté de fonctionnement, la surveillance a pour but de garantir notamment la sécurité et la disponibilité des moyens de production. La seule différence se situe au niveau où interviennent ces deux concepts au sein du cycle de vie d'un système automatisé. La sûreté de fonctionnement intervient dès la spécification. Elle permet de choisir les systèmes élémentaires et les composants les mieux adaptés au degré de confiance désiré. Par contre, la surveillance intervient en phase d'exploitation. Nous pouvons donc la définir comme **la sûreté de fonctionnement opérationnelle**.

I.4.2.2 Terminologie

Avant d'aller plus loin, il est important de préciser la définition que nous donnons à certains termes couramment utilisés. Il s'agit entre autres des termes **“panne, défaillance, défaut, erreur, faute et mode de défaillance”**.

Pendant de nombreuses années la communauté scientifique a utilisé ces termes sans qu'il y ait de réel consensus sur leur signification. C'est de la communauté informatique que sont parties les premières tentatives de "normalisation" [LAP84], [LAP85]. Dans le cadre des systèmes industriels mécaniques, nous emprunterons à A. VILLEMEUR les définitions suivantes :

Définition I.3 : La défaillance [CEI74].

C'est la cessation de l'aptitude d'une entité (composant ou système) à assurer une fonction requise, dans des conditions données.

Il est nécessaire d'indiquer les conditions dans lesquelles une entité est supposée fonctionner. Cela peut être des conditions physiques (plage de températures, plage d'efforts, ...) ou simplement des conditions qualitatives généralement en rapport avec le maintien de la sécurité.

Prenons l'exemple du roulement de broche. Il a deux fonctions :

- assurer la transmission du mouvement de rotation,
- supporter les contraintes émises sur la broche (chocs ...).

Lorsqu'il est usé, le roulement émet un bruit perceptible à l'oreille humaine. Dans quelle mesure peut-on dire ici qu'il y a défaillance ? S'il a été explicitement spécifié dans le cahier de charges que le fonctionnement doit se dérouler en absence totale de bruit, alors il y a défaillance dès que le bruit est détecté. Autrement, il n'y a défaillance que si l'une des deux fonctions, énumérées précédemment, n'est plus remplie.

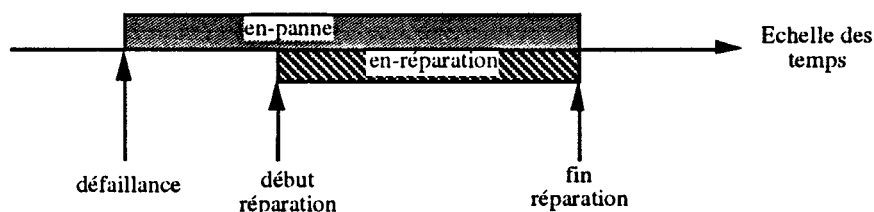


Figure I.9 : Chronogramme liant les concepts de défaillance, de panne et de réparation.

Définition I.4 : La panne [VIL88].

C'est l'inaptitude d'une entité (composant ou système) à assurer une fonction requise.

Si nous écartons la possibilité d'erreurs de conception, la définition précédente implique que toute défaillance entraîne une panne. La défaillance correspond à un événement et la panne à un état. Sur le plan temporel, la défaillance correspond à une date et la panne à une durée comprise entre la date d'occurrence de la défaillance et la date de fin de réparation (Fig. I.9).

Dans le cas d'un roulement, l'état de panne se manifestera souvent par le blocage de la rotation ou même par sa rupture.

Définition I.5 : Un défaut.

C'est l'écart entre une caractéristique de l'entité et la caractéristique voulue.

Toute panne entraîne un défaut. Par contre il peut y avoir défaut sans panne effective. C'est le cas des défauts de conception qui n'apparaissent pas lors du test de l'entité.

Le mode de défaillance est un concept moins classique pour les profanes. Toutefois, il est essentiel pour la compréhension de certaines méthodes d'évaluation de la sûreté de fonctionnement des systèmes. Sa définition est la suivante :

Définition I.6 : Mode de défaillance [CEI74].

C'est l'effet par lequel est observée une défaillance.

Il est important de noter ici, que l'effet doit être rapporté à une entité précise. Si nous reprenons l'exemple du roulement d'une broche. Le bruit émis par le roulement en cas d'usure, constitue un mode de défaillance du roulement et non de la broche. En effet, ce bruit est l'effet produit par l'usure du roulement. Cet effet serait identique même si le roulement appartenait à un autre système (Par exemple un moyeu de roue de voiture ...).

Selon le niveau où l'on se place, un événement peut être considéré comme une défaillance ou un mode de défaillance. Considérons par exemple le système de rotation de notre tête de broche (Fig I.10).

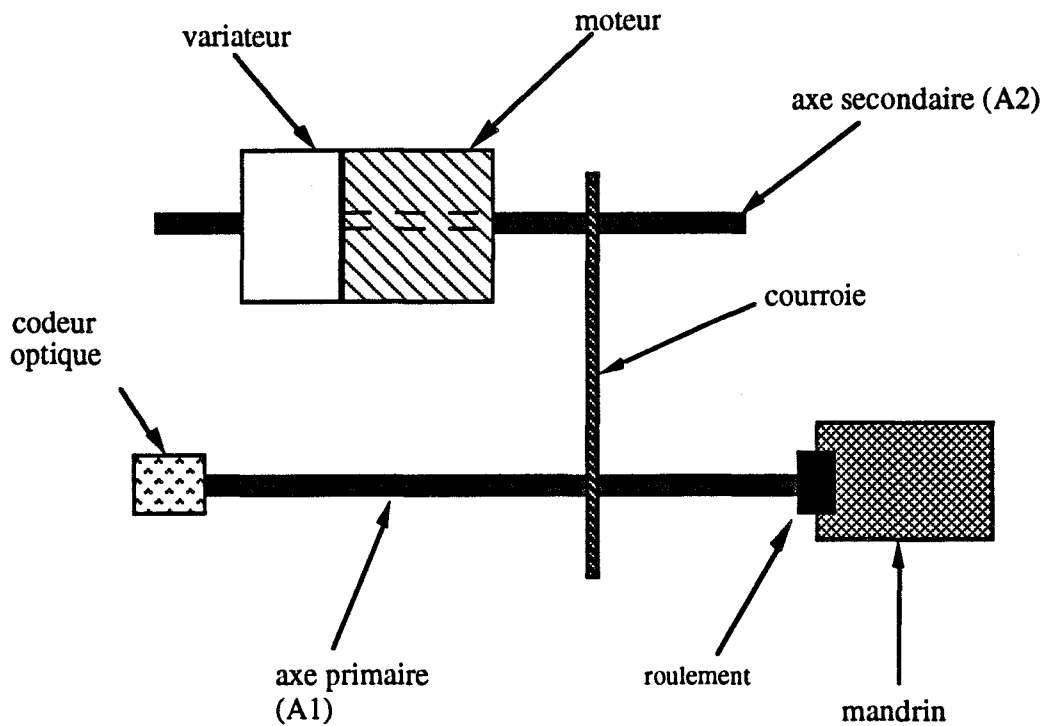


Figure I.10 : Synoptique du système broche/mandrin d'une M.O.C.N.

Dans une M.O.C.N., en cas d'usure du roulement de broche, la boucle de régulation de la vitesse va compenser la décélération provoquée par cette défaillance. Cette compensation se fait par une sollicitation plus importante du moteur. Le moteur va chauffer de manière plus forte que la normale. Pour le moteur, l'augmentation de température représente un mode de défaillance. Par contre, nous constatons que dans cet exemple, la défaillance du moteur est une conséquence de celle du roulement. Donc dans ce cas, la défaillance du moteur est un mode de défaillance du roulement.

Cette dualité entre défaillance et mode de défaillance est une des difficultés rencontrées pour mener une AMDE (Analyse des Modes de Défaillance et de leurs Effets).

I.4.2.3 Classification des défaillances

Il existe plusieurs critères pour classer les défaillances. Parmi ces critères, il y a l'amplitude, la rapidité de manifestation et les causes.

I.4.2.3.1 Classification en fonction de la rapidité de leur manifestation

Définition I.7 : Défaillance progressive [VIL88].

Elle est due à une évolution dans le temps des caractéristiques d'une entité.

L'usure du roulement de broche est une défaillance progressive.

Définition I.8 : Défaillance soudaine [VIL88].

Elle ne se manifeste pas par une perte progressive des performances, mais par un changement brusque des caractéristiques de l'entité.

En cours de rotation, l'arrêt de l'axe primaire est une défaillance soudaine, s'il survient en absence de toute commande.

I.4.2.3.2 Classification en fonction de l'amplitude

Définition I.9 : Défaillance partielle [CEI74].

Elle résulte de la déviation d'une ou plusieurs caractéristiques au-delà des limites spécifiées, mais telle qu'elle n'entraîne pas une disparition complète de la fonction requise.

Définition I.10 : Défaillance complète [CEI74].

Elle résulte de la déviation d'une ou plusieurs caractéristiques au-delà des limites spécifiées, telle qu'elle entraîne une disparition complète de la fonction requise.

I.4.2.3.3 Classification en fonction de la rapidité de leur manifestation et de leur amplitude

Définition I.11 : Défaillance par dégradation [CEI74].

C'est une défaillance qui est à la fois progressive et partielle

C'est le cas de nombreux organes mécaniques pendant la période de vieillissement. A titre d'illustration on peut citer comme défaillance par dégradation :

- l'usure d'un roulement,
- l'usure d'une courroie de transmission,
- l'usure d'une vis à bille,
- l'usure d'un outil de coupe,

- l'obturation d'un conduit du circuit hydraulique, pneumatique ou de graissage d'une machine par effet d'encrassage.

Définition I.12 : Défaillance catalectique [CEI74].

C'est une défaillance qui est à la fois soudaine et complète.

Comme défaillance catalectique, nous citerons :

- la rupture d'un outil à la suite d'un choc,
- l'arrêt d'un vérin à la suite de la rupture d'une conduite en surpression,
- l'arrêt brusque d'une broche en rotation.

I.4.2.3.4 Classification en fonction des causes.

Les causes possibles d'une défaillance sont de deux types :

- les causes internes
- les causes externes ou induites.

Considérons une entité matérielle E_i donnée. Les causes internes expliquant les défaillances de cette entité sont en fait des défaillances de ses composants. L'entité E_i peut elle-même appartenir à un système S . Elle est alors en relation avec d'autres entités E_j de S . Les défaillances de certaines de ces entités E_j peuvent entraîner la défaillance de E_i . On dit alors que les défaillances des E_j influant sur E_i sont des causes externes de celles de E_i .

La notion de cause externe est liée à la capacité de propagation des défaillances à travers la structure d'un système. Pour cette raison, afin de caractériser les origines d'une défaillance, on introduit une autre terminologie.

Définition I.13 : Défaillance première [CEI74].

C'est la défaillance d'une entité dont la cause directe ou indirecte n'est pas la défaillance d'une autre entité.

Une défaillance première caractérise à un niveau structurel donné, la défaillance initiale d'une série de défaillances. Par définition, elle est donc issue d'une cause interne.

Définition I.14 : Défaillance seconde [VIL88].

C'est la défaillance d'une entité dont la cause directe ou indirecte est la défaillance d'une autre entité et pour laquelle cette entité n'a pas été qualifiée et dimensionnée.

Définition I.15 : Défaillance de commande [VIL88].

C'est la défaillance d'une entité dont la cause directe ou indirecte est la défaillance d'une autre entité et pour laquelle cette entité a été qualifiée et dimensionnée.

La différence essentielle entre défaillance seconde (Ds) et défaillance de commande (Dc) se situe au niveau du recouvrement. Pour corriger une défaillance seconde, il est nécessaire de réparer ou de changer l'entité considérée. Dans le cas de la défaillance de commande par contre, le recouvrement de la cause entraîne le recouvrement de la défaillance elle même.

Nous allons illustrer les définitions précédentes en reprenant l'exemple de la broche d'un tour horizontal. Sur ce type de machine, la broche a pour fonction d'entraîner une pièce en rotation à une vitesse donnée.

Supposons que lors d'un essai à vide, la broche s'arrête brusquement après l'émission continue d'un bruit. Il y a défaillance de la broche. Nous avons vu que l'émission de bruit est un mode de défaillance de l'usure d'un roulement. On en déduit que l'arrêt intempestif de la broche est dû à une cause interne : la défaillance du roulement de tête de broche.

Supposons à présent qu'en cours d'usinage, l'ensemble outil/porte-outils vienne percuter violemment la pièce, provoquant l'arrêt de la broche. Supposons par ailleurs qu'un examen de la broche nous indique :

- 1- la détérioration du roulement,
- 2- le déclenchement du relais de sécurité du moteur de broche.

Nous en déduisons que la défaillance première est celle de l'ensemble outil/porte-outils. Cette défaillance a provoqué celle du roulement (défaillance seconde) qui est elle-même à l'origine de celle du moteur de broche (défaillance de commande). On dit qu'on a des défaillances en cascade d'ordre 3.

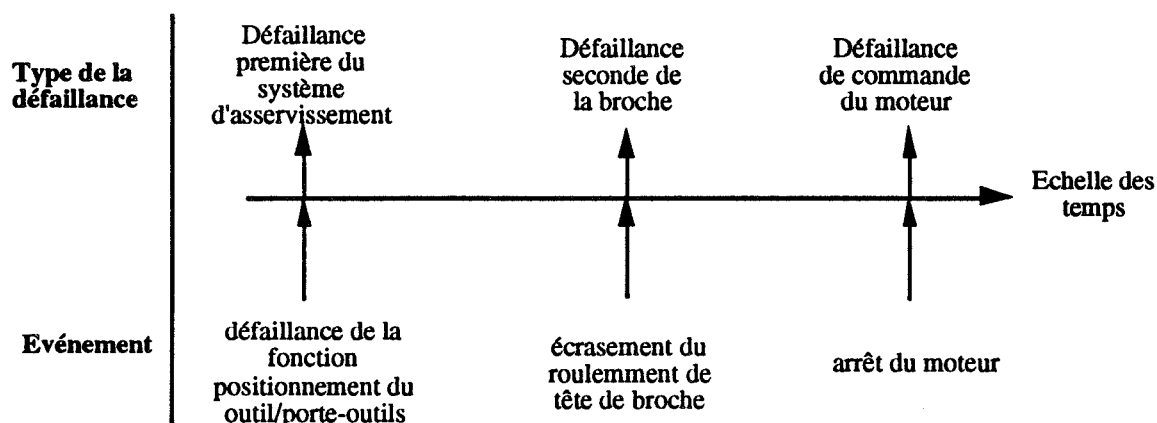


Figure I.11 : Exemple de défaillance en cascade.

Dans le même ordre d'idée, lorsque plusieurs défaillances sont dues à la même cause, on dit qu'elles sont de mode commun.

I.4.2.4 Présentation des méthodes d'analyse de la sûreté de fonctionnement

On distingue deux types de méthodes : les méthodes inductives et les méthodes déductives.

Dans les méthodes inductives la démarche consiste à partir de chaque défaillance à en étudier tous les effets et les conséquences sur le système et sur son environnement. Les principales méthodes inductives sont l'AMDE (l'Analyse des Modes de Défaillance et de leurs Effets), la MTV (Méthode de la Table de Vérité), la MCPR (Méthode des Combinaisons des Pannes Résumées) et la MACQ (la Méthode de l'Arbre des Conséquences).

Les méthodes déductives ont une démarche descendante. Ainsi on partira d'une défaillance du système pour en déterminer toutes les causes potentielles. La principale méthode déductive, est la MAC (Méthode de l'Arbre des Causes). Mais il existe d'autres méthodes comme :

- l'APD ou Analyse Préliminaire des Dangers,
- la MDS ou Méthode du Diagramme de Succès,
- la MDCC ou Méthode du Diagramme Causes-Conséquences,
- la MEE ou Méthode de l'Espace des Etats.

Dans ce mémoire, nous nous limiterons à la présentation de l'AMDE et de la MAC.

I.4.2.4.1 L'Analyse des Modes de Défaillances et de leurs Effets (AMDE).

L'AMDE a été initialement développée pour les analyses de sécurité dans l'industrie aéronautique, au début des années 60. Depuis l'accident de la centrale de Three Miles Island aux Etats-Unis, elle s'est également répandue dans l'industrie nucléaire.

Les caractéristiques principales de cette méthode sont :

- l'évaluation des effets de chaque mode de défaillance des composants d'un système sur ses différentes fonctions,
- l'identification des modes de défaillances ayant d'importants effets sur la disponibilité, la fiabilité, la maintenabilité ou la sécurité du système.

Considérons le système broche/mandrin représenté par la figure I.10. On commence par répertorier les fonctions de chaque composant. Par exemple le moteur a pour fonction d'entraîner en rotation l'axe secondaire A2. On peut répertorier ensuite plusieurs modes de défaillance du moteur :

- la montée en température du moteur,
- l'arrêt de la rotation,
- la disjonction du relais de sécurité.

Les causes potentielles sont multiples. Elles peuvent être internes (court circuit) ou externes (usure de la courroie ou du roulement).

Ensuite on s'intéresse aux conséquences de la défaillance. En cas d'élévation de la température, il n'y a aucune conséquence. Par contre dans les deux autres cas, il y a arrêt de A2.

L'AMDE se poursuit par la recherche des moyens de détection pour chaque mode de défaillance. L'utilisation d'une sonde de température est le moyen approprié pour détecter si la température dépasse un seuil anormal. Pour les deux autres modes de défaillances du moteur, la détection peut être faite à partir de la mesure de la puissance électrique consommée par le moteur.

Puis on classe la défaillance par rapport à son amplitude, la rapidité de ses manifestations et son ampleur.

On a l'habitude de résumer les résultats d'une AMDE dans des tableaux comme celui donné par la figure I.12.

Figure I.12 : Extrait de l'AMDE du système broche/mandrin d'une machine outil à broche horizontale.

Projet:		Système: Broche-mandrin		Date :		Folio: 1		
Identification (localisation) du composant	Fonctions Etats	Modes de défaillance	Causes (possibles)	Conséquences	Moyens de détection	Amplitude	Rapidité des manifestations	Ampleur
moteur	- rotation de A2 à une vitesse de base	- élévation de la température	- surcharge (défaillance de commande),	arrêt de A2	sonde thermique	partielle	progressive	mineure
		- disjonction du relais de sécurité, - arrêt rotation.	- surcharge (défaillance de commande), - court circuit (défaillance interne) - défaillance mécanique (défaillance première).		par la puissance électrique consommée	complète	soudaine	significative
variateur	régler la vitesse de A2	vitesse de A2 , en régime permanent, différente de la consigne (peut être nulle)	- Arrêt du moteur - défaillance mécanique	- Rotation de A2 à une vitesse non conforme	tachymètre lié à A2	partielle	soudaine	significative
A2	Entrainer la courroie	A l'arrêt	- rupture de A2 (défaillance première mécanique) - défaillance du variateur	courroie immobile	tachymètre lié à A2	complète	soudaine	significative
courroie	Transmettre le mouvement de rotation à A1, à une vitesse donnée	- courroie détendue	- rupture	Arrêt de A1		complète	soudaine	significative
		- bruit - vibrations	- usure	réduction de la vitesse transmise à A1	capteurs de vibration ou acoustique	partielle	progressive	mineure
		- courroie immobile	- arrêt de A2	Arrêt de A1		complète	soudaine	significative
A1	transmettre la rotation au mandrin, à une vitesse donnée	arrêt	- défaillance première mécanique (ex: rupture de l'axe) - arrêt de la courroie	arrêt du mandrin	tachymètre lié à A1	complète	soudaine	significative

L'AMDE complète du système broche/mandrin est donnée en annexe A.

I.4.2.4.2 La Méthode de l'Arbre des Causes (MAC).

Elle a été conçue dans les années 61-62 par la Société Bell Téléphone, dans le cadre de l'évaluation de la fiabilité du système de lancement du missile "Minutemen" [VIL88]. Aujourd'hui, elle a été largement adoptée par toutes les grandes industries (aéronautique, nucléaire, chimique, ...). Elle est également connue sous le nom de méthode de l'arbre de défaillances ou méthode de l'arbre des fautes.

Les deux concepts de base de la MAC sont :

*** La notion d'événement indésirable.**

Dans le contexte d'un usinage peuvent être considérés comme événements indésirables :

- la rupture d'un outil,
- le desserrage de la pièce,
- l'échauffement de la pièce ou de l'outil.

*** La notion d'événements de base (Eb).**

Ce sont des événements dont l'analyse sort du cadre de fonctionnement du système considéré.

Par exemple, on peut considérer comme événements de base :

- une erreur humaine, type erreur de montage de l'outil d'usinage,
- un défaut de fabrication dans la pièce ou l'outil,
- l'usure de l'outil.

Le but de la démarche est alors de relier un événement indésirable (racine de l'arbre) à des événements de base (feuilles de l'arbre). Elle comprend les étapes suivantes :

- Etape1 : recherche des causes immédiates, nécessaires et suffisantes (INS) ;
- Etape2 : classement des événements intermédiaires ;
- Etape3 : analyse des défauts de type composant ;
- Etape4 : recherche des causes INS jusqu'à obtention des événements de base ;
- Etape5 : affiner l'arbre de départ à partir des connaissances acquises sur le système.

L'une des raisons du succès de cette méthode est la représentation du résultat de l'analyse sous forme graphique. Chaque événement de l'arbre est représenté par un des symboles de la figure I.13.

symbole	nom du symbole	signification du symbole
	rectangle	Représentation d'un événement (événement indésirable ou intermédiaire) résultant de la combinaison d'autres événements par l'intermédiaire d'une porte logique
	cercle	Représentation d'un événement élémentaire ne nécessitant pas de futur développement.
	losange	Représentation d'un événement qui ne peut être considéré comme élémentaire mais dont les causes ne sont pas et ne seront pas développées.
	double losange	Représentation d'un événement dont les causes ne sont pas encore développées mais le seront ultérieurement.
 	triangle	La partie de l'arbre des causes qui suit le symbole  est transférée à l'emplacement indiqué par le symbole 
 	triangle inversé	Une partie semblable mais non identique à celle qui suit le symbole  est transférée à l'emplacement indiqué par le symbole 

Figure I.13 : Représentation des événements de l'arbre des causes.

A chaque niveau de l'arbre, les événements sont liés au niveau précédent par des portes du type de celles utilisées en logique combinatoire (Fig. I.14).

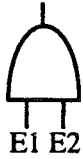
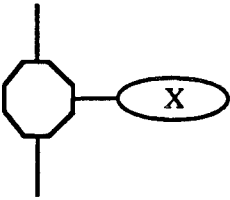
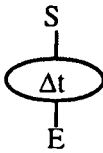
symbole	nom du symbole	signification du symbole
	porte OU	L'événement de sortie est généré si l'un au moins des événements d'entrée est présent.
	porte ET	L'événement de sortie est généré si tous les événements d'entrée sont simultanément présents.
	porte OU EXCLUSIF	L'événement de sortie est généré si l'un au moins des événements d'entrée est présent et si la condition est réalisée (i.e E1 et E2 ne doivent pas être simultanés)
	porte ET avec condition	L'événement de sortie est généré si tous les événements d'entrée sont présents et vérifient la condition (i.e E1 avant E2)
	porte SI	L'événement de sortie est généré si l'événement d'entrée est présent et si la condition X est vérifiée
	porte DELAI	L'événement de sortie est généré avec un retard de Δt par rapport à l'événement d'entrée à condition qu'il n'est pas disparu entre temps.

Figure I.14 : Portes logiques utilisées pour relier les événements.

Illustrons l'application de la MAC par l'exemple de la **rupture d'un outil** en cours d'usinage. La recherche des INS se fait par étapes :

*** Etape1** : on recherche par rapport à l'outil ce qui peut causer la rupture ;

- défaut de fabrication (Eb),
- usure de l'outil (Eb),
- une contrainte trop importante dans l'outil (Ds).

*** Etape2** : on part des INS de l'étape précédente qui ne sont pas considérées comme des événements de base (Eb) ; dans notre exemple, il s'agit de rechercher les causes pouvant provoquer une contrainte trop importante dans l'outil ;

- collision de l'outil avec la pièce, le mandrin ou la contre-pointe (Ds),
- passes trop importantes (Ds),
- défaut d'adaptation de la matière de la pièce (résistance, plasticité) à l'outil (Ds),
- défaut de rotation de la pièce (Dc),
- échauffement du couple pièce/outil (Dc).

A chaque étape, les événements sont qualifiés par Eb (Événement de base), Ds (Défaillance seconde) et Dc (défaillance de commande). Cette manière de faire facilite le développement de l'arbre, dans la mesure où le concepteur repère facilement les événements dont il doit rechercher les INS. Il est important de noter ici que l'arbre obtenu modélise la connaissance du concepteur à une date donnée. Certains événements considérés comme événements de base à cette date (événements représentés par un double losange) pourront être développés par la suite. C'est le cas de la **collision** et de l'**échauffement** sur la figure I.15.

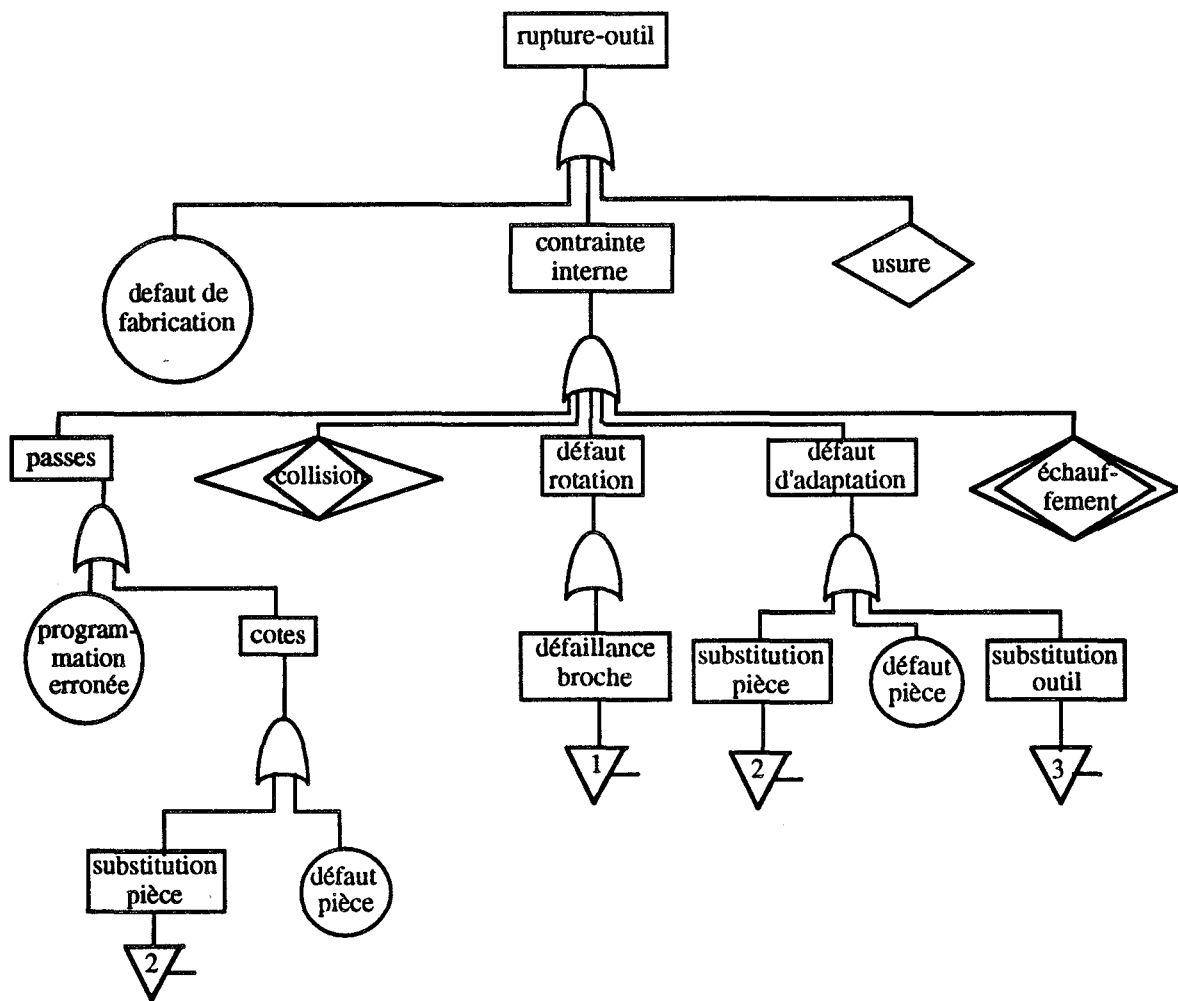


Figure I.15 : Exemple d'arbre de défaillance.

On trouvera la résolution complète de l'exemple précédent dans l'annexe A.

Un des intérêts majeurs de la MAC est qu'elle permet de déterminer les conjonctions d'événements de base minimaux, pouvant entraîner l'occurrence de l'événement indésirable : ces conjonctions d'événements sont appelées *implicants premiers* (cas des systèmes non cohérents) ou *coupes minimales* (cas des systèmes cohérents).

La recherche des coupes minimales se fait en affectant une variable booléenne à chaque événement de base. Ensuite on applique les règles de l'algèbre de Boole pour calculer l'expression booléenne F associée à l'événement indésirable. Cette expression est ensuite réduite par les méthodes classiques (tableau de Karnaugh, algorithme de Mac Cluskey ou de Tison) pour être mise sous la forme :

$$F = C1 + C2 + \dots + Cn \quad (I.5)$$

où les C_i représentent des coupes minimales égales au produit d'événements de base :

$$C_i = E_{b_i^1} \cdot E_{b_i^2} \cdot \dots \cdot E_{b_i^{n_i}} \quad (I.6).$$

Les arbres de défaillance sont également exploités de manière quantitative pour le calcul de la probabilité d'occurrence de l'événement indésirable. Pour cela, à chaque événement de base est associé un taux de défaillance [RAC91]. Le calcul peut être mené directement à partir de l'expression réduite F , ou de proche en proche à partir de l'arbre de défaillance réduit.

I.4.3 Classification de la maintenance.

Les techniques de maintenance ont longtemps été réduites à 2 classes de méthodes :

- la **maintenance préventive** ou maintenance programmée,
- la **maintenance curative**.

La maintenance préventive s'appuie sur la connaissance des paramètres de bon fonctionnement (MTTF, MTBF³) de chaque système et de leurs principaux composants. A partir de ces paramètres, on établit un calendrier pour le contrôle de chaque machine. Comme il est souligné dans [BRU90], cette forme de maintenance coûte cher. En effet, les visites étant périodiques, les pièces sont remplacées aux dates prévues même si elles ne sont pas usées. En outre, on ne tient pas compte du vécu des organes des systèmes. Ainsi, un outil neuf prématurément usé par l'usinage d'une pièce dont le matériau est imparfait (ou inadapté à cet outil) ne sera pas remplacé avant la date prévue. La conséquence est un risque certain de rupture de l'outil avant cette date.

Avec le développement de l'informatique et des techniques de traitement du signal, est apparue une nouvelle forme de maintenance : la maintenance sur état [BRU90]. Sous ce label on regroupe : la maintenance curative et la maintenance prédictive [TAR90].

La maintenance curative est réservée au traitement des défaillances cataclysmiques. Elle est basée sur l'identification rapide des organes défaillants de manière à minimiser le MDT et sur une gestion fine des stocks de pièces de rechange et des agents de maintenance.

³ Voir les définitions en annexe A.

La maintenance prédictive consiste à assurer le suivi des phénomènes d'usure et de vieillissement caractéristiques des défaillances par dégradation. Ce suivi permet de planifier la maintenance de la machine en fonction de l'état réel de ses composants et des impératifs de production.

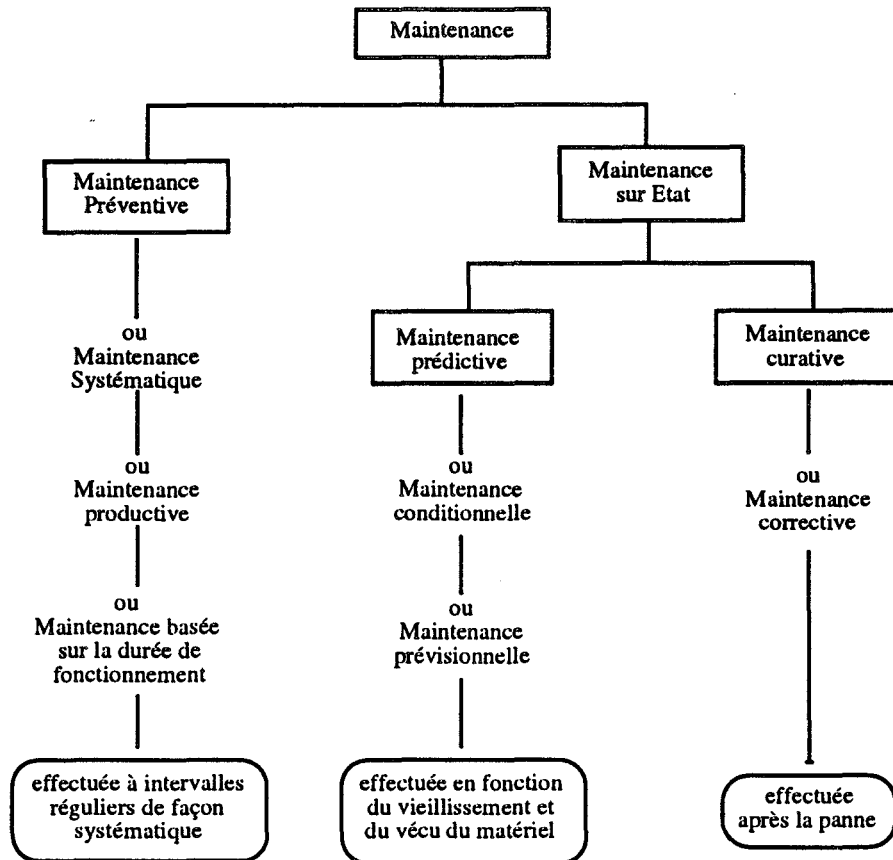


Figure I.16 : Techniques de maintenance.

Depuis plusieurs années, de nombreux outils informatiques ont été développés dans le cadre de l'automatisation de la maintenance. Ils sont connus sous le vocable de GMAO (Gestion de Maintenance Assistée par Ordinateur).

I.4.4 Conclusion.

La fabrication et la maintenance ont toujours constitué les deux principales fonctions de la production. Mais l'automatisation de plus en plus poussée de la fabrication conduit également à moderniser la maintenance car son rôle est devenu critique pour assurer la productivité. Aujourd'hui, non seulement elle doit assurer sécurité, fiabilité et disponibilité des moyens de production, mais en plus elle doit permettre de réduire les coûts de production. Cela passe

nécessairement par une réduction des coûts de maintenance eux-même. Mais cela passe également par une diminution de la quantité de pièces à envoyer au rebut pour leur qualité médiocre. Une bonne politique de maintenance est donc un gage de productivité et de qualité, donc en un mot d'efficacité [LEC91]. Pour atteindre cette efficacité, **dans le cadre de la commande des SFP, nous pensons qu'il est nécessaire d'adopter une politique de maintenance sur état**. Même si tous les problèmes n'ont pas encore été résolus, d'importants progrès ont été réalisés dans le cadre de la maintenance prédictive [TAR90], [BOI91/a], [LEV91], [DEV91]. Par contre aussi surprenant que cela puisse paraître, on peut parler de retard en ce qui concerne la maintenance curative. Non pas en ce qui concerne les actions correctives à mener une fois la panne identifiée ; là où les progrès sont attendus c'est au niveau du diagnostic et de l'identification même des pannes. Les techniques n'ont pas suffisamment évolué. Elles sont toujours basées sur le diagnostic hors ligne du procédé. Cela n'est pas concevable dans les SFP où la disponibilité est un critère aussi important que la sécurité. Toute approche novatrice doit permettre de traiter en ligne les défaillances : c'est le but de la surveillance en ligne.

1.5 Conclusion

Comme nous venons de le voir tout au long de cette partie, la surveillance est une fonction critique, nécessaire au bon fonctionnement de tout SA. Au niveau des SFP une telle fonction doit notamment permettre de traiter en ligne les défaillances. Elle ne peut pas être conçue à posteriori, lorsque le système a été pensé par rapport à un mode de fonctionnement dit normal. Nous pensons que la gestion des modes dégradés doit être prévue dès la phase de spécification dans le cadre d'une politique de maintenance sur état. L'expérience des chaînes de production et des ateliers flexibles pilotes, montre que le fonctionnement normal est rarement atteint en raison des aléas de la production. Il serait à notre sens plus correct de l'appeler **mode de fonctionnement optimal**. Ainsi, les modes dit dégradés, correctement gérés, seraient assimilés à des fonctionnements normaux. A notre sens, il n'y a qu'un seul fonctionnement anormal : la panne totale du système. Comme c'est le cas dans la conduite-surveillance des systèmes continus, le rôle du concepteur est d'intégrer tous les outils nécessaires pour contrôler et contraindre le système à suivre sa consigne au point de fonctionnement optimal : c'est la fonction assurée par le pilotage et la surveillance (cf. figure I.6).

La surveillance doit être conçue dans le cadre d'une politique de maintenance, commençant par une analyse préliminaire de sûreté de fonctionnement, pour se poursuivre par la surveillance et se terminer par l'entretien et la réparation des moyens de production.

Supposons que pour assurer une fonction de production (usinage, assemblage, ...) selon des ratios donnés, le concepteur ait le choix entre une machine à forte capacité et deux machines de capacité moyenne. S'il se place dans un cadre de fonctionnement normal, il y a de fortes chances qu'il choisisse la machine à forte capacité : pour des raisons de coût d'achat, mais également de coût d'entretien. En effet, rapporté à une politique de maintenance préventive, une machine (serait-elle à forte capacité) demande de fois moins d'efforts de maintenance ... que deux machines. Pourtant, il est évident que la redondance de fait, obtenue par deux machines améliore la disponibilité moyenne de la fonction. Cette analyse rapide, nous montre que dans les préliminaires de toute conception de SFP, une étude de sûreté de fonctionnement du système s'impose. Une telle étude permet non seulement de faire le choix des composants machines du SFP, mais également donne une première architecture pour leur répartition. L'architecture définitive est obtenue en réalisant un compromis entre les contraintes de la commande nécessaires pour respecter les flux de produits optimaux, et les contraintes de la surveillance nécessaires pour assurer des flux minimaux.

II. CHOIX D'UNE POLITIQUE DE SURVEILLANCE ADAPTÉE À LA MAINTENANCE DES SFP.

Dans la partie précédente, nous avons conclu que la surveillance en ligne intégrée à une politique de maintenance sur état est la meilleure approche de surveillance des SFP. Dans cette partie, nous allons définir notre conception de la surveillance en ligne à partir d'une analyse de l'existant.

II.1 Fonctionnalités d'un système de surveillance : terminologie et principaux outils.

D'après [BRU90], dans le contexte le plus général, les fonctionnalités d'un système de surveillance sont :

- l'acquisition des données,
- la perception des défauts,
- la détection,
- la localisation-diagnostic,
- le pronostic,
- l'analyse des conséquences,
- la planification des actions.

* **L'acquisition des données.**

Il s'agit d'acquérir les informations pertinentes du procédé ou de sa commande. Cette fonction fait appel aux techniques de traitement du signal pour conditionner, filtrer et valider les signaux. Dans certains cas, la détection de défaut capteur peut se faire au niveau de l'acquisition.

* **La perception des défauts.**

Elle est conçue comme une fonction de **compressions de données**. En effet, son rôle est d'extraire de la masse de données brutes, les **indicateurs de défauts** nécessaires à la détection. Les méthodes utilisées sont essentiellement à base de **traitement du signal** (analyse temporelle du signal et traitement statistique, ou analyse fréquentielle) ou à base de **modélisation**.

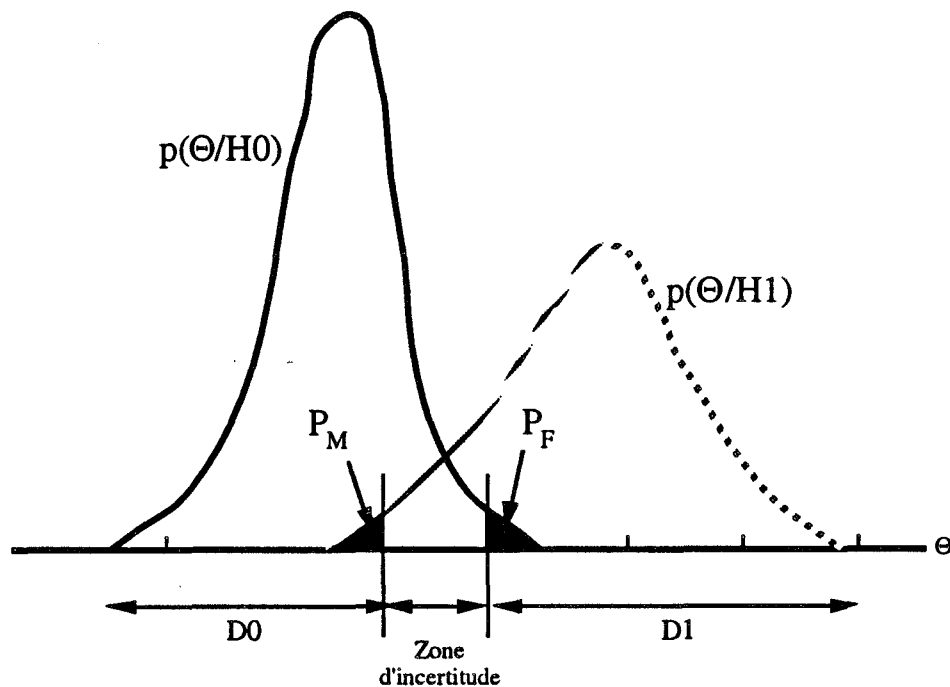
Le principe des méthodes à base de modélisation est de comparer l'évolution du système surveillé à celle d'un modèle soumis à la même commande. Dans ce cas les indicateurs de défauts sont appelés **résidus**. Les modèles utilisés appartiennent à deux types :

- Les modèles de structure qui établissent des relations causales entre les entrées et les sorties par le biais de paramètres représentatifs de la structure physique du système.
- Les modèles mathématiques qui sont donnés par des équations d'états ou des fonctions de transfert (cas des systèmes mono-variables).

* **La détection.**

C'est la fonction qui doit décider si le système est défaillant. C'est donc une opération logique dont la réponse doit être binaire (vrai, faux). Les signaux étant généralement bruités, il y a un risque d'erreur. Ce risque est traduit sur la figure I.17, par une zone d'incertitude délimitée par les probabilités de manque (non détection) PM, et de fausse alarme PF. Pour réduire ce risque, il est courant d'utiliser en amont de la détection, des techniques de filtrage et d'estimation (Méthode des moindres carrés, filtre de Kalman, ...) des variables utilisées pour le calcul des résidus.

Lorsque la détection est effectuée en temps réel, le risque d'erreur est minimisé en utilisant le test séquentiel d'hypothèses. Cela consiste à ne pas prendre de décision lorsque le test tombe dans la zone d'incertitude, mais à la reporter aux observations suivantes. Dans ce cas, il faut faire un compromis entre précision de la détection et rapidité.



Légende :

Θ est le vecteur observation

$p(\Theta/H_i)$ est la probabilité conditionnelle par rapport à l'hypothèse H_i

P_M est la probabilité de manque ou non détection

P_F est la probabilité de fausse alarme

D_i est l'espace de décision

Figure I.17 : Test d'hypothèses.

* La localisation-diagnostic.

La localisation consiste à attribuer les défaillances détectées à des sous ensembles du système : capteurs, organes de commande, processus, unité de commande. Quant au diagnostic, il doit assurer deux fonctions : l'**identification des causes d'une défaillance** et sa **classification** (par rapport à son amplitude et son ampleur).

Les principaux outils utilisés pour l'identification sont :

- la reconnaissance de formes [DUB83],
- l'analyse discriminante floue,
- la logique floue,

- les arbres logiques,
- les systèmes experts.

La classification concerne en particulier les défaillances dues à l'usure ou au vieillissement.

* **Le pronostic.**

Le pronostic a pour rôle de **prédire l'évolution future de la défaillance**. Il doit notamment indiquer le temps restant avant que la défaillance ne devienne critique. Les outils du pronostic sont basés sur des techniques de traitement du signal (méthodes de lissage et d'extrapolation) ou de modélisation.

* **L'analyse des conséquences.**

C'est une fonction complémentaire de la précédente. Son rôle majeur est la prédiction des pannes induites. Dans ce cadre, les méthodes de sûreté de fonctionnement (MAC, MACQ, ...) sont souvent utilisées pour la conception de systèmes experts.

L'analyse des conséquences peut également concerner des points comme :

- la sécurité,
- la qualité (Faut-il mettre au rebut la pièce qui était en cours d'usinage lors de la rupture d'un outil ?),
- les aspects économiques (risque de ne pas fournir le produit à la date due).

* **La planification des actions.**

Il s'agit ici de planifier dans le temps, les actions correctives nécessaires à la poursuite du fonctionnement du système. Ces actions doivent permettre le confinement des défaillances et les reconfigurations et passages en mode dégradé. Les outils utilisés sont généralement à base de systèmes experts.

II.2 Les différentes formes de surveillance : définitions et classifications.

On distingue plusieurs modes de classification de la surveillance :

- en fonction du mode d'acquisition,
- en fonction de la partie du SA surveillé,
- en fonction de la nature du procédé.

II.2.1 Classification en fonction du mode d'acquisition des données.

II.2.1.1 Surveillance continue.

Ce type de surveillance est utilisé dans le cadre d'une politique de maintenance prédictive. Elle est caractérisée par une acquisition en continue des mesures nécessaires à la détection et au diagnostic. Elle concerne en général les systèmes ou processus continus. A titre d'illustration, nous pouvons citer la surveillance de la température d'un four, la surveillance des machines tournantes, ...

Dans l'industrie manufacturière, elle est utilisée en surveillance des outils en cours d'usinage [CET85].

Toutes les techniques industrielles utilisent des capteurs spécifiques à la surveillance :

* Capteurs pour mesure de forces de coupe.

Ils sont montés soit directement sur les outils (solution coûteuse), soit dans un sous-ensemble de la machine (torelle, porte-outil, broche, ...). Le but est de suivre les efforts de coupe pour les comparer à une courbe de référence (Fig I.18).

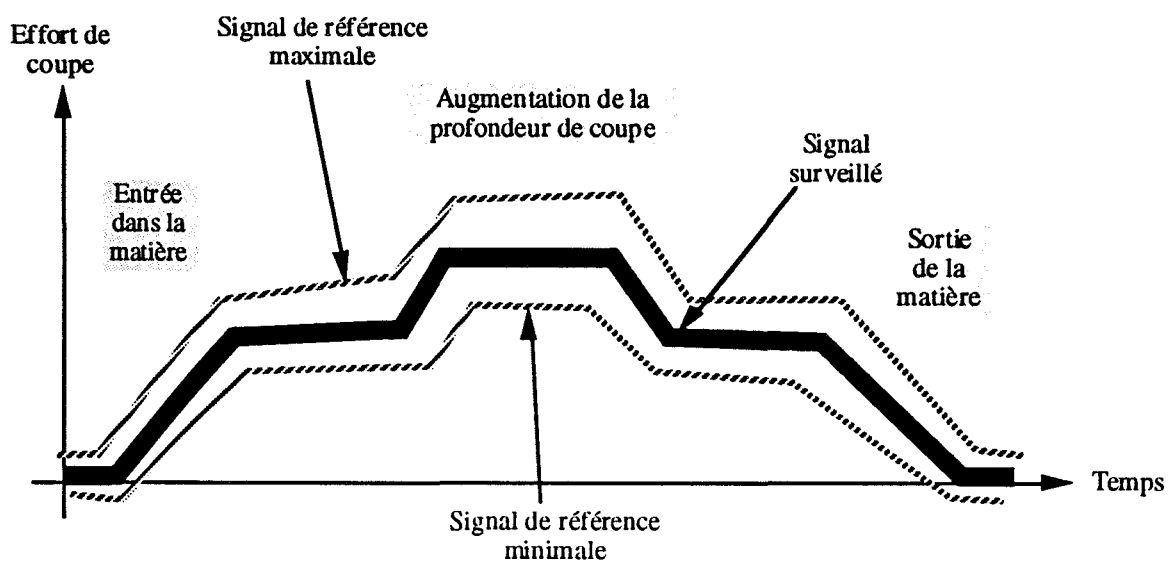


Figure I.18 : Suivi des efforts de coupe.

La courbe de référence peut être obtenue par apprentissage sur des séries de couple pièces-outils. Cette méthode est lourde car elle nécessite de nombreux essais qui sont fonction de la nature des matériaux (pièce et outil), des vitesses d'avance et de rotation, des profondeurs de passe. Il est préférable dans ces conditions de modéliser le processus de coupe.

*** Capteurs de vibrations.**

Ils sont également utilisés en maintenance préventive. Ils conviennent bien à une surveillance multi-paramètres avec diagnostic par la reconnaissance de forme. En effet, positionnés sur un système comme une machine outil, ils vont détecter les vibrations de sources différentes : roulements, processus de coupe, paliers, moteurs ... C'est pourquoi leur utilisation est liée à la qualité du traitement du signal qui doit éliminer les fausses informations.

*** Capteurs d'émission acoustique.**

Ce sont en général des capteurs piezoélectriques ultrasonores sensibles à des rayonnements de très hautes fréquences (100 à 500 kHz). Leur implantation est aisée et de plus ils sont généralement peu sensibles aux bruits des machines.

Il existe également des techniques utilisant la mesure de la puissance absorbée par les moteurs de broche ou d'avance. Ces techniques ne sont valables que lorsque la puissance absorbée par le processus de coupe (tournage, chariotage, fraisage ...) et du même ordre que la puissance à vide.

II.2.1.2 Surveillance périodique.

Le principe est le même que celui de la surveillance continue. La différence se situe seulement au niveau de la fréquence des mesures sur le système surveillé. La surveillance périodique est adaptée au suivi de phénomènes de vieillissement dont on a une bonne connaissance de l'évolution. Elle peut être utilisée pour le suivi du vieillissement d'une huile de lubrification, ou de l'encrassement d'une conduite d'un des circuits (graissage, hydraulique, pneumatique) d'une machine. Pour que ce type de surveillance soit efficace, on doit positionner plusieurs seuils d'alerte de manière à augmenter la fréquence des mesures au fur et à mesure du vieillissement. A partir d'un certain seuil, une surveillance périodique peut même être transformée en surveillance continue. Quoiqu'il en soit, une surveillance périodique pour être efficace, ne doit jamais permettre le passage d'une défaillance par dégradation à la défaillance catalectique.

II.2.1.3 Surveillance apériodique.

Dans les deux formes de surveillance précédente, l'acquisition des données est à l'initiative du système de surveillance. La communication avec la PC et le procédé est en mode espion, ou au moins en mode requête-réponse [CRA89]. Pour la surveillance apériodique, la

communication est à l'initiative du système surveillé. La différence est que le système de surveillance n'est pas saturé par des données inutiles. Il est juste alerté lorsque que l'état du système surveillé le nécessite. On verra que ce type de surveillance est particulièrement efficace dans le contexte d'une mise en œuvre temps réel.

II.2.1.4 Conclusion

Le choix d'une des deux premières formes de surveillance et surtout lié à la durée du phénomène surveillé, aux capacités de stockage des données acquises et au degré de sécurité attendu. Ainsi pour les phénomènes d'usure de faible durée comme les usinages, on privilégiera la surveillance continue. Pour les phénomènes de vieillissement qui sont relativement lents au début, on adaptera la fréquence des mesures au degré du vieillissement.

II.2.2 Classification en fonction de la partie surveillée.

II.2.2.1 Surveillance du procédé.

Dans le procédé, on distingue classiquement les produits, les moyens de production et les processus de fabrication également appelés opérations. A chaque partie du procédé peuvent être associées des méthodes de surveillance.

II.2.2.1.1 Surveillance des produits.

Les approches peuvent être aussi bien quantitatives que qualitatives. Au niveau quantitatif, on compare les taux de produits fabriqués aux taux planifiés. Tout écart important (à définir au moment de la planification) est révélateur d'une anomalie dans le processus de fabrication. Cette approche peut être utilisée à différents niveaux dans la surveillance d'un atelier flexible.

Au niveau machine, il suffit de comparer le flux de pièces sortant par rapport au flux programmé rapporté au flux de pièces entrant. Ainsi, le flux programmé étant établi à partir des caractéristiques nominales de la machine, une dérive du flux sortant d'une catégorie de pièces peut être révélateur de l'usure d'une catégorie d'outils.

Au niveau d'un atelier une méthode similaire basée sur la notion de "profiles de flux" a été proposée dans [ARC91].

La qualité des produits peut également être indicatrice de dérives dans le processus de production. Dans l'industrie manufacturière, cette approche repose sur une instrumentation de

métrologie. Ainsi une dérive dans des cotes de pièces conduira à court terme à une reprogrammation des jauges outils. A plus long terme on pourra changer les outils. Par contre, des ébavures sur la surface d'une pièce seront synonyme d'une rupture de d'outil.

L'inconvénient de ces méthodes est qu'elles sont relativement lentes et peu précises. Ce n'est pas ainsi que l'on peut gérer des arrêts d'urgence. A notre avis, elles doivent être associées à des méthodes plus directes, par exemple comme outils de perception des défaillances par dégradation. Elles peuvent également servir à assurer une certaine robustesse, par la mise en œuvre d'une surveillance redondante, dans le cadre des défaillances catalectiques.

II.2.2.1.2 Surveillance des moyens de production.

Il s'agit d'une surveillance directe des moyens de production par des capteurs de surveillance dédiés : capteurs de vitesse, de pression, thermocouples etc ... Les capteurs peuvent être positionnés directement sur les éléments surveillés. C'est ce que l'on fait lorsqu'on positionne un thermocouple dans une glissière ou sur un moteur. Avantage de la méthode, la localisation de la défaillance est directe et l'on peut déclencher tout de suite les procédures de recouvrement à court terme. C'est ce que préconise des méthodes comme l'AMDE. Elles présentent toutefois l'inconvénient qu'on ne peut pas placer un capteur sur chaque composant du système surveillé.

L'approche contraire consiste à répartir des capteurs dans toutes la structure : par exemple, la fixation de capteurs de vibration à différents endroits sur le bâti d'une machine outils. La méthode est moins coûteuse mais offre moins de sécurité. En effet, la localisation et le diagnostic sont plus lents car on doit recouper les informations issues des différents capteurs. Dans le cadre de l'analyse des phénomènes vibratoires, la méthode la plus utilisée pour le diagnostic est l'approche par la reconnaissance des formes [DUB83]. Un des inconvénients majeurs de cette approche réside dans le passage obligatoire par une phase d'apprentissage afin de diviser l'espace paramétrique en mode de fonctionnement. Par ailleurs, il n'existe aucune approche formelle pour le positionnement des capteurs. Il est souvent le fruit de l'expérience.

II.2.2.1.3 Surveillance des opérations.

Elle consiste à vérifier que chaque opération participant au processus de fabrication se déroule correctement. Dans les procédés continus complexes comme ceux de l'industrie chimique, ce rôle est souvent rempli par des opérateurs humains (cf. § I.2). Dans l'industrie manufacturière, ce contrôle est automatiquement réalisé par la Partie Commande, par l'intermédiaire des comptes-rendus transmis par des capteurs de position ou de présence. Par

exemple, l'opération de chargement d'une pièce sur une machine outil pourra être surveillée à l'aide de trois capteurs (Fig I.19) :

- un capteur de présence de la pièce sur son support de départ,
- un capteur de présence de la pièce dans la pince du manipulateur de chargement,
- un capteur de présence de la pièce entre les mors de la broche de réception.

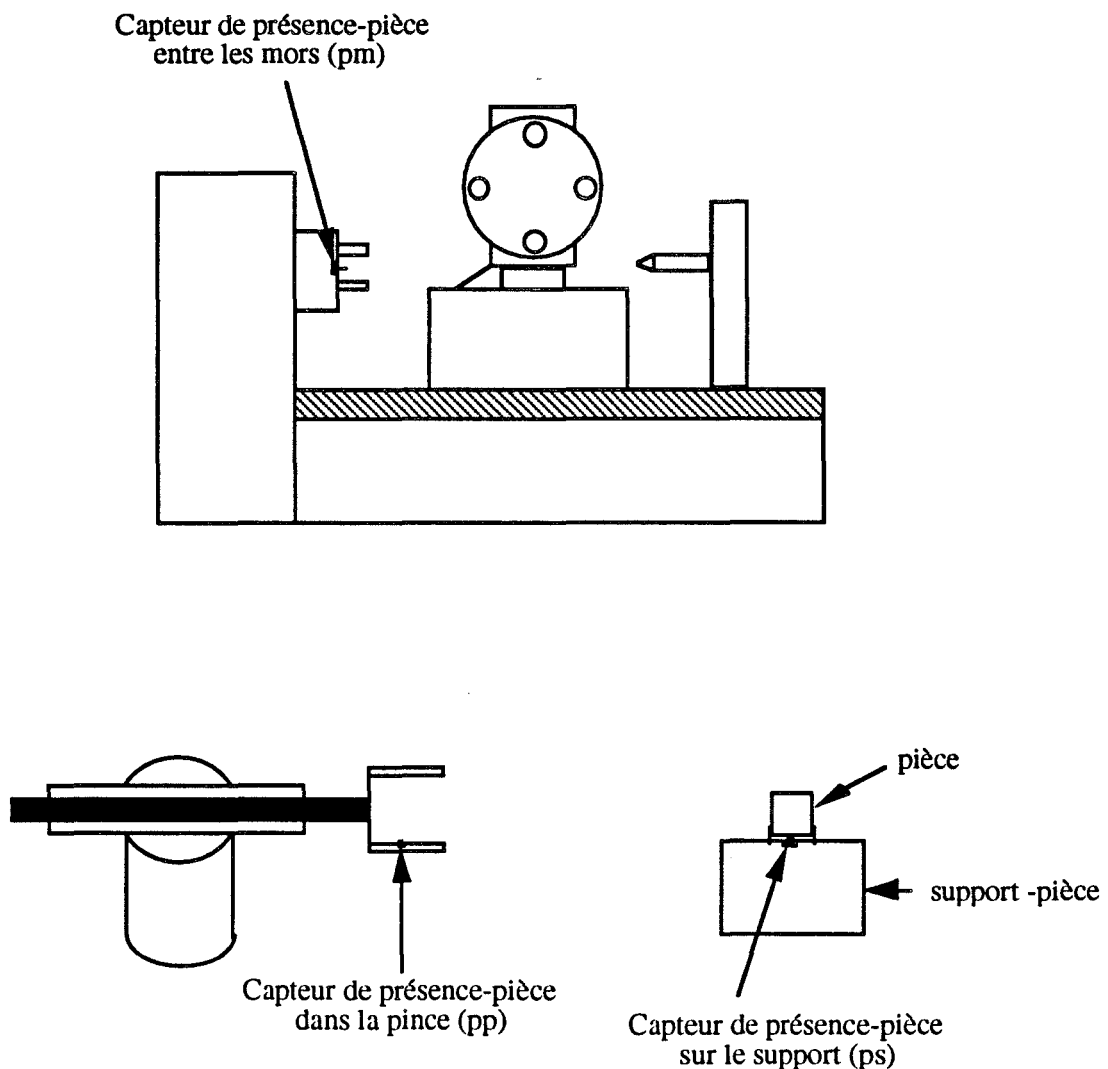


Figure I.19 : Positionnement des capteurs utilisés pour la surveillance d'une opération de chargement de pièce.

Les capteurs utilisés dans ce contexte, délivrent généralement une information binaire qui est directement interprétée au niveau de la commande (Fig I.20) : c'est le **concept de sûreté de séquence** [SAH87], [SAH92].

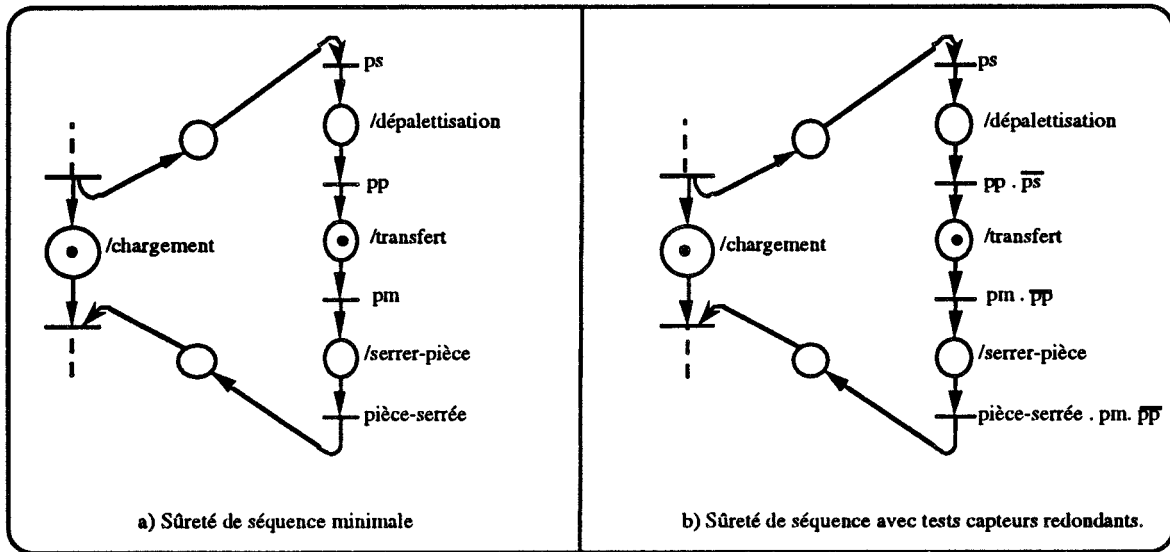


Figure I.20 : Illustration de la sûreté de séquence pour le chargement d'une machine outil.

En fait, une défaillance d'opération traduit soit une défaillance du procédé, soit une défaillance de commande. Aussi, dans l'hypothèse d'une commande saine, la surveillance d'opération est équivalente à une surveillance directe du procédé.

II.2.2.2 Surveillance de la commande.

Obtenir une commande saine est l'un des objectifs de toute méthodologie de conception. Mais on n'est jamais à l'abri des erreurs de conception. C'est la raison pour laquelle la commande doit également être surveillée.

Dans la méthodologie CASPAIM, la surveillance de la commande est prise en compte à deux niveaux :

- au niveau conception d'une part par la modularité et la structuration de la commande [BOU88], et d'autre part par la simulation [CAS87],
- au niveau opérationnel par la notion de filtre de commande [CRU91], et celle de contrôle commande [ELK92/a], [ELK92/b].

Considérons le procédé représenté sur la figure I.21. Il comprend deux manipulateurs, réalisant le chargement de deux machines outils, positionnées en vis à vis. Les deux manipulateurs ont accès au même stock de pièces brutes. Il y a donc un risque de collision dans la zone dangereuse délimitée autour du stock. Nous supposons que matériellement, la zone

est couverte par un champ magnétique détectant la présence de l'un ou l'autre des manipulateurs à l'intérieur. Du point de vue de la commande, comment va-t-on surveiller l'accès à cette zone ?

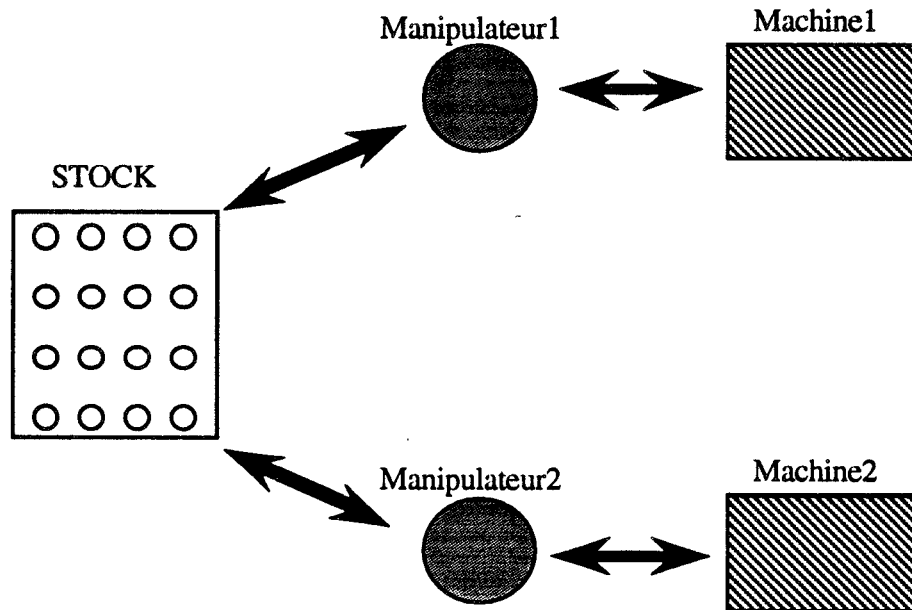


Figure I.21

Dans le contexte de la méthodologie CASPAIM-1, la surveillance de la commande de l'accès des deux manipulateurs au stock de pièces, passe par la définition au niveau de la commande, d'une "ressource à accès exclusif" [BOU88], modélisant la zone dangereuse. Ainsi, l'évolution des processus de transfert relatifs à chaque manipulateur nécessite de s'approprier au préalable la ressource (Fig. I.22).

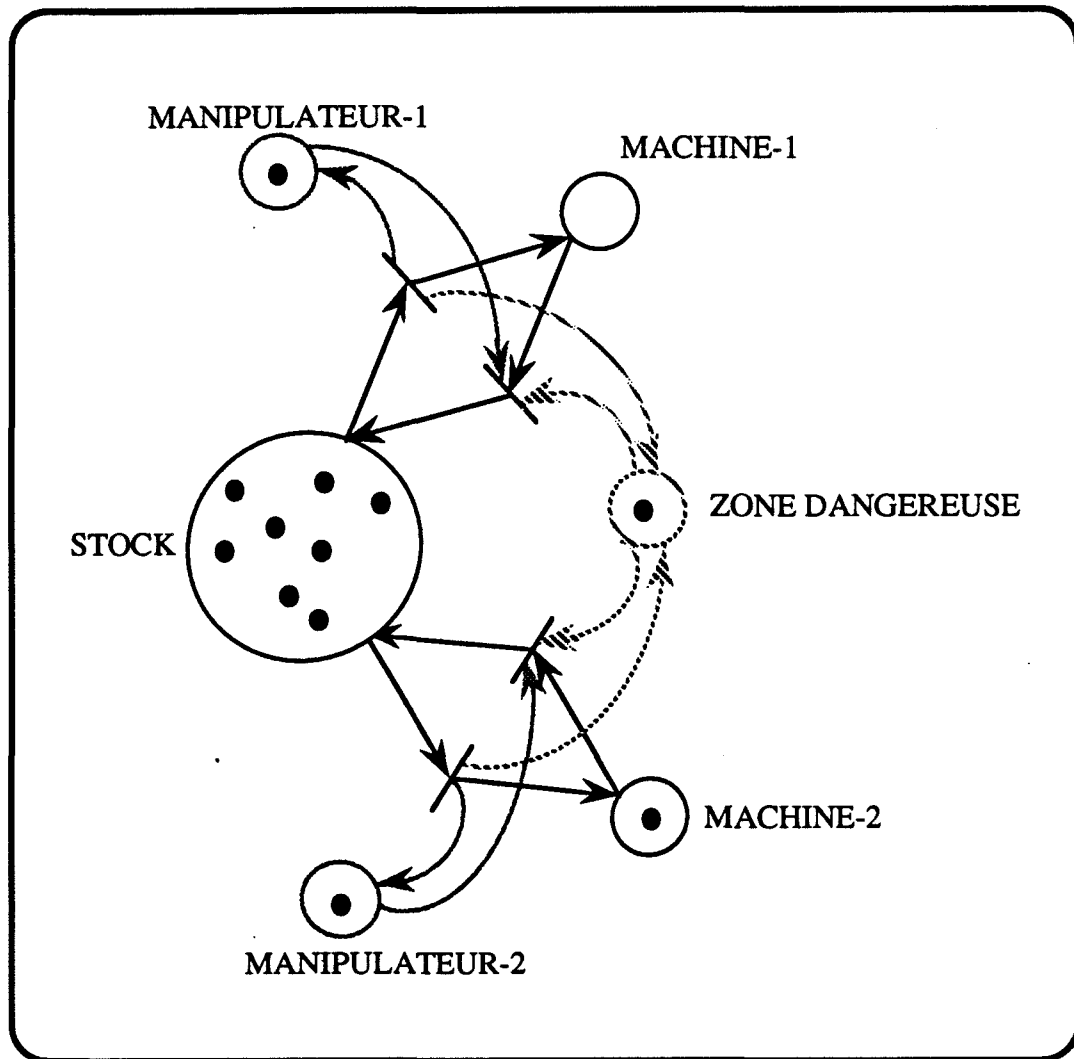


Figure I.22 : Surveillance implicite de la commande dans CASPAIM-1.

Il est important à ce stade, de bien comprendre que la définition de la ressource à accès exclusif n'est pas liée au contrôle, mais relève d'une surveillance diluée dans le graphe de commande. Cette remarque montre clairement les limites d'une telle approche et illustre bien notre conviction que la surveillance doit être séparée à la fois de la commande et du procédé (cf. § III.2.1).

Dans CASPAIM-2, la surveillance de la commande est liée à la notion de filtre de commande. Les filtres de commandes sont des automates modélisant le comportement de chaque objet physique commandable [CRU91]. Illustrons ce concept sur notre exemple (Fig. I.21). Le comportement de chaque manipulateur est modélisé par un graphe à 7 états (Fig. I.23).

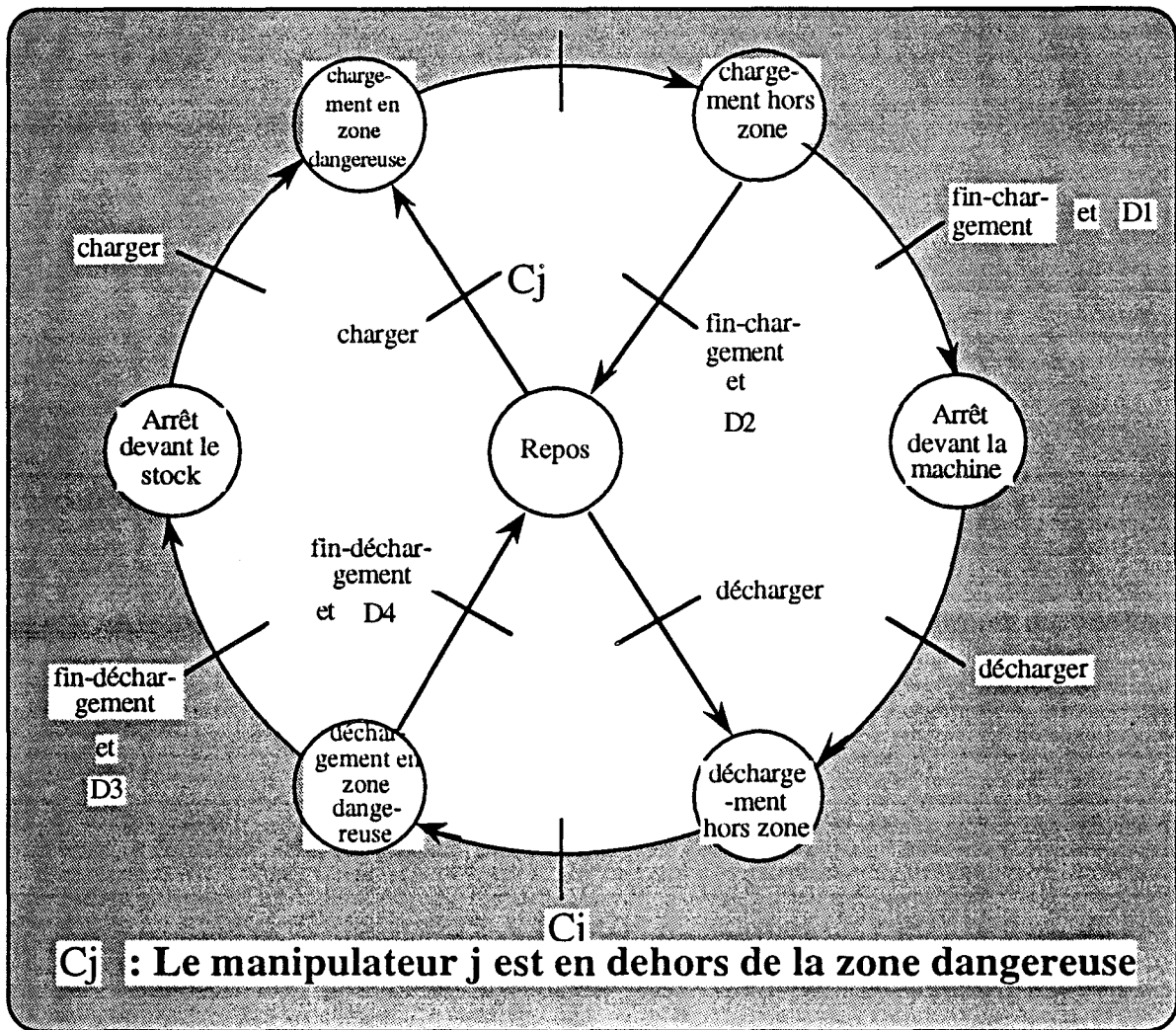


Figure I.23 : Graphe de comportement du manipulateur (i).

Dans le graphe précédent, le chargement est une opération composée du transfert et du chargement effectif des pièces dans la machine. A la fin du chargement, chaque manipulateur peut rester devant la machine pour récupérer la pièce une fois usinée. Il peut également retourner à sa position de repos. Le comportement du manipulateur est décidé par le Niveau Hiérarchique (variables D1 ... D4).

Pour assurer leur fonction de surveillance, chaque filtre doit vérifier deux propriétés :

- il doit être déterministe par rapport à la commande et aux comptes-rendus de fin d'action,
- il doit tenir compte des contraintes opérationnelles le liant aux graphes de comportement d'autres objets commandables situés au même niveau de commande que lui.

Par exemple la prise en compte des contraintes opérationnelles entre les deux manipulateurs de notre exemple a été matérialisé par la condition C_j sur le graphe de comportement du

manipulateur (i). Cette condition pose que le manipulateur (i) ne doit pas rentrer dans la zone dangereuse si le manipulateur (j) s'y trouve déjà. On notera, que le modèle comportemental défini ici, permet d'amorcer le transfert de chaque manipulateur jusqu'à la zone dangereuse, quitte à attendre ensuite que celle-ci soit libérée.

Fort de cette analyse, S. Elkhatabi propose la structuration suivante pour la fonction surveillance de la commande :

- un module **filtre de commande**, chargé de vérifier l'adéquation entre les commandes émises par la Partie Commande et l'état réel du procédé,
- un module de **contrôle commande**, qui a pour fonction de contrôler la réalisation à court terme d'une action filtrée,
- un module de **recouvrement d'erreur** qui a pour but d'entreprendre la **reprise** des actions filtrées non réalisées.

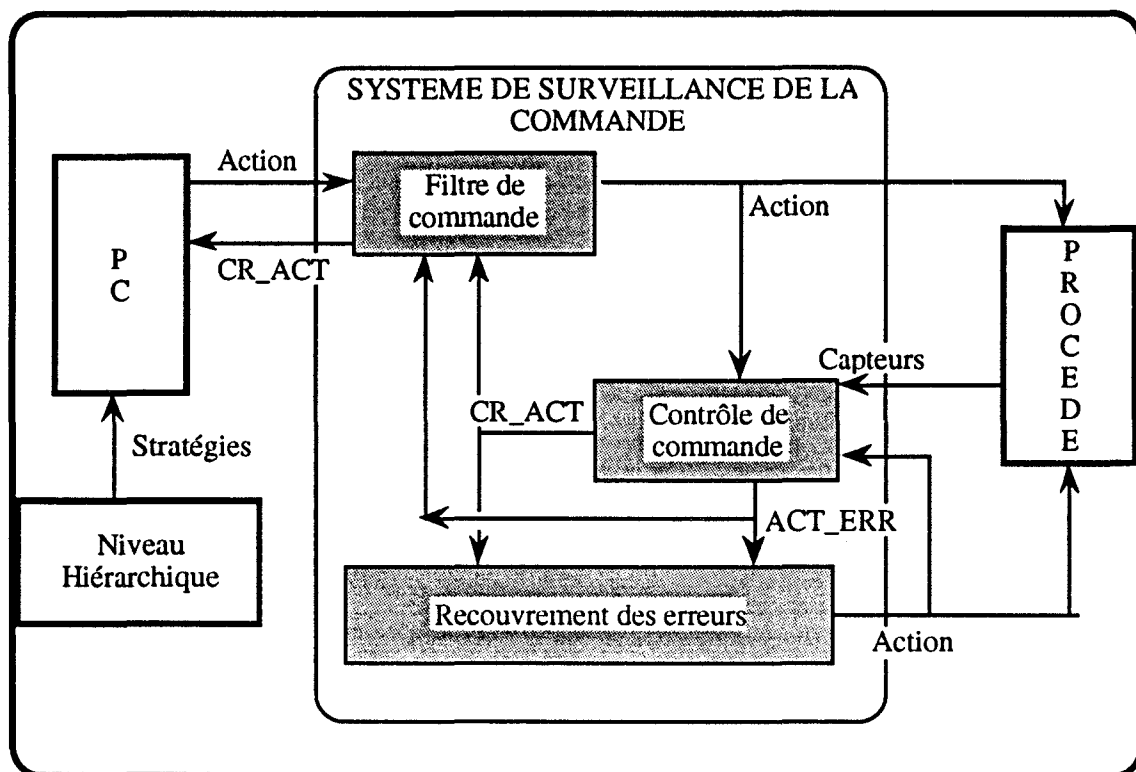


Figure I.24 : Proposition de structuration de la fonction surveillance de la commande d'après [ELK92/b].

Les travaux de S. Elkhatabi, sont basés sur la notion d'OCE (Objet Commandable Élémentaire) [ELK92/a]. C'est une notion proche de celle d'EPO (Elément de Partie Opérative) introduite par l'équipe du CRAN/LACN (cf. § III.2.2). La différence essentielle entre ces deux

méthodes se situe au niveau de la prise en compte des contraintes opérationnelles dans l'approche de S. Elkhatabi. Par ailleurs, ce dernier propose une méthodologie de conception pour établir un modèle comportemental de tout Objet Commandable à partir de la définition des OCE du système. Au niveau, des langages de spécification, les deux approches diffèrent également. S. Elkhatabi, préconise l'utilisation de langages synchrones tels que Esterel, Lustre ou Signal. L'hypothèse qui explique ce choix, c'est que seuls de tels langages peuvent assurer le comportement réactif [BEN89] nécessaire à l'intégration de la fonction surveillance dans la commande. Notons que l'approche du CRAN/LACN utilise le grafcet comme outil de spécification.

La fonction surveillance de la commande doit être interprétée comme une fonction de sécurité.

II.3 Influence de la nature du système sur le type de surveillance et de maintenance.

II.3.1 Les systèmes continus.

Ce type de système est caractérisé par un fonctionnement en continu ne permettant pas de discrétiser le processus de fabrication. Ainsi, les variables (variables de type Température, Pression, Volume ...) caractérisant l'état du procédé sont généralement continues et de nature analogique. Par ailleurs la commande comprend des boucles de régulation qui ont tendance à masquer les perturbations de faibles amplitudes dues à des défaillances du procédé. Pour toutes ces raisons, nous pensons que la surveillance continue est la plus appropriée à ce type de système. Néanmoins, la lenteur d'évolutions des phénomènes dans ce type de procédé (de l'ordre de la minute), permet également d'envisager une surveillance périodique.

Le suivi "continu" des principales variables du système permet de connaître à tout moment son état. On peut donc suivre l'évolution des défaillances et anticiper leur transformation en défaillance catalectique. La maintenance prédictive est donc adaptée aux systèmes continus.

II.3.2 Les systèmes discrets.

Dans ces systèmes la notion d'événements est essentielle. En effet, dans un système discret, tout processus peut être caractérisé par deux événements : **l'événement de début** de processus et **l'événement de fin** de processus. En général, dans un SA, l'événement de début correspond à un **ordre de la PC** et l'événement de fin correspond à un **compte-rendu du procédé**. Une manière simple de surveiller le procédé consiste alors à vérifier que l'on peut associer un compte-rendu unique à chaque ordre émis. Dans ce contexte, l'acquisition des données de la surveillance est à l'initiative du système surveillé : on fait de la surveillance apériodique.

Cette manière de surveiller a cependant un inconvénient. Il y a toujours un décalage d'une "période" entre le procédé réel et l'image qu'en a le système de surveillance. La conséquence de ce décalage, est qu'il n'est pas possible d'anticiper des défaillances premières. Seules les défaillances de ces conséquences peuvent être anticipées. Donc une telle approche de la surveillance se situe dans le cadre d'une maintenance curative.

II.4 Conclusion

Nous venons de voir que la nature des ateliers flexibles avait pour conséquence une approche apériodique de la surveillance dans le cadre d'une politique de maintenance curative. Toutefois, nous devons nuancer ce résultat. En effet, l'analyse des moyens de production utilisés dans l'industrie manufacturière montre que certaines défaillances (comportements anormaux) se manifestent par des phénomènes continus (vibrations, bruits, élévation de température, croissance anormale de la puissance consommée). Ces types de défaillances peuvent être détectés par les techniques classiques de la maintenance prédictive grâce à l'ajout de capteurs spécifiques à la surveillance (capteurs de surveillance).

Nous pensons donc que les deux types de maintenance (prédictive et curative) doivent être mises en œuvre de manière complémentaire dans les ateliers flexibles. Ceci s'explique d'autant plus qu'on n'a pas la même vision du système, selon le niveau de perception auquel on se situe. Au niveau auquel se situe la conception de la commande, la vision est macroscopique et l'objet est d'organiser le séquençage et l'agencement des différentes opérations entre elles. C'est ce niveau macroscopique qui permet de classer les SFP dans la catégorie des SED. Si l'on s'intéresse par contre à la commande des éléments de la partie opérative, le niveau d'approche devient microscopique et de nombreux éléments doivent être pilotés de façon continue. Cela montre le changement de classe et l'opportunité d'une maintenance prédictive.

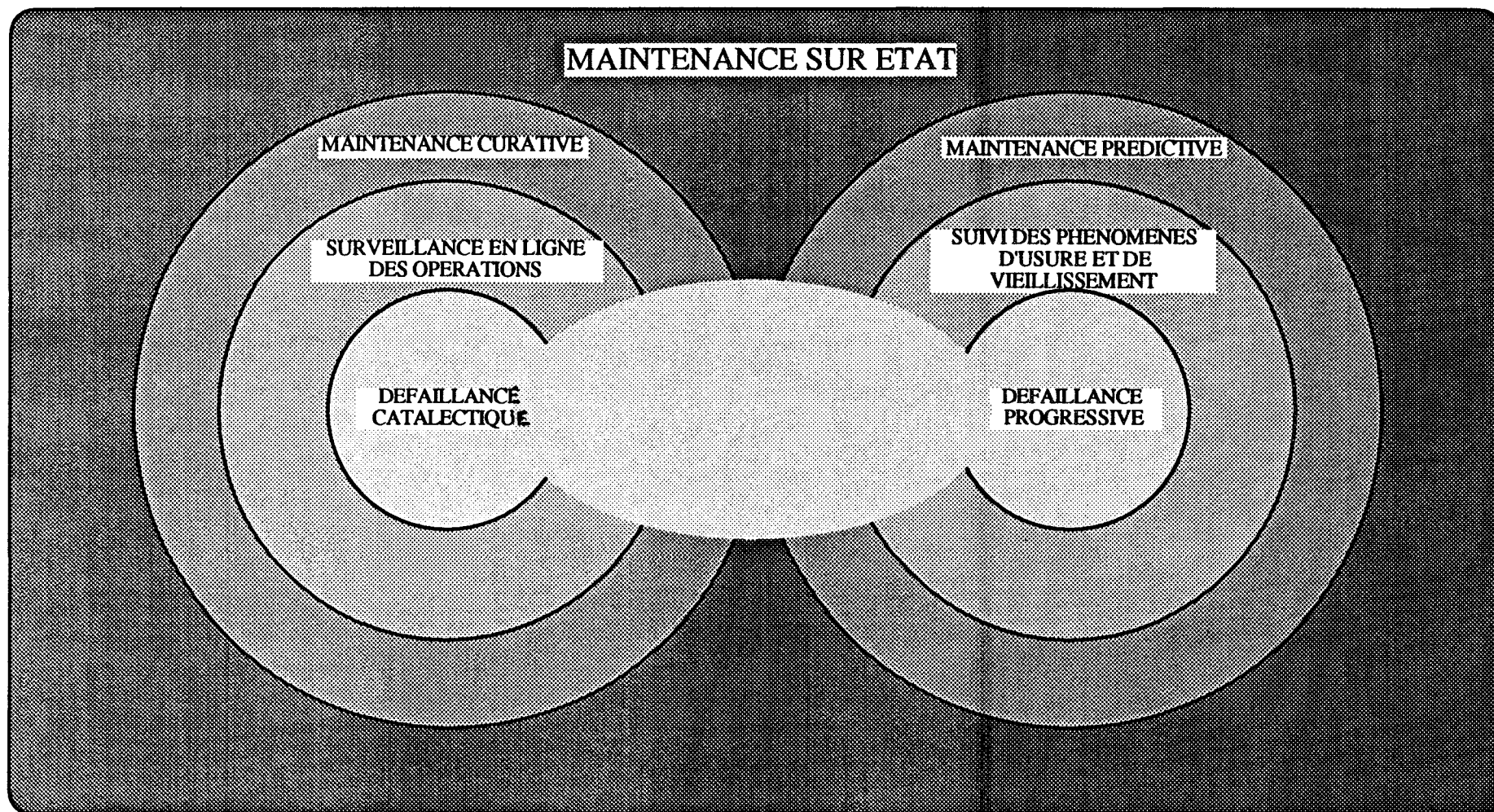


Figure I.25 : Schéma de synthèse relatif à la maintenance sur état du procédé

III. STRUCTURATION DE LA SURVEILLANCE DANS LE CONTEXTE DES ATELIERS FLEXIBLES.

III.1 Rappels sur nos hypothèses de travail.

Notre but est d'aider la conception de systèmes de surveillance en ligne pour les ateliers flexibles. Afin de faciliter la compréhension de notre approche, nous allons poser les hypothèses de base sur lesquelles s'appuie notre travail :

H1 - Dans le contexte SED, le système de surveillance s'intègre à une stratégie de maintenance curative ;

H2 - La commande est saine ;

H3 - Le système de surveillance a une connaissance discrète de l'état du procédé ;

H4 - La surveillance est réalisée en ligne et en temps réel.

De ces hypothèses, on peut déjà tirer un certain nombre de conséquences :

C1 - On peut utiliser les capteurs de contrôle du séquençement de la commande à des fins de surveillance.

C'est une conséquence de H3. Cette conséquence est très importante dans notre approche car elle nous permettra de minimiser l'ajout de capteurs de surveillance.

C2 - L'information donnée par tout capteur dédié à la surveillance doit être discrète.

En pratique cela signifie que même si on ajoute des capteurs à signaux analogiques ou numériques, l'information délivrée sera transformée sous forme binaire (normal, anormal) par utilisation de la notion de seuil (Fig. I.26). Afin d'éviter les fausses alarmes dues à des seuils fixés quelle que soit la configuration du procédé, nous supposons que les seuils varient en fonction des paramètres caractéristiques de l'entité ou du processus surveillé.

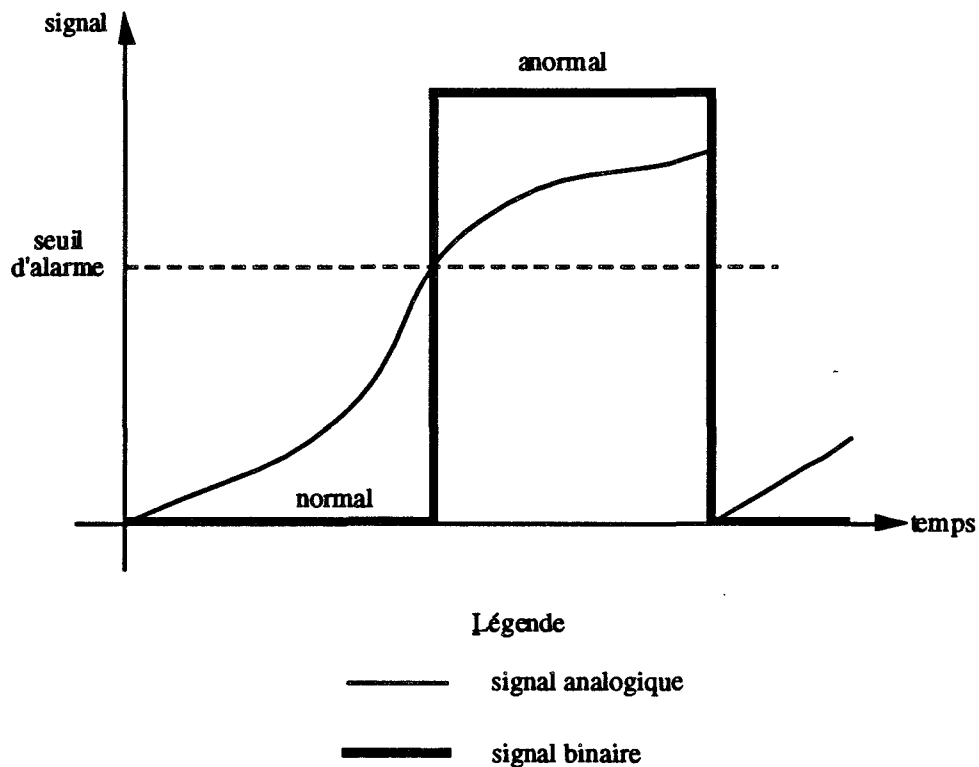


Figure I.26 : Discrétisation binaire d'un signal analogique .

C2 est une conséquence de H3.

C3 - La surveillance d'opération est une surveillance indirecte du procédé.

C'est une conséquence basée sur l'hypothèse H2 de cohérence de la commande.

C4 - Le système de surveillance doit d'une part identifier la cause de toute défaillance et d'autre part anticiper toutes les erreurs latentes dues à cette défaillance.

C'est une conséquence de H1.

C5 - Le système de surveillance doit permettre de réaliser un bouclage automatique de type régulation entre le procédé et la PC.

C'est une conséquence de H4 et C4. En effet pour anticiper le déclenchement d'erreurs latentes, il faut que la PC connaisse les sous-ensembles défaillants du procédé, ce qui implique le bouclage.

La conséquence C5 induit une conséquence C6.

C6 - Les informations transmises par le système de surveillance à la commande doivent être des faits certains.

Cette conséquence s'explique par le fait que la production planifiée ne peut être réduite que pour des raisons de sécurité, établies par le diagnostic certain d'une défaillance et de ses conséquences.

III.2 Structuration de la surveillance dans le contexte des ateliers flexibles.

Un certain nombre de travaux se sont intéressés ces dernières années, à une structuration de la surveillance dans le contexte des ateliers flexibles. Nous pensons notamment aux travaux du LAAS-CNRS de Toulouse et à ceux du CRAN/LACN de Nancy.

III.2.1 Approche du LAAS.

L'approche du LAAS est caractérisée par une vision hiérarchique d'une surveillance intégrée à la commande. Ainsi, à chaque niveau d'abstraction de la commande correspond un module de surveillance. L'approche est du type surveillance des opérations dénommées ici "Activités" [COM91]. Chaque activité est modélisée par une primitive générique (Fig. I.27) basée sur l'utilisation des Réseaux de Petri Objets (cf. Annexe B) [SIB88].

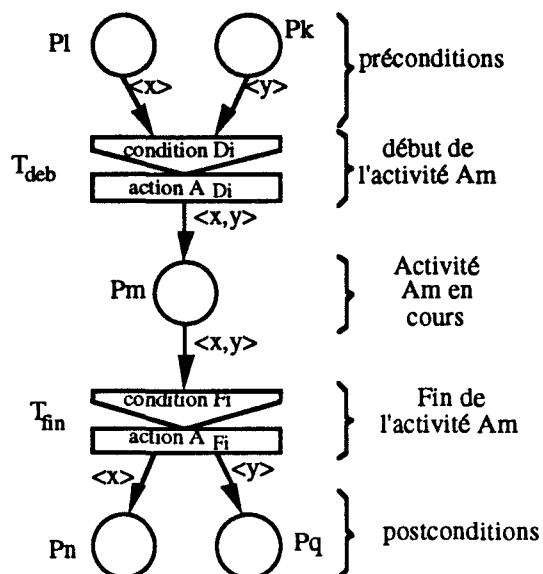


Figure I.27 : Modélisation d'une activité par RdPO.

Le principe de surveillance peut être schématisé par le raisonnement suivant :

Si une activité est défaillante, c'est que l'une des conditions nécessaires à sa réalisation n'était pas bonne ou l'une des préconditions était fausse (propagation des défaillances). Les conditions et préconditions (cf. figure I.27) concernent les éléments du procédé : produits, outils, moyens de production, moyens de transport.

La démarche consiste alors à déterminer à partir de quelle activité antérieure s'est propagée la défaillance. Le système de surveillance déroule donc le "scénario" de séquençement des activités dans le sens inverse de celui réalisé par la commande.

Lorsqu'une activité est défaillante, le niveau de commande-surveillance supérieur en est averti.

L'approche du LAAS est caractérisée par deux concepts de base : la **surveillance intégrée** et la **surveillance séparée** [SAH87].

La surveillance intégrée consiste à mettre en œuvre les fonctions de la surveillance au sein même de la commande. J. Cardoso a étendue ce concept en introduisant la notion de **surveillance tolérante** [CAR90]. Il s'agit d'intégrer au niveau de la commande dite "normal", le comportement anormal du procédé. De cette manière on le gère comme le comportement normal. Pour cela elle a conçu un nouvel outil pour la spécification de la commande des ateliers flexibles : les RdPO temporels à marquage flou. Dans la quatrième partie de ce chapitre, nous utiliserons les RdPO temporels pour spécifier le mécanisme des filtres de détection.

La surveillance séparée est basée sur un principe de séparation stricte entre cette fonction et la commande de coordination (le séquentiel). La surveillance est alors conçue comme une fonction du Niveau Hiérarchique.

Sur cette base, l'équipe du LAAS distingue six fonctions essentielles pour mettre en œuvre la surveillance [COM90], [COM91] :

*** La détection.**

Elle est intégrée à la commande. Elle est mise en œuvre par la technique du chien de garde. Dans un module, deux situations de défaillance peuvent être détectées :

- une activité n'a pas encore été lancée à **DEB_{tard}** (Date de lancement au plus tard) car toutes ces préconditions ne sont pas vraies ; cet état correspond à un manque de ressources ;
- une activité n'a pas été acquittée à la date de fin au plus tard (**Fint_{tard}**) ; cet état correspond à un dysfonctionnement.

Les dates **DEB_{tard}** et **Fint_{tard}** sont déterminées à partir du plan de fabrication.

*** Le diagnostic.**

Il est séparé de la commande. Il est mis en œuvre par un système à base de règles. Il est déclenché automatiquement sur détection d'une défaillance. Il établit sa base de faits à partir de l'image du procédé vue de la commande au niveau considéré. Aussi, il ne peut déterminer que les défaillances des entités du procédé modélisées au niveau de la commande. En cas d'échec ou d'insuffisance de données sur le procédé, il fait appel à l'opérateur.

*** La décision.**

Comme le diagnostic, elle est séparée de la commande et est mise en œuvre par un système à base de règles. Ces fonctions sont :

- la détermination d'un point de reprise,
- la détermination d'une séquence de reprise,
- le déclenchement d'une procédure d'urgence en cas de nécessité,
- le passage en mode interactif pour dialoguer avec l'opérateur,
- la propagation du traitement des défaillances vers le niveau supérieur.

*** La reprise.**

C'est un ensemble de procédures intégrées au système de commande.

*** Les procédures d'urgence.**

Elles permettent de déclencher en priorité certaines activités nécessaires pour garantir la sécurité.

Une des caractéristiques de l'approche du LAAS, relève de la structuration de la commande d'un niveau, en deux modules indépendants :

- le **module de commande** : il gère le séquençement des activités sans s'occuper des contraintes inhérentes aux moyens de production ; la commande est vue ici au même niveau que les gammes logiques de CASPAIM-2.
- le **module de référence** : il modélise le comportement normal du procédé. Il modélise notamment la flexibilité potentielle du procédé restreinte uniquement par les contraintes de coopération, de synchronisation et d'exclusion mutuelle. Ce module sert également de base de données pour les inférences des modules de diagnostic et de décision. C'est à son niveau également que sont détectés les symptômes de défaillance.

Avec ces spécifications la structure d'un niveau de commande surveillance est donnée par la figure I.28.

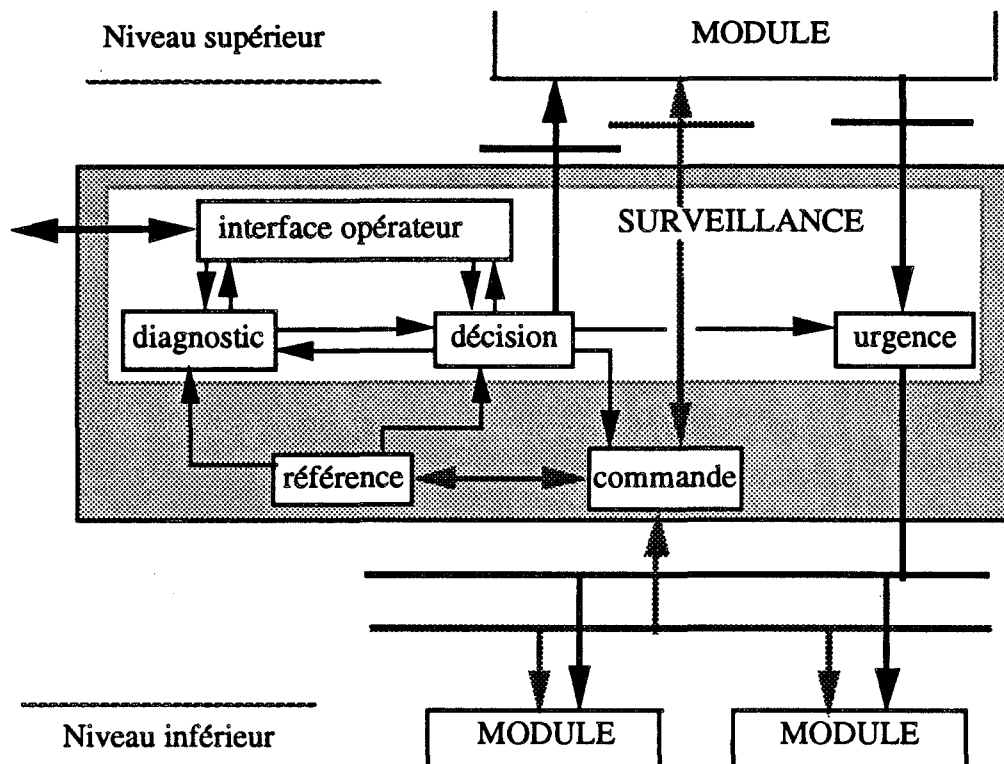


Figure I.28 : Structuration d'un niveau de commande-surveillance d'après M. COMBACAU [COM91].

L'approche du LAAS repose sur la surveillance des opérations. Elle est liée à une stratégie de pilotage consistant à fixer de manière définitive les dates des opérations du plan de fabrication de chaque produit. Un des inconvénients majeurs de cette approche se situe au niveau du diagnostic. Celui-ci repose sur l'hypothèse de propagation d'une défaillance sur plusieurs activités. Supposons par exemple qu'il y ait rupture de l'outil lors de l'opération de rainurage d'une pièce initialement tournée (Fig. 29). Lors du rainurage, les préconditions de l'activité consistent à vérifier informatiquement (i.e au niveau de la structure de donnée du jeton associé) que le diamètre de la pièce tournée correspond bien à la valeur prévue. En cas de rupture de l'outil de rainurage, on va réfuter les préconditions de l'activité. Aussi, le diagnostic consiste alors à déterminer quelle activité antérieure a pu aboutir à une précondition fausse pour l'activité courante, causant sa défaillance. Dans le cadre de cet exemple, la cause de la rupture serait un diamètre après tournage supérieur à celui prévu, en raison de l'usure de l'outil.

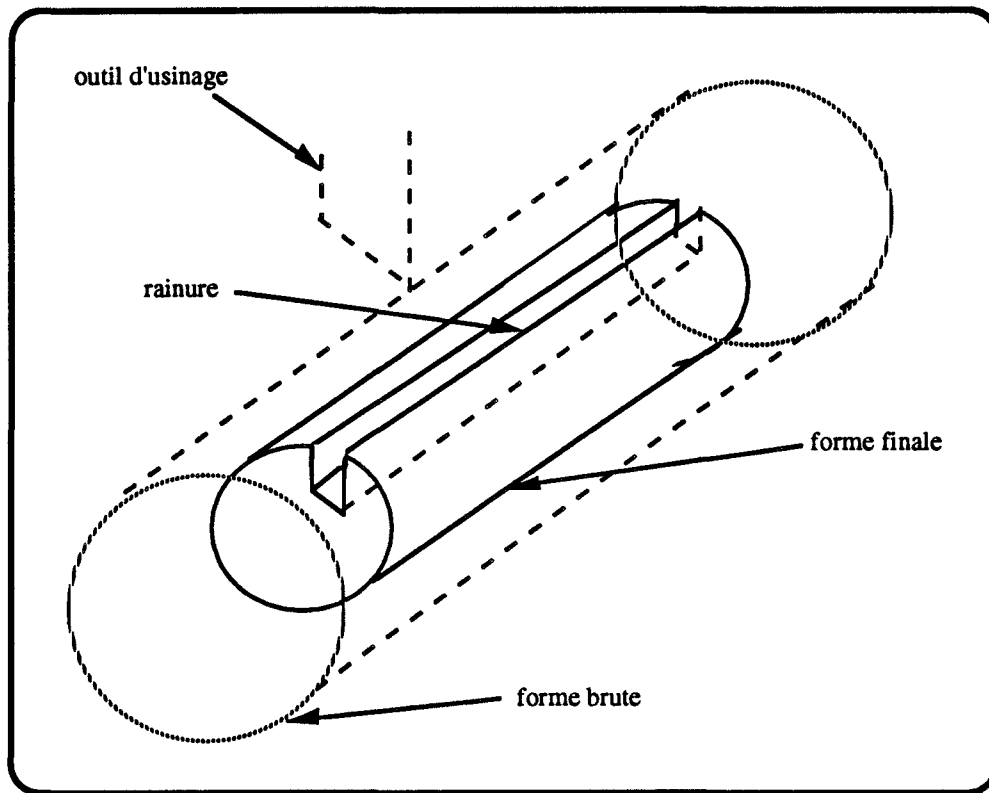


Figure 29 : Rainurage.

Il est important ici de préciser une différence essentielle entre cette approche et la nôtre. Dans cette approche, le diagnostic est lié à une propagation horizontale (i.e à un niveau structurel donné) des défaillances. Notre approche par contre repose sur l'hypothèse de propagation verticale des défaillances. C'est à dire que si une opération (activité pour le L.A.A.S.) est défaillante les causes sont locales à l'opération. Le diagnostic consiste donc en la recherche des causes dans l'ensemble des niveaux du procédé directement sollicité pendant l'opération. Ainsi, dans l'exemple précédent, notre approche de la surveillance nous aurait amené à **détecter** l'usure de l'outil de tournage pendant cette opération. Il n'y aurait donc pas eu de propagation horizontale entraînant la défaillance de l'opération de rainurage. Cette détection peut être réalisée "simplement" par le suivi des efforts de coupe pendant le tournage [BRU90], [CET85].

Une autre différence entre l'approche du L.A.A.S. et la notre réside dans ce qu'on diagnostique. Dans l'approche du L.A.A.S., le diagnostic est basé sur l'image du procédé modélisé dans le module de référence. Il n'y a pas de modèle autonome du procédé. De ce fait, lorsqu'une défaillance est détectée à un niveau donné de l'architecture commande-surveillance, le diagnostic est limité à la vision qu'on a du procédé à ce niveau. Supposons que nous nous situons au niveau coordination des moyens de production. Sur une machine donnée, nous

lançons par exemple le cycle de l'opération de rainurage. En cas d'anomalie, il y aura un dépassement du temps de cycle qui sera détecté par la commande de coordination. La vision qu'on a, à ce niveau du procédé va conduire à diagnostiquer une défaillance pendant l'opération de rainurage, due à la défaillance antérieure du système de tournage. Par contre, l'impossibilité d'interroger les niveaux inférieurs, ne permettra pas par exemple, de savoir si la défaillance est due à l'outil de coupe ou au système d'entraînement de cet outil. A la limite, en utilisant une connaissance surfacique basée sur les paramètres de la maintenance préventive (MTBF, ...) on peut **supposer** la défaillance pour usure de l'outil de tournage.

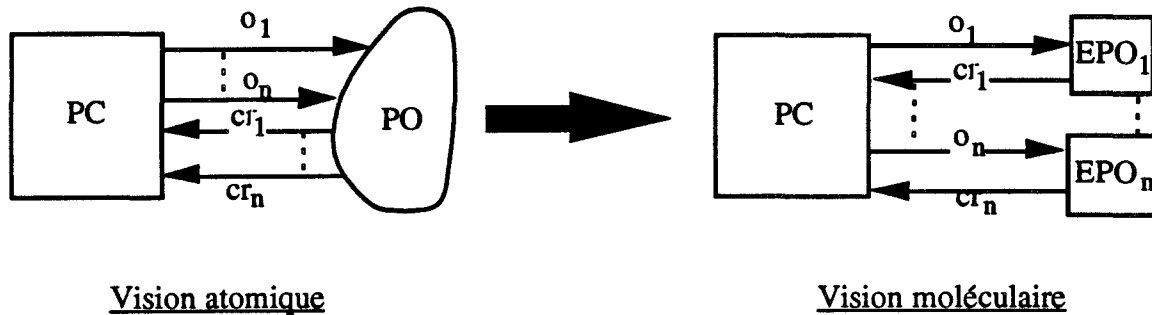
Dans la terminologie du diagnostic, on pourra dire que l'approche du LAAS permet une **localisation** des défaillances propagées horizontalement. Mais elle ne permet pas l'**identification** des causes de défaillances. Pour résumer, deux raisons permettent d'expliquer ce constat :

- * L'absence d'un modèle du procédé, spécifique au diagnostic, et basé sur une connaissance profonde. Bien sûr des données supplémentaires sur le procédé sont utilisées par le module de diagnostic. Mais elles sont disséminées dans des règles modélisant une connaissance superficielle du procédé [CHA89], [FIN85].
- * Le mécanisme de propagation du traitement des défaillances à travers les différents niveaux de la hiérarchie de commande-surveillance : c'est une propagation ascendante (mis à part les procédures d'urgence), liée au processus de recouvrement immédiat (cf. chapitre IV, § I.2.2.1). Au contraire les mécanismes d'identification (démarche consistant à aller du général au particulier) sont résolument descendants [CHA83].

Dans l'approche du LAAS, le module de référence tient lieu de base de données temps réel pour le diagnostic et la décision. Cela rend le système de surveillance fortement dépendant de leur mode de conception de la commande et du système de pilotage des ateliers flexibles.

III.2.2 Approche du CRAN/LACN.

Cette approche se situe dans le cadre général de la modélisation du comportement des Eléments de la Partie Opérative. Le fondement de base consiste à remplacer la vision "atomique" classique de l'interface PC-PO, par une vision "moléculaire" [ALA86]. C'est à dire que l'on appréhende la PO comme constituée d'un ensemble de modules élémentaires : les E.P.O (Fig. I.30). Chaque EPO est caractérisé par un ensemble d'ordres et de compte-rendus disjoints de ceux des autres EPO.



Légende :

$o_i \rightarrow$ ordre n° i
 $cr_i \rightarrow$ compte-rendu n°i

Figure I.30 : Passage d'une vision atomique à une vision moléculaire de l'interface PC-PO.

L'idée est alors de se dire qu'il existe une "relation sémantique" entre les ordres et les comptes-rendus d'un EPO : c'est le comportement de l'EPO.

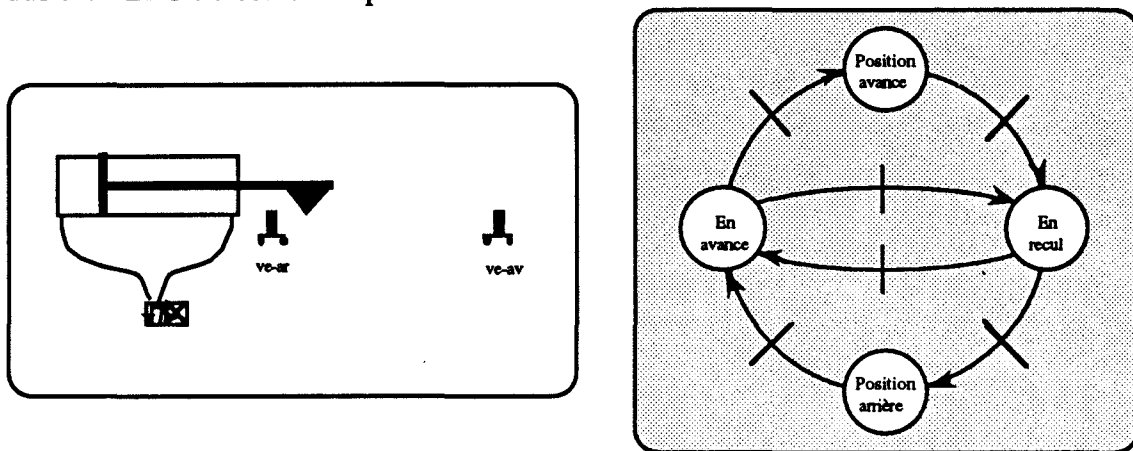


Figure I.31 : Modèle du comportement normal d'un vérin double-effet.

Considérons le vérin représenté sur la figure I.31. Il peut exécuter deux actions : l'avance et le recul. Ne disposant que de deux capteurs de fin de course (ve-av et ve-ar) ces deux actions seront caractérisées par quatre états du vérin : en avance, en position avant, en recul, en position arrière. On supposera par ailleurs, qu'avant la fin d'une action, on peut lui demander d'exécuter une action contraire. Avec ces spécifications on obtient le graphe de la figure I.31, modélisant le **comportement normal** du vérin.

Les modèles comportementaux des EPO peuvent être utilisés dans deux cadres : en émulateurs et en filtres.

Le principe de l'émulation consiste à générer par le modèle, les comptes-rendus de l'EPO à partir des ordres émis par la PC. Nous avons représenté ce principe sur la figure I.32. En émulation on introduit des temporisations pour contrôler l'évolution à partir de certains états.

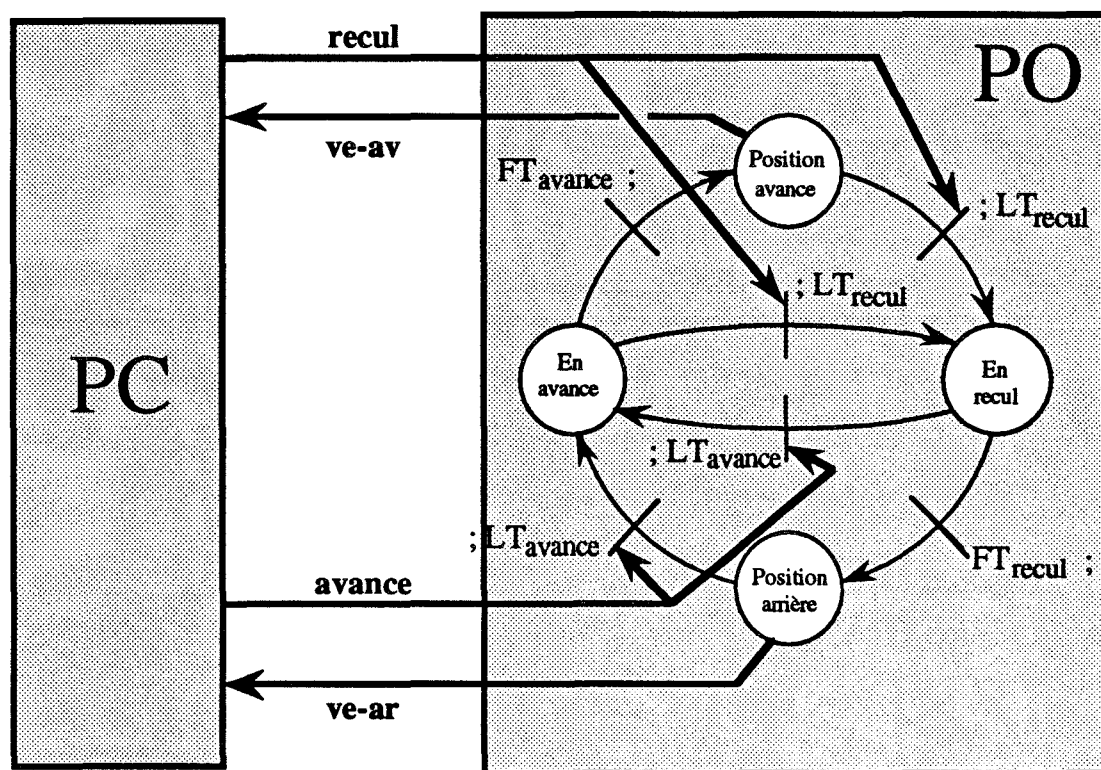


Figure I.32 : Utilisation en émulation.

Positionné en filtre entre la PC et la PO, le rôle d'un modèle comportemental est double. D'une part contrôler la cohérence des ordres émis et de l'état réel de la PO : c'est le concept du filtre de commande. D'autre part, vérifier en retour que les comptes-rendus reçus correspondent à ceux attendus. Dans les deux cas si les signaux reçus ne correspondent pas aux prévisions, le filtre génère des erreurs qui sont traitées au niveau de la Partie Commande. Dans un souci de lisibilité, nous n'avons pas représenté tous les cas d'erreurs sur le modèle de filtre de la figure I.33.

Modèle EPO

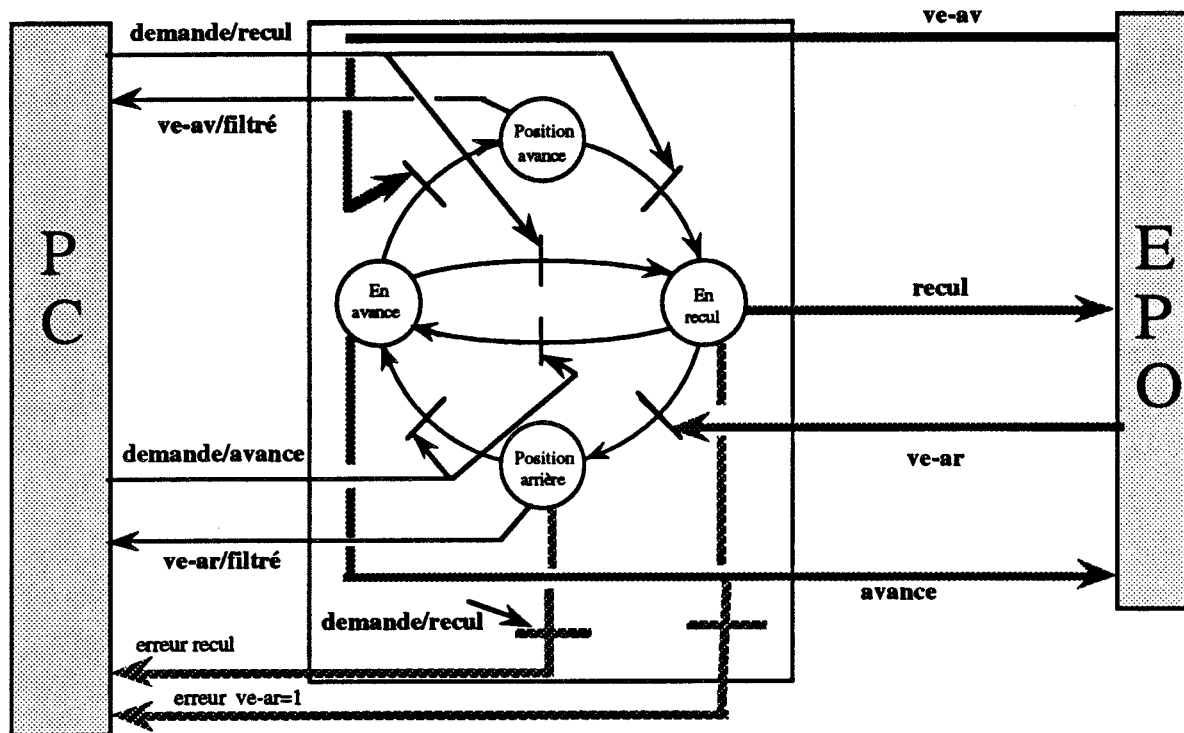


Figure I.33 : Surveillance par prévision du comportement.

Avec le concept du filtre, la PC est structurée comme représenté sur la figure I.34.

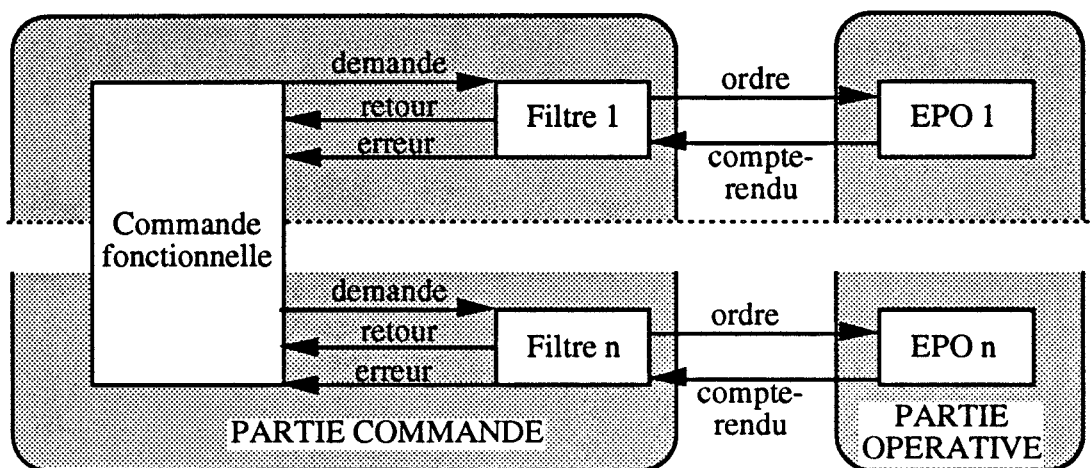


Figure I.34 : Structuration de la PC d'après [LHO91].

La notion de filtre apporte beaucoup en fiabilité et sécurité. Dans [SFA89], les auteurs ont montré par une étude numérique comparative que les filtres permettent de réduire de près de 75% le temps de détection d'une défaillance.

Ces résultats ont conduit au développement d'outils de CAO pour la manipulation de modèles de comportement en émulation ou en filtre (ADELAIDE, SPEX).

Quoique très intéressante la notion de filtre proposée par le CRAN/LACN reste localisée au niveau le plus bas de procédé : la Partie Opérative ou PO. Au LAIL, nous cherchons à exploiter la notion de modèle comportemental, dans un contexte plus large. Concernant la validation par émulation les idées du modèle comportemental ont été reprises par S. Amar dans le cadre d'une modélisation de l'ensemble du procédé [AMA90]. La notion de filtre de commande a également été reprise par D. Cruette dans le cadre de la conception de l'interface entre la commande et le procédé [CRU91].

D'autre part, nous pensons que l'idée du filtrage des comptes-rendus dans un but de surveillance peut être étendu à tous les niveaux d'abstraction commande-procédé. Mais pour conserver au filtre son caractère "réactif", il faut le concevoir à partir d'un modèle de comportement simple. Or aussi bien l'approche du CRAN/LACN que celle de D. Cruette, ne permet pas de conserver cette simplicité en raison de l'explosion du nombre de places des graphes d'états des sous-ensembles complexes du procédé.

III.2.3 Proposition de structuration dans le cadre de la méthodologie CASPAIM.

Dans notre approche nous nous sommes inspirés des travaux de structuration de la surveillance proposés par A.E.K Sahraoui [SAH87] et du concept de filtre introduit par le CRAN/LACN.

Nous distinguons trois fonctions principales dédiées à la surveillance d'un atelier flexible (Fig. I.35) :

*** La détection.**

Cette fonction met en œuvre la tâche de **filtrage-perception**. Elle est bâtie à partir de la connaissance du modèle temporel reliant les comptes-rendus de chaque entité commandée, aux ordres de la PC. La détection est une **tâche réactive** positionnée entre la Partie Commande et le procédé. Elle joue un rôle de **filtre des comptes-rendus** émis par le procédé. Si un

compte rendu reçu correspond à un compte-rendu attendu, alors le compte-rendu filtré est émis vers la PC. Sinon un symptôme est détecté.

Le système de détection établit aussi une **classification des symptômes en fonction de leur gravité**. Si un symptôme est jugé dangereux (symptôme potentiel d'une défaillance critique ou catastrophique) il est émis simultanément vers les modules de diagnostic et de recouvrement. Sinon ils sont uniquement émis vers le module de diagnostic.

*** Le diagnostic.**

Le diagnostic est décomposé en trois phases : la **localisation**, l'**identification** et le **pronostic**. La localisation consiste à déterminer le sous-ensemble défaillant. Une fois la localisation effectuée, le rôle de l'identification est de déterminer le composant défaillant. Cette identification précoce permet de planifier la maintenance du sous-ensemble défaillant dans le cadre du recouvrement. La phase de pronostic est déclenchée chaque fois qu'un composant a été identifié comme étant défaillant. Le but est de prédire les pannes induites par la défaillance. Le pronostic tel que nous le concevons ici est en fait une phase de **post-recouvrement** utilisé dans le cadre de l'évitement des fautes.

*** Le recouvrement.**

C'est la fonction la plus liée à notre approche de la conception de la commande des ateliers flexibles. Le concept de base utilisé ici est la flexibilité : **flexibilité de gammes, flexibilité des moyens de production et flexibilité des moyens de transport**. A partir de ces trois flexibilités nous proposerons une structuration de la commande en couches permettant de gérer les mécanismes classiques du recouvrement : le **gel**, la **reprise** et la **reconfiguration dynamique**. Notre conception du recouvrement intègre les résultats de S. Bois [BOI91/b] sur la gestion des modes de marches des ressources d'un SFP.

Les fonctions diagnostic et recouvrement sont "interfacées" avec le **module opérateur**. Le diagnostic notamment pourra dans un nombre de cas limité, utiliser l'opérateur comme capteur de surveillance. En cas d'échec, l'opérateur interviendra pour résoudre le problème de diagnostic à partir des données brutes que lui transmettra le module de diagnostic. Une fois le diagnostic établi, l'opérateur aura pour rôle d'enrichir la connaissance de base du module de diagnostic et d'enrichir la base de données du gestionnaire des modes de marche [BOI91/b].

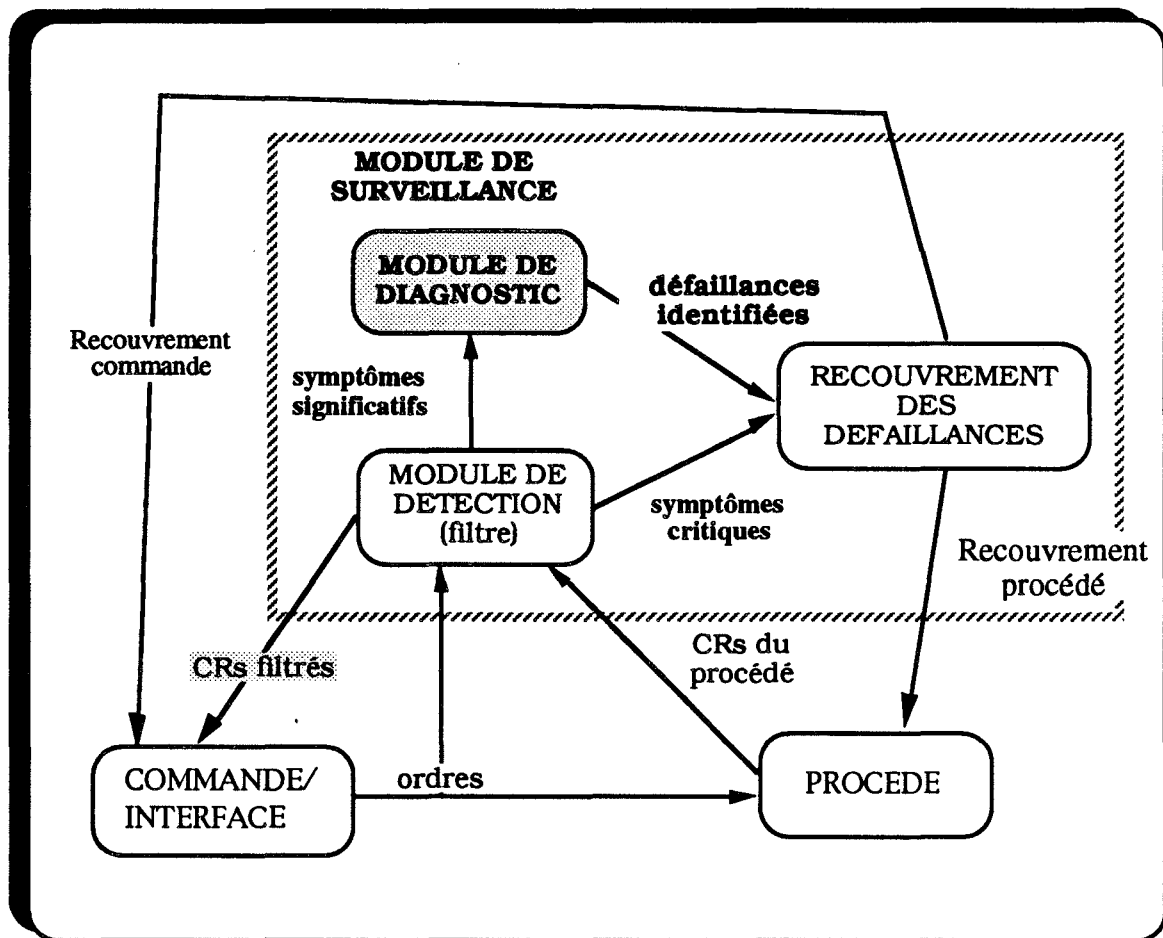


Figure I.35 : Structuration de la fonction surveillance et positionnement par rapport à la PC et au procédé.

Le positionnement de la surveillance du procédé dans le niveau hiérarchique dès le § I.2.3, illustre une des idées fortes de notre approche : la surveillance du procédé est une **fonction intelligente** exigeant parfois une vision d'ensemble du sous-ensemble défaillant. Cette approche de la surveillance du procédé en tant que fonction intelligente explique le choix d'une approche séparée de la commande. En effet, **une fonction intelligente ne peut pas être réactive**. Bien entendu, la sous-fonction détection n'est pas une fonction intelligente.

III.2.3.1 Le module de détection.

C'est le module soumis à la plus **forte contrainte temps réel**. En effet, de par sa position de filtre entre le procédé et la PC, il doit réagir en "temps nul" à toute réception de compte-rendu, de manière à ne pas ralentir le fonctionnement de la PC. En mode de fonctionnement normal, tout se passe comme s'il y avait une interaction directe, en retour, entre le pro-

céder et la PC. Dès qu'une défaillance est détectée la partie correspondante de **la commande est implicitement gelée** du fait de la non réception des comptes-rendus conditionnant le franchissement des transitions sensibilisées. Le gel de la commande ayant sollicitée les services d'éléments du procédé défaillant, permet d'empêcher la propagation d'anomalie par la commande elle même.

Le module de détection n'assure pas une fonction classique de détection. Cela tient au fait que nous manipulons des données binaires (capteur à l'état haut ou à l'état bas). Aussi sa fonction correspond plus à la perception des résidus nécessaires à la localisation. Nous appelons ces résidus, symptômes car ils indiquent implicitement que le système est défaillant. Il n'y a donc pas de tests d'hypothèses comme dans les méthodes continues.

Comme nous l'avons souligné dans les paragraphes antérieurs, la contrainte temporelle est plus forte dans les SED que dans les systèmes continus. Dans les systèmes continus, on peut l'estimer comme étant au pire de l'ordre de la minute. Dans les SED, elle croit au fur et à mesure que l'on descend dans le procédé. Au niveau atelier, elle est de l'ordre de la minute. Au niveau moyens de production, elle est de l'ordre de la seconde. Au niveau actionneurs, elle est de l'ordre du 1/10 de seconde. Aussi, dès la détection d'une défaillance, il faut réagir rapidement avant que la défaillance ne se propage au reste du procédé. Prenons l'exemple de la commande d'avance de l'axe Z (axe horizontal) d'un tour à broche horizontale. Supposons que le système d'asservissement en position de l'axe Z soit en panne (par exemple due à une défaillance du codeur optique). Supposons que pour une raison inconnue, le relais de sécurité de ce système n'ait pas été déclenché. La conséquence est que le porte-outils va entrer en collision avec la broche. Une solution consiste à discrétiser l'axe Z en positionnant des capteurs de position à intervalle régulier. En cas de dépassement du capteur positionné au delà du point commandé, il y aura détection du symptôme. Nous sommes alors dans un contexte d'urgence, ou l'on doit arrêter la machine avant que le diagnostic n'ait eu lieu.

La solution pour pallier ce type de situation consiste à inclure une fonction de classification des symptômes au sein du module de détection.

III.2.3.2 Le module de diagnostic.

C'est la fonction la plus intelligente du système de surveillance. Il est également soumis à une forte contrainte temporelle. En effet, la localisation des sous-systèmes défaillants, doit être assez rapide pour minimiser la propagation des défaillances par le procédé. Néanmoins à ce niveau la contrainte temporelle est beaucoup moins forte qu'au niveau détection. C'est ce qui nous permet d'envisager une approche intelligente basée sur un **raisonnement qualitatif**. Le

raisonnement qualitatif a pour avantage par rapport aux approches exclusivement numériques de nécessiter moins de calcul et donc d'être plus rapide. Dans [CHA89], B. Chandrasekaran en explique les raisons. Face à un problème complexe, l'être humain commence par raisonner qualitativement pour réduire la complexité. Ensuite, des méthodes numériques peuvent être appliquées localement pour bénéficier de leur précision.

Une tâche de diagnostic en ligne se doit d'être complète. Pour assurer cette complétude, nous avons bâti notre approche sur une modélisation fonctionnelle du procédé [CHA89]. Une telle approche de modélisation distingue trois points de vue fondamentaux de représentation du procédé :

- le point de vue structurel,
- le point de vue fonctionnel,
- le point de vue comportemental.

Au chapitre III, nous proposerons une méthode pour la modélisation fonctionnelle du procédé. Elle aboutit à la définition de deux modèles :

- le modèle structurel qui représente la topologie du système,
- le modèle fonctionnel qui modélise les liens de dépendance fonctionnelle entre les **fonctions principales** du système, et les **fonctions initiatrices**.

A partir du modèle fonctionnel, nous mettons en œuvre deux types de raisonnements :

- un raisonnement temporel qui est mené localement par rapport à chaque fonction du modèle,
- un raisonnement fonctionnel de nature hypothétique, et qui s'appuie sur le modèle global.

Dans le cadre du diagnostic, nous proposerons un **algorithme pour le positionnement des capteurs supplémentaires nécessaires au diagnostic**. L'idée directrice de ce positionnement, est qu'il n'est pas utile d'observer le comportement de chaque fonction. En utilisant les liens de dépendance et par là, la redondance analytique induite, nous pouvons limiter leur nombre au strict nécessaire.

III.2.3.3 Le module de recouvrement.

Le recouvrement a pour but d'assurer la sécurité du système et le maintien de la production à un niveau proche du fonctionnement optimal, en dépit des défaillances. Il met en œuvre des actions sur le procédé et sur la commande.

Le recouvrement du procédé intègre les directives exprimées par l'ADEPA dans le GEMMA (Guide d'Etude des Modes de Marche et d'Arrêt). Il est donc essentiellement basé sur

différents types d'arrêts des ressources : arrêt en mode de repli (arrêt en fin de cycle, arrêt dans un état déterminé, arrêt des mouvements, ...), arrêt d'urgence, ... L'étude des modes de marche et d'arrêt a été entreprise au LAIL par S. Bois [BOI91/b].

On a pour habitude de distinguer trois modes de recouvrement de la commande : le **gel**, la **reprise** et la **reconfiguration dynamique**.

Nous avons déjà indiqué au § III.2.3.1 que le filtrage réalisé par la fonction détection rend implicite le gel de la commande.

Concernant les problèmes de reprise, ils sont actuellement à l'étude dans les cadres des travaux sur la surveillance de la commande (cf. § II.2.2.2).

La reconfiguration dynamique de la commande constitue le principal apport que nous proposerons au chapitre IV. Notre démarche est fondée sur une approche fonctionnelle de la commande. L'idée directrice est d'exploiter les redondances fonctionnelles de l'atelier à tous ses niveaux de commande. Cela nécessite une décomposition en couche de la commande. La décomposition proposée s'inspire de celle proposée par l'OSI dans le cadre de l'interconnexion des systèmes informatiques hétérogènes (Norme ISO).

III.3 Conclusion

Notre démarche se situe complètement dans le cadre d'une surveillance séparée. Ce choix s'explique d'abord par notre volonté de concevoir un système de surveillance indépendant de la manière dont est conçue la commande séquentielle. C'est ce que nous avons réalisé pour la détection et le diagnostic. Ces deux fonctions s'appliquent aussi bien à une commande conçue selon les règles de CASPAIM-1 qu'à une commande conçue selon CASPAIM-2. De même, ces fonctions pourraient être utilisées telles quelles, pour surveiller un SFP conçu selon l'approche du LAAS (cf § III.2.1).

En fait, la fonction la plus dépendante de la nature de la commande, c'est le recouvrement. Cela est d'ailleurs compréhensible, car l'un des rôles de cette fonction est la modification dynamique de la commande. Dans le cadre de CASPAIM-2, nous proposerons au dernier chapitre, une méthode de conception de la commande visant à rendre transparente la gestion des modes de marche dégradés. En fait, notre démarche s'inspire de la philosophie de J. Cardoso [CAR90] : **rendre normal ce qui est normalement anormal**.

La séparation de la surveillance offre d'autres avantages. Cela permet notamment de décharger le programme d'application dans la mesure où nous pouvons utiliser des processeurs

spécifiques à la surveillance. La séparation apporte donc un plus du point de vue pilotage global du SFP en temps réel.

La séparation permet également de faciliter la maintenance logiciel .

Enfin, le dernier avantage de la séparation, réside dans le fait qu'elle laisse une liberté totale au concepteur concernant la mise en œuvre de la surveillance sur site. Elle peut être conçue selon une **approche centralisée** ou selon une **approche répartie et hiérarchique**. La difficulté de la mise en œuvre répartie est que le concepteur doit gérer lui-même la synchronisation de toutes les horloges des différents processeurs de surveillance. Malgré cette difficulté, c'est l'approche que nous recommandons pour garantir une surveillance en temps réel.

IV. DÉTECTION

IV.1 Introduction

Le module de détection doit assurer deux tâches principales : le filtrage-perception des comptes-rendus du procédé et la classification des symptômes détectés.

IV.2 Etat de l'art.

Les Systèmes à Événements Discrets sont caractérisés par une évolution discrète : c'est à dire que l'application d'une commande discrète, les font évoluer d'un état identifiable vers un autre état identifiable. Aussi, la commande des SED repose sur l'application de concept de sûreté de séquence (cf. § II.2.2.1.3). C'est à dire qu'elle est représentée sous la forme de séquences d'opérations appliquées dans chaque état identifiable du procédé. L'évolution de la commande est contrôlée par le test des comptes-rendus émis par le procédé.

De ce fait, les méthodes classiques de détection de défaillances dans les SED, utilisent toutes la technique du chien de garde. Elle consiste à temporiser chaque opération du graphe de commande (Fig. I.36). Cette temporisation nécessite la connaissance de la durée maximale de chaque opération. Deux cas de figure doivent être envisagés pour évaluer cette durée :

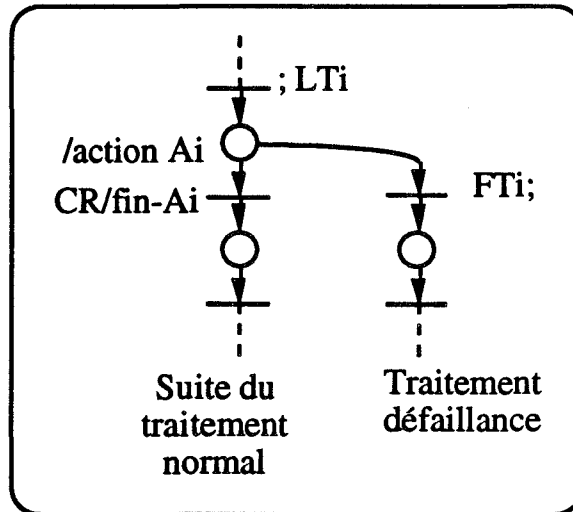


Figure I.36 : Technique du chien de garde.

* L'opération est une action simple réalisée par un actionneur.

La durée d'opération tout ou rien comme l'ouverture d'une butée est facilement évaluable à partir des paramètres caractéristiques de l'actionneur. Prenons l'exemple d'un vérin. Il est caractérisé par les grandeurs suivantes qui sont supposées connues :

- P : pression appliquée au vérin,
- S : surface du piston,
- L : longueur du vérin,
- m : masse du piston,
- V : vitesse moyenne du piston supposée.

La durée d'avance du piston est alors calculée par la relation :

$$\Delta T = \frac{m \cdot V}{P \cdot S} \quad (I.8)$$

* L'opération est une macro-action.

Nous appelons macro-action une opération décomposable en plusieurs actions situées à un niveau inférieur de commande. Si la macro-action est composée uniquement d'opérations en séquence, alors sa durée est égale à la somme des durées élémentaires. S'il existe des indéterminismes, sa durée est égale au maximum des durées opératoires des différentes combinaisons de tir possibles.

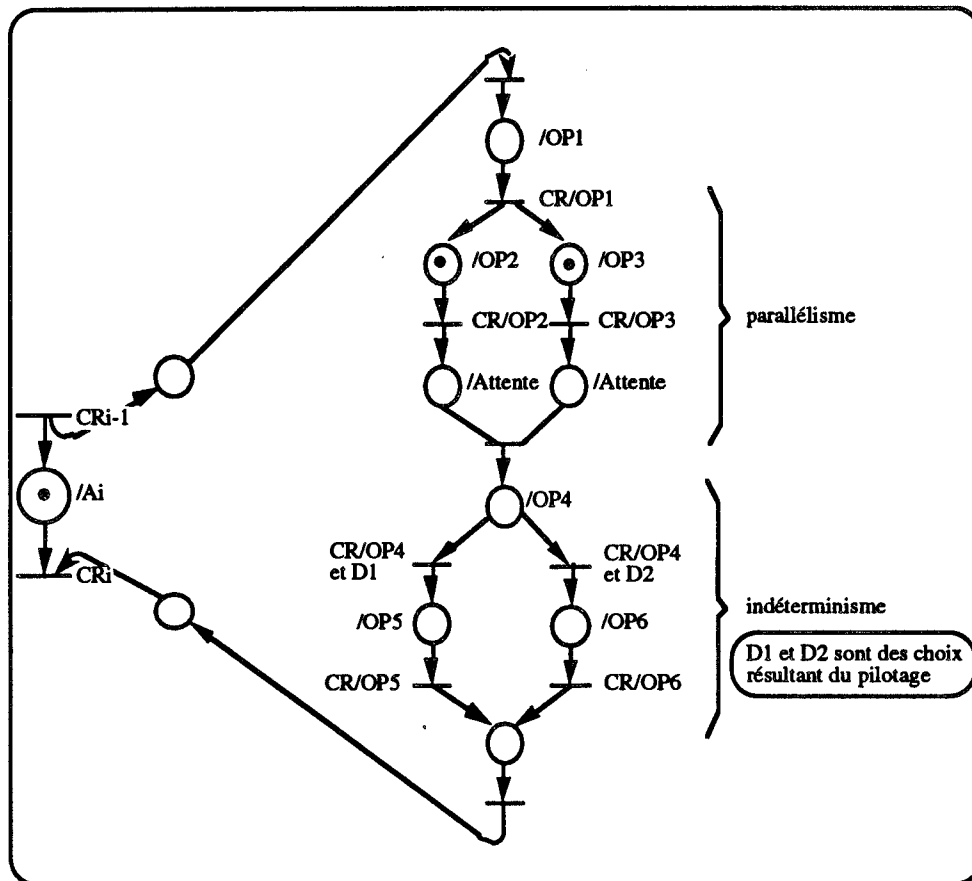


Figure I.37 : Représentation d'une macro-action.

Prenons comme exemple la macro-action représentée par la figure I.37. A chaque opération OP_i correspond une durée notée T_i . La durée maximale de la macro-action est :

$$\Delta T_M = T_1 + \max(T_2, T_3) + T_4 + \max(T_5, T_6) \quad (I.9)$$

Notons qu'à cette macro-action peut également être associée une durée minimale donnée par la relation :

$$\Delta T_m = T_1 + \max(T_2, T_3) + T_4 + \min(T_5, T_6) \quad (I.10)$$

L'utilisation de la technique du chien de garde est un mécanisme simple, permettant de détecter facilement des comptes-rendus absents. Mais cette technique présente cependant de nombreux inconvénients :

*** Elle est contraire aux principes du génie automatique.**

En effet, la technique du chien de garde est souvent associée à l'intégration au sein de la commande même, des mécanismes de recouvrement des défaillances. Cela rend la commande difficilement maintenable dans la mesure où le graphe résultant peut avoir un volume double de celui correspondant à la commande du fonctionnement normal. Par ailleurs en cours d'exploitation, l'enrichissement des mécanismes de recouvrement passe forcément par un arrêt des modules concernés.

*** Elle nécessite l'ajout de nombreux capteurs dédiés à la surveillance.**

D'abord cet ajout n'est pas toujours possible pour des raisons techniques et de coûts. Mais surtout, il est désormais largement admis que dans les SED, les éléments tombant le plus souvent en panne, sont les capteurs. Selon le CETIM, 60% des défaillances sont dues aux capteurs. Cela signifie que si leur ajout augmente la sécurité, par contre on perd énormément en disponibilité.

*** Elle ne permet pas de détecter les comptes-rendus intempestifs (non prévus).**

Ces comptes-rendus sont importants car ils apportent des informations qui peuvent être complémentaires de celles données par la technique du chien de garde. Nous allons illustrer cela à l'aide de deux exemples :

Exemple I.3 : Ralentissement du mouvement d'avance du piston d'un vérin.

La conséquence, c'est que le temps effectif du mouvement sera supérieur au temps normal calculé par la relation I.8. Le déclenchement du chien de garde entraînera la suppression du jeton de la place modélisant la demande d'avance. Aussi, lorsque le compte-rendu surviendra, il ne sera pas pris en compte.

Exemple I.4 : Défaillance de l'aiguillage d'un convoyeur à bande.

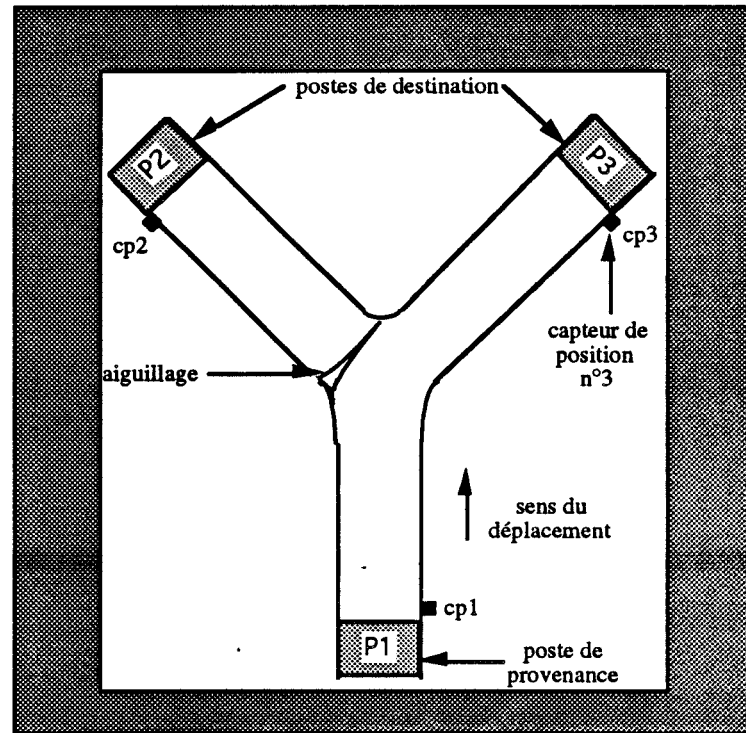


Figure I.38 : Convoyeur à bande.

Le système considéré correspond au schéma de la figure I.38. Supposons que l'on désire transférer un couple palette-pièce du poste source P1 vers le poste de destination P2. A la suite d'une défaillance de l'aiguillage, la palette se dirige vers le poste P3. La conséquence c'est qu'au lieu d'obtenir le compte rendu de cp2, on aura celui de cp3 qui ne sera pas pris en compte par la commande. En effet, nous pouvons remarquer sur la figure I.39, qu'un jeton est en attente dans la place P1-> P2/ TRSF_END, modélisant la confirmation par le capteur cp2, de l'arrivée de la palette transférée sur P2. Aussi tant que le capteur cp2 ne passera pas à 1 (détection du passage d'une palette), bien que sensibilisée, la transition en aval de cette place ne sera pas tirable. Les deux processus de transferts (P1->P2 et P2-> P3) sont en exclusion mutuelle (contrainte non représentée ici pour conserver la lisibilité du graphe). De ce fait, la transition en aval de P1->P3/TRSF-END n'est pas tirable même si sa réceptivité est vraie, car elle n'est pas sensibilisée.

Les comptes-rendus intempestifs peuvent également correspondre à des défaillances de capteurs. Non traités, ils peuvent alors entraîner des défaillances d'entités physiques du procédé. Considérons de nouveau le graphe de commande (Fig. I.20) traduisant l'opération de

chargement du § II.2.2.1.3. Le collage du capteur "pm" entraînerait le déclenchement de l'opération de serrage avant la fin du transfert. Il y aurait donc collision entre la pince du robot et le mors de tour.

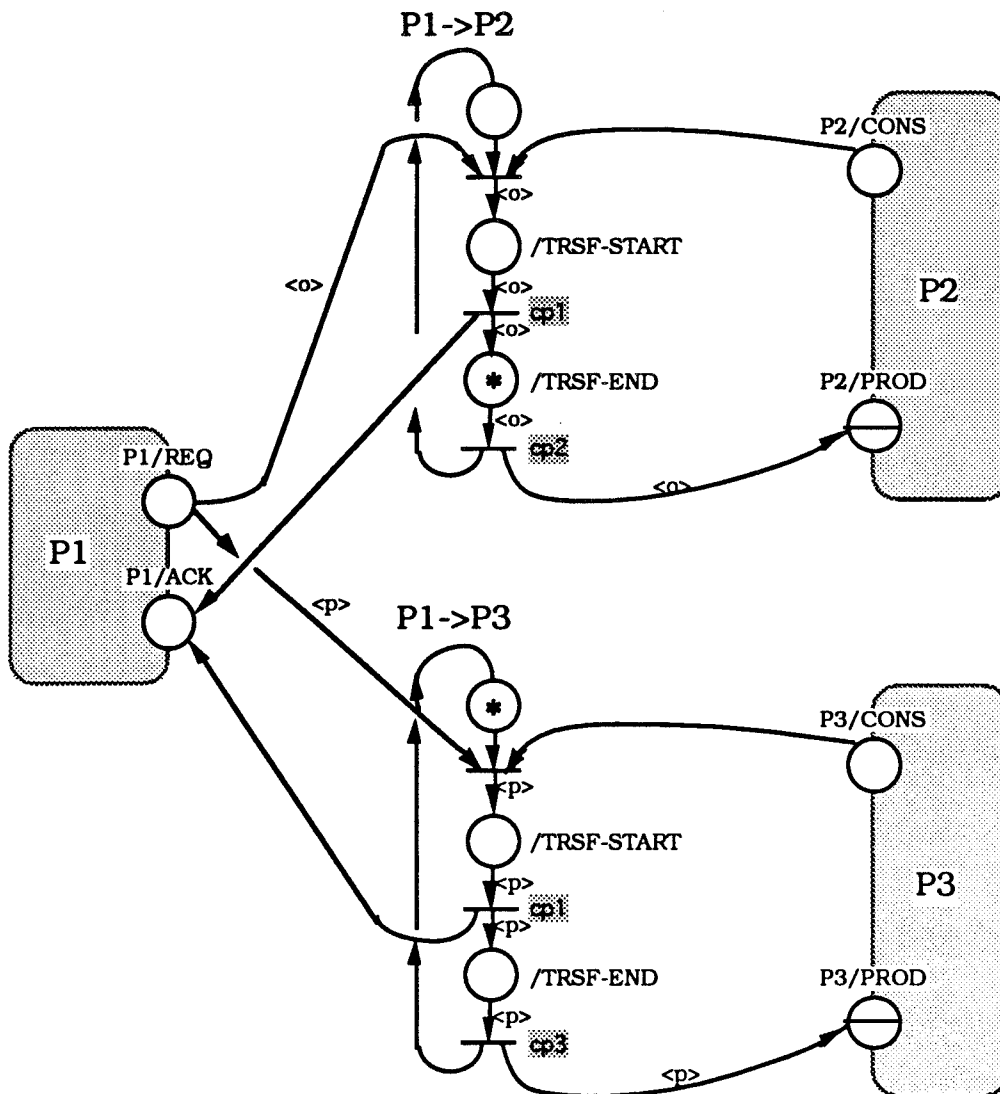


Figure I.39 : Modélisation de la commande de 2 transferts exclusifs d'après CASPAIM-1.

Pour éviter la propagation de défaillance due aux comptes-rendus intempestifs, il est possible d'imposer des durées de sensibilisation minimale aux transitions en aval d'une place opération. On utilise alors les durées opératoires minimales ΔT_m . Certains constructeurs d'API comme Télémécanique, ont intégré cette solution dans leur offre logiciel (notion de temps minimal d'activité d'une étape). Mais l'utilisation n'est liée à aucune méthode de conception de la surveillance.

IV.3 Principe du filtrage des comptes-rendus.

Le principe utilisé pour implanter le filtrage des comptes-rendus est basé sur l'approche classique de la perception. Cela consiste à faire évoluer en parallèle du procédé, un modèle soumis aux mêmes ordres. Le modèle considéré représente le fonctionnement normal du procédé. A tout instant les comptes-rendus issus du procédé (CRr) sont comparés à ceux issus du modèle (CRm) (Fig. I.40).

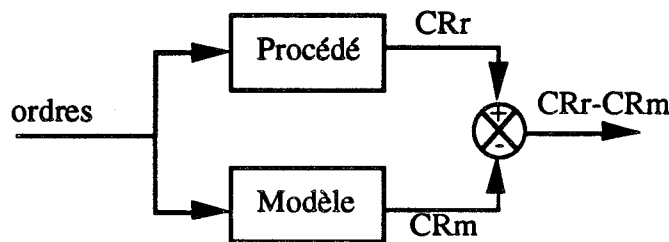


Figure I.40 : Schéma de principe de la fonction filtrage-perception.

Nous distinguons trois cas :

- * $CRr - CRm = 0$: nous sommes dans un cas de fonctionnement normal ;
- * $CRr - CRm = 1$: le procédé a émis un compte-rendu intempestif ;
- * $CRr - CRm = -1$: le compte-rendu du procédé n'est pas présent à la date prévue.

On voit qu'avec cette approche, la détection d'une défaillance est implicite. Elle correspond à $CRr - CRm \neq 0$.

IV.4 Modèle des opérations dans le cadre de la détection [TOG90], [TOG91].

IV.4.1 Modèle temporel associé.

On appelle modèle, toute représentation d'un système ou d'un processus. On distingue deux types de modèle [BRU90] :

- * Les modèles de représentation type boîte noire.

Ils sont généralement caractérisés par leurs variables externes (variables d'entrées et de sorties). Ils se contentent de modéliser une causalité entre variable de sortie et variables d'entrée.

*** Les modèles de connaissances.**

En plus des caractéristiques des modèles de représentation, ils modélisent le comportement interne du système.

Nous reviendrons plus en détail sur la notion de modèle au § III.2 du chapitre 2.

Pour la détection, un modèle de représentation est suffisant. Nous montrerons par contre que dans le cadre du diagnostic, il est nécessaire d'avoir un modèle de connaissance du système.

Ayant fait le choix de séparer détection et diagnostic, pour des questions de différence de réactivité, nous pouvons nous contenter d'utiliser un modèle de représentation pour mettre en œuvre la fonction filtrage-perception.

Au paragraphe § III.2.2, nous avons vu un exemple de modèle de représentation : les filtres introduits par le laboratoire du CRAN/LACN. Ce modèle étant toutefois limité à la partie opérative, nous avons cherché une approche qui puisse être facilement généralisée à la modélisation de toute opération (action, ou macro-action), quelque soit le niveau de commande. Aussi nous avons choisi de modéliser toute opération à partir d'une **approche temporelle**. A toute opération A_i , à laquelle est associée un compte-rendu CR_i , nous associons une **fenêtre temporelle** $[\Delta T_m, \Delta T_M]$. CR_i est valide uniquement à l'intérieur de la fenêtre. ΔT_m et ΔT_M sont des dates relatives à l'instant de déclenchement de A_i . Pour cette raison A_i est considéré comme un événement initiateur du processus de détection. Nous l'appellerons désormais "**start-event**".

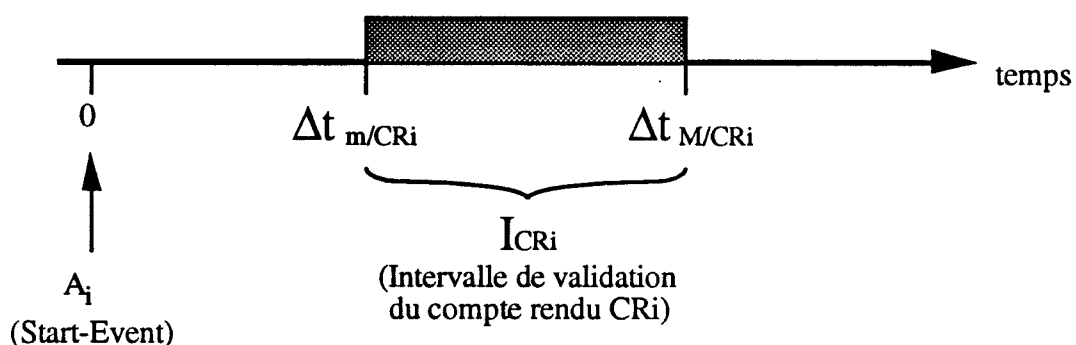


Figure I.41 : Modèle associé à une opération [TOG90].

Dans certaines conditions (disponibilité de capteurs dédiés à la surveillance d'une entité physique en particulier), la détection peut donner suite à un diagnostic immédiat de la cause de la défaillance. Cette approche nécessite de combiner les aspects détection-diagnostic en se basant sur l'exploitation d'un modèle de connaissance du procédé. A partir de l'expérience acquise sur les filtres de commande, le LAIL s'est intéressé à une telle approche. C'est l'objet des travaux de S. Elkhatabi [ELK92/b].

IV.4.2 Classes de symptômes.

L'utilisation de notre approche par modélisation temporelle des opérations permet de simplifier le schéma classique de la perception. En fait, nous proposons de comparer le comportement temporel du procédé à un modèle de référence, temporel, élémentaire, suffisant pour apporter l'information nécessaire à la détection. La comparaison a lieu directement dans le modèle du procédé (Fig I.42).

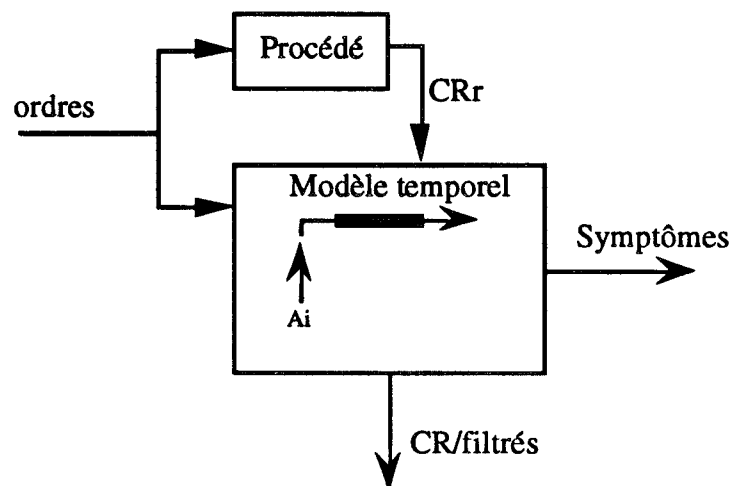


Figure I.42 : Filtrage à partir de la modélisation temporelle.

Sur la base du modèle temporel générique, nous définissons deux types de symptômes [TOG90], [TOG91] ayant chacun une sémantique particulière :

*** Les symptômes de type I notés S^1_{CRI} .**

Dans cette notation, CR_i désigne le compte-rendu qui a donné naissance au symptôme considéré. On notera ici que le symptôme est caractérisé par le capteur et non par l'action engagée A_i . Nous reviendrons ultérieurement sur cette particularité.

Cette classe de symptôme correspond à un compte-rendu attendu et non reçu à $\Delta T_{M/CRi}$ (Fig. I.43), date donnée par rapport l'occurrence du start-event A_i . Le mécanisme de détection de ce type de symptôme correspond au mécanisme classique du chien de garde, mais mis en œuvre de manière séparée de la commande.

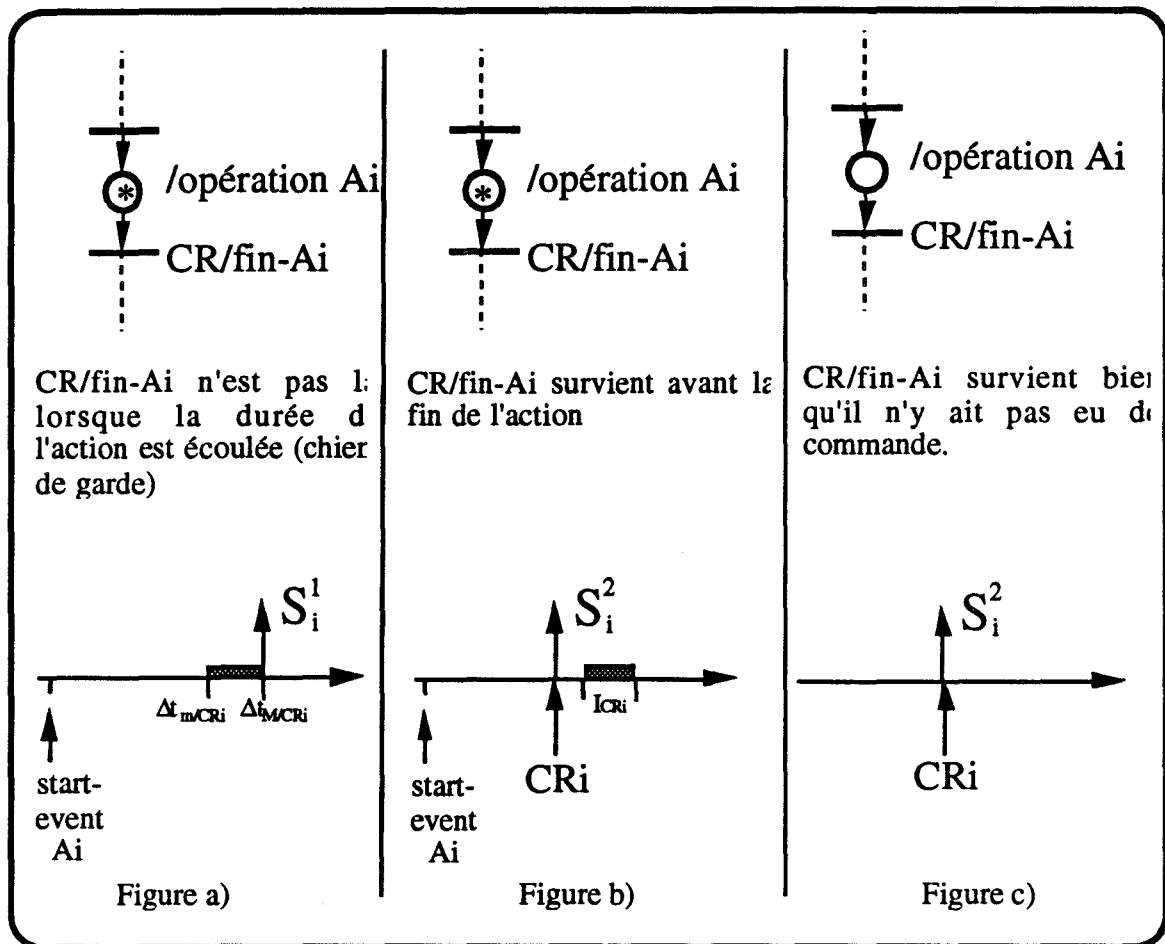


Figure I.43 : Définition des symptômes et contexte opérationnel associé.

*** Les symptômes de type II notés S^2_{CRi} .**

Ils correspondent à des comptes-rendus intempestifs. Nous distinguons les deux cas de figure modélisés par les figures I.43/b et I.43/c.

Dans la figure I.43/b, une action a été demandée, mais son compte-rendu survient avant l'intervalle de validation. Ce type d'anomalie peut être dû à la propagation d'une défaillance antérieure ou à une défaillance de capteur. Considérons l'exemple d'une jonction sur un convoyeur ou un système de chariots filoguidés (Fig. I.44).

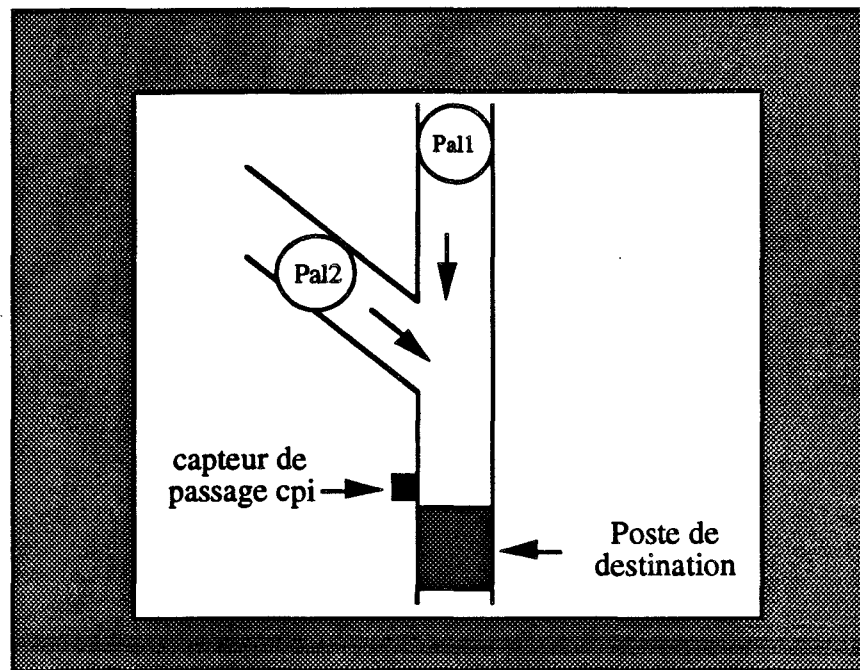


Figure I.44 : Schéma d'une jonction.

L'opération considérée consiste en un transfert de l'objet pal1 sur le poste destination. A l'entrée de ce poste est placé un capteur chargé de détecter les palettes entrant dans le poste. Supposons que suite à une défaillance du système de transport, l'objet pal2 devance pal1 sur le poste destination. Le passage de pal2 devant cpi va produire un compte-rendu intempestif. La figure I.43/c, illustre l'occurrence d'un compte-rendu en absence de toute commande pouvant la créer. Ce cas est identique au précédent. Pour s'en rendre compte, il suffit de reconsidérer le problème de la jonction. Mais cette fois-ci, on suppose qu'il n'y a pas d'objet pal1 à transférer sur le poste destination.

L'exemple de la jonction, utilisé ici est tout à fait significatif du problème lié aux comptes-rendus intempestifs. En pratique, ils ne sont liés à aucun contexte opérationnel cohérent. Le marquage du graphe de commande ne permet pas de les interpréter. Seule la définition d'un contexte "opératio-temporel" permettra d'interpréter ce type d'anomalie et d'en tenir compte.

IV.4.3 Exemple : Détection des symptômes liés à la défaillance d'un axe d'avance sur une M.O.C.N.

Ce type d'axe est généralement composé de patins liés à un banc et coulissants dans des glissières. Le mouvement d'avance est transmis par une vis sans fin à bille.

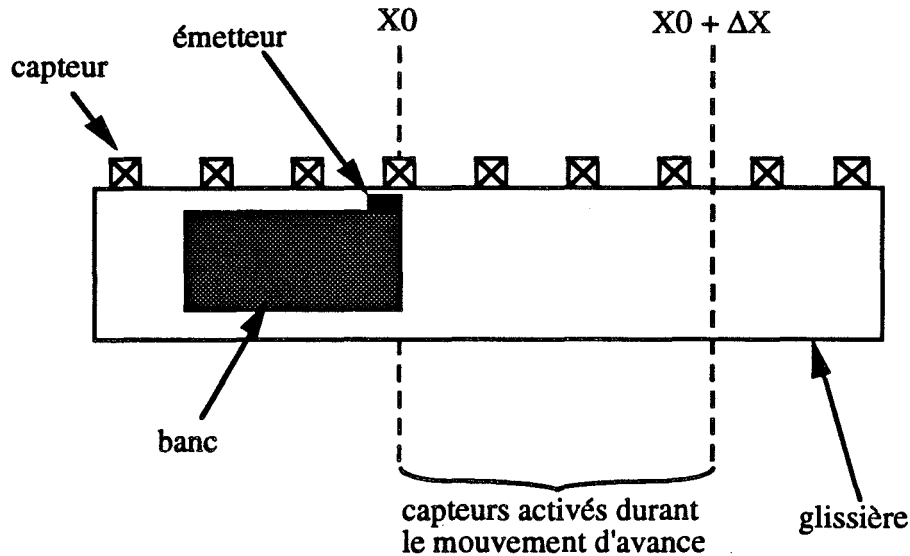


Figure I.45 : Principe de la surveillance d'axe.

L'idée pour surveiller le système, est de lier au banc un émetteur photoélectrique, et de disposer un certain nombre de capteurs, sur l'une des glissières (Fig. I.45). Supposons qu'à une date donnée la commande consiste à effectuer une avance de ΔX , à partir de l'abscisse X_0 . On connaît alors tous les capteurs qui seront activés par l'avance du banc. Si il n'y a pas de mouvement d'avance (cas de la défaillance du moteur d'axe), on va détecter autant de symptômes de type I qu'il aurait dû y avoir de capteurs activés. De même, s'il y a un retard à l'avance dû à l'usure de la vis sans fin, on pourra détecter sur un, voir plusieurs capteurs, un symptôme de type I, puis un symptôme de type II. Avec le même dispositif on détectera également une panne du système de régulation. Il suffit pour cela que l'on détecte des symptômes de type II liés à des capteurs situés au delà de l'abscisse $X_0 + \Delta X$.

Nous avons appliqué ici notre méthode de détection à un sous-ensemble local à un moyen de production. Les filtres sont donc activés ici par des ordres issus du niveau local de commande. Au chapitre III, § I.4.5.4, nous appliquerons le même principe, à la détection des symptômes des défaillances d'un convoyeur piloté par la commande de coordination. Ces deux exemples, montrent que la méthode de détection proposée ici est applicable quelque soit le niveau de commande.

IV.5 Outils temporels pour la spécification du mécanisme de filtrage-perception.

IV.5.1 Les RdP temporisés.

On distingue deux approches classiques, pour la temporisation des réseaux de Petri. L'une consiste à temporiser les transitions du RdP et l'autre les places. Les deux approches sont duales.

Pour temporiser un RdP au niveau des transitions, il suffit d'associer une **durée de tir** à chaque transition. Cela signifie que le franchissement des transitions associées à des durées non nulles, n'est pas instantané. Pendant la durée du tir le jeton n'est plus disponible même pour les transitions en conflit avec celle tirée (Fig. I.46).

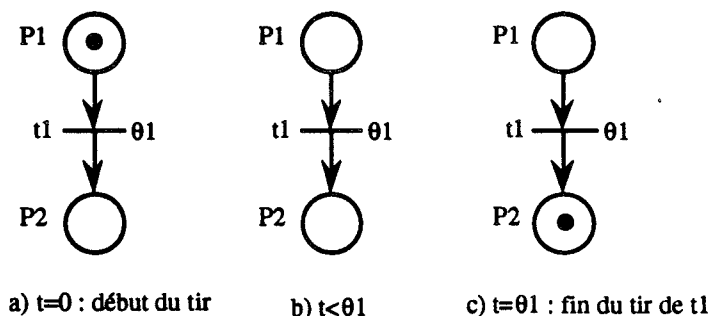


Figure I.46 : Mécanisme de tir dans un RdP temporisé.

Il existe une autre possibilité de temporisation des réseaux de Petri. L'approche consiste à associer des durées de tir aux arcs. Cette approche a été utilisée pour réaliser un simulateur de prégraphe dans CASPAIM-1 [SEY89].

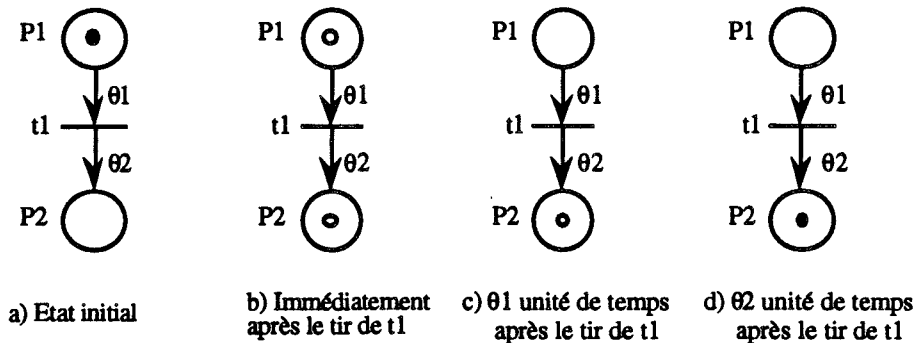


Figure I.47 : Modélisation d'un transfert par RdP temporisé au niveau des arcs d'après [SEY89].

Pour justifier une telle approche, citons l'exemple d'un transfert entre deux postes P1 et P2. On distingue quatre cas (Fig. I.47) :

* L'instant initial : le jeton est dans la place modélisant le poste de départ.

* Immédiatement après le tir : on est dans une configuration transitoire où l'objet transféré est quelque part entre les deux postes. Le jeton n'est disponible dans aucune des deux places. Mais pour indiquer qu'elles sont virtuellement occupées, des jetons "flous" sont mis dans chaque place.

* θ_1 unité de temps après le tir de la transition t1 : on est à présent sûr que l'objet à quitter le poste P1. Alors le jeton flou de P1 est retiré.

* θ_2 unité de temps après le tir de la transition t1 : on est sûr que l'objet est arrivé en P2. D'où la transformation du jeton flou, en l'habituel jeton opaque.

Il est à noter que θ_1 peut être supérieur à θ_2 . Par exemple pour modéliser le temps de reconfiguration d'une machine après l'usinage d'une pièce d'un type donné. Le fait de laisser un jeton "flou" pendant un temps T1 supérieur au temps de transfert de la pièce usinée signifie que la machine n'est pas prête à accueillir une autre pièce.

Nous n'avons pas retenu l'utilisation des RdP temporisés parce que la notion de temps est associée au tir de la transition. En fait, nous recherchons une notion associée à la sensibilisation de la transition comme dans la mise en œuvre du mécanisme de chien de garde.

IV.5.2 Les RdP temporels [CAR90].

Ils sont obtenus en associant à chaque transition une **durée de sensibilisation**. Le tir est instantané, mais la transition doit être sensibilisée pendant une durée donnée. Pendant toute la durée de cette sensibilisation le jeton reste disponible. Cela signifie que si plusieurs transitions sont en conflit, la transition franchie sera celle ayant été sensibilisée pendant un temps égal à sa durée de sensibilisation. En fait, il n'y a plus de conflit (cf. figure I.48).

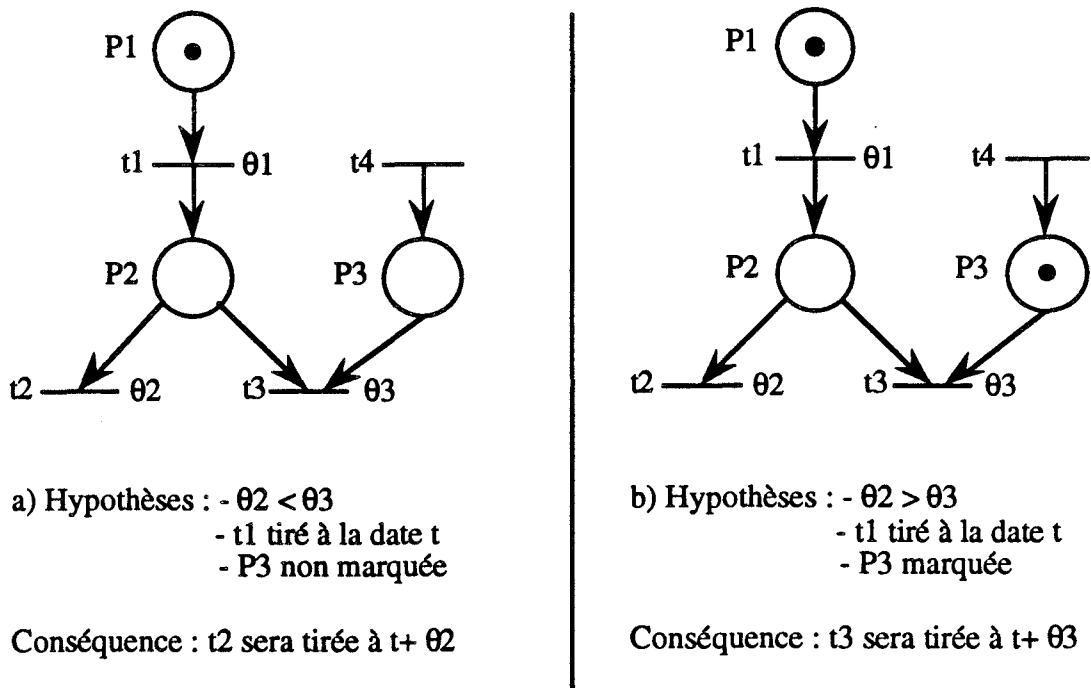


Figure I.48 : Résolution d'un indéterminisme par réseau de Petri temporel d'après [CAR90].

Le mécanisme de chien de garde se modélise facilement à l'aide des réseaux de Petri temporels. Dans le graphe de la figure I.48, il suffit de poser $\theta_3 = 0$. La valeur du chien de garde est alors égale à θ_2 .

Nous allons nous intéresser maintenant à un type de réseau de Petri temporel particulier : celui proposé par Merlin. Plutôt que d'associer des durées de sensibilisation, Merlin associe un intervalle $[a, b]$ à chaque transition. Cela signifie que la sensibilisation de la transition doit être supérieure ou égale à la valeur a et inférieure ou égale à la valeur b.

Silva et R. Valette ont montré que les RdP temporels de Merlin, permettent d'interpréter tout type de RdP. Les RdP ordinaires correspondent au cas où les intervalles de validations sont

égaux à $[0, \infty[$. On retrouve les réseaux de Petri temporels à durée en associant à chaque intervalle des transitions du type $[\theta, \theta]$. D'autre part on passe facilement des RdP temporisés au RdP temporels. A chaque transition t du RdP temporisé, associée à une durée de tir θ , il suffit de faire correspondre une séquence de transition $(t1, t2)$ du RdP temporel, avec :

- $\theta_{\min}(t1) = \theta_{\max}(t1) = 0$ et,
- $\theta_{\min}(t2) = \theta_{\max}(t2) = \theta$.

IV.5.3 Mécanisme de filtrage par RDPI temporels [TOG91].

Les RdPI ou Réseaux de Petri Interprétés sont définis à partir des ensembles suivants [ZAN90] :

- un ensemble de variables,
- un ensemble d'opérateurs agissant sur ces variables et activés dans les places du réseau,
- un ensemble d'événements et de conditions enrichissant les règles de franchissement des transitions.

Mis à part l'hypothèse de synchronicité, les RdPI saufs ont une sémantique analogue de celle du grafcet. Aussi, nous utiliserons la réceptivité pour désigner les conditions logiques liées au franchissement des transitions.

Nous introduisons ici les Réseaux de Petri temporels interprétés. Ce sont des RdPI dont les transitions sont temporisées par des intervalles de temps tels qu'ils sont définis par Merlin. La sémantique de ces Réseaux est la suivante : une transition est franchissable si sa réceptivité est vraie à une date comprise dans son intervalle de sensibilisation. La sensibilisation d'une transition dépend uniquement du marquage de ses places amonts.

Nous allons utiliser les RdPI temporels pour spécifier le mécanisme de filtrage. L'idée du filtrage est de positionner la fenêtre temporelle de chaque compte-rendu CRI lorsque l'on a reçu son Start-event. L'attente d'un compte-rendu avec filtre positionné est modélisé par la place P2 de la figure I.49.

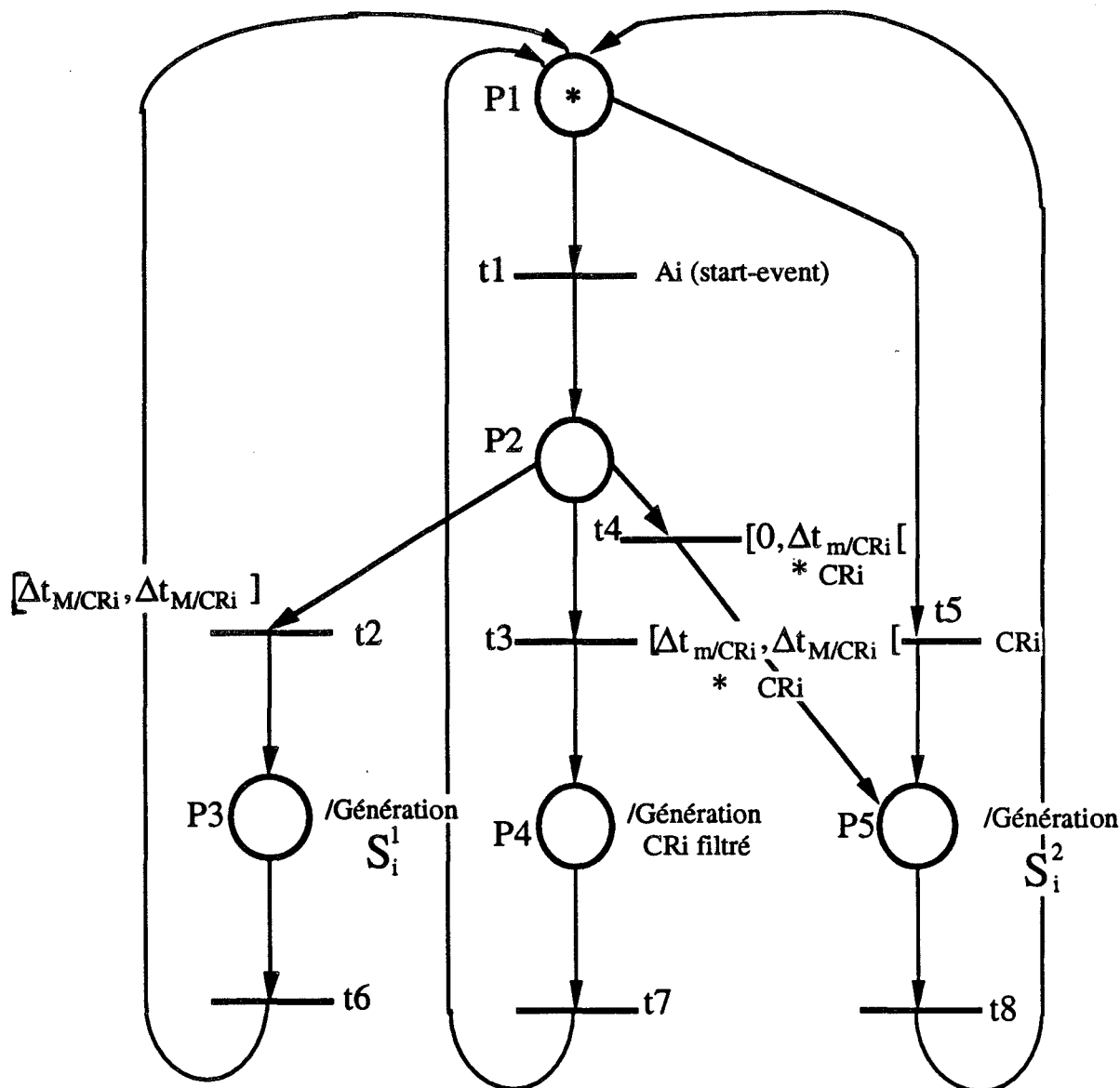


Figure I.49 : Mécanisme de filtrage.

Lorsque le filtre est positionné trois cas peuvent survenir :

* Le compte-rendu CRI arrive entre 0 (la date de positionnement du filtre) et $\Delta T_{m/CRI} - 1$.

CRI se situe avant la fenêtre temporelle : c'est donc un symptôme intempestif. Cela entraîne le tir de la transition t4. On génère alors un symptôme de type II (place P5).

* CRI arrive entre $\Delta T_{m/CRI}$ et $\Delta T_{M/CRI}$.

CRI est donc dans la fenêtre temporelle : cela correspond au comportement normal. On émet l'image de CRI à destination de la Partie Commande (tir de t3).

* A $\Delta T_{m/CRI} + 1$, on n'a toujours pas reçu CRI_i .

On déclenche le chien de garde modélisé par la transition t_2 . Un symptôme de type I (place P_3) est alors généré.

Enfin, lorsque le filtre n'est pas positionné, CRI_i peut apparaître. Conformément à nos spécifications du § IV.4.2, on génère un symptôme du type II par le tir de la transition t_4 .

Ce mécanisme de filtrage est simple et générique. On peut facilement le mettre en œuvre en grafcet, par exemple avec la version V4 du système des API de Télémécanique. On peut également mettre en œuvre ce mécanisme à partir de langages synchrones. L'avantage dans ce cas est qu'on obtient un code réellement réactif, facilement "implantable" sur calculateur.

L'utilisation du mécanisme proposé par la figure I.49, pose toutefois un problème. On est obligé d'associer un graphe de ce type à chaque couple opération-capteur. De ce fait, le mécanisme de filtrage, paramétré par $\Delta T_{m/CRI}$ et $\Delta T_{M/CRI}$, s'identifie au filtre lui même. Ce problème est dû aux bornes fixes des intervalles de sensibilisation. Pour agréger les mécanismes de filtrage de chaque compte-rendu, il faut disposer d'un langage permettant d'utiliser des intervalles de sensibilisation paramétrables en fonction du compte-rendu filtré. Avec les Réseaux de Petri temporels à Objets, nous disposons d'un tel outil.

IV.5.4 Les RdP temporels Objets.

Nous introduisons directement dans ce paragraphe la définition des RdPO temporels. On pourra trouver celle relative aux RdPO dans l'annexe B.

IV.5.4.1 Définition formelle des RdP temporels Objets [CAR90].

Cet outil a été spécifié par J. Cardoso [CAR90]. C'est en fait un RdPO ayant le comportement temporel des RdP temporels de Merlin. L'idée dans ce cadre d'hypothèses, est d'associer aux transitions, des intervalles de sensibilisation dont les bornes sont calculées pour chaque substitution d'objets aux variables des arcs prédécesseurs. Le calcul peut utiliser les valeurs d'attributs des objets.

Définition I.16 :

Un Réseau de Petri Temporel à Objet est un doublet $\langle R, I \rangle$ où :

* R est réseau de Petri à Objets et

* I est une fonction **intervalle de temps** définie par⁴ :

$$I : T \otimes S \longrightarrow (Q^+ \cup \{0\}) \otimes (Q^+ \cup \{0\} \cup \{\infty\}), \quad S : V \longrightarrow O$$

$$(t_j, s) \longrightarrow I(t_j, s) = [a_j, b_j] \text{ avec } 0 \leq a_j \leq b_j, \forall j, 1 \leq j \leq n, \\ n = \text{card}(T)$$

La fonction I associe à chaque couple **transition-substitution** du réseau un intervalle de temps dans lequel la transition peut être franchie pour la substitution.

Définition I.17 : Etats.

Les états d'un réseau de Petri Temporel à Objets sont définis par un couple (m, I) , où :

* m est la fonction marquage du réseau,

* I est la fonction intervalle de temps.

Définition I.18 : Transition franchissable.

Soit m un marquage, I la fonction intervalle de temps et t_j une transition d'un Réseau de Petri à Objets. t_j est franchissable à la date τ si et seulement si :

* t_j est franchissable depuis m ,

* τ est compris, borne incluse, entre a_j et la plus petite des dates de tir au plus tard des autres transitions sensibilisées, l'origine de temps pour τ étant la date à laquelle t_j est devenue sensibilisée.

Remarque :

Dans la définition précédente les autres transitions auxquelles il est fait allusion sont celles en "conflit" avec t_j (cf. Fig. I.48). Mais cette définition permet également de régler les conflits dus au flux de pièces dans une gamme opératoire. Si deux pièces sont en conflits pour une opération, le pilotage peut calculer les intervalles de sensibilisation respectifs des deux couples $(t_3, \langle p_1 \rangle)$ et $(t_3, \langle p_2 \rangle)$ en utilisant l'attribut date due de chaque pièce. Ces intervalles peuvent être différents permettant à l'une d'avoir la priorité sur l'autre (Fig. I.50).

⁴ S est la fonction substitution associant un objet à chaque variable. Pour sa définition, se reporter à l'annexe B.

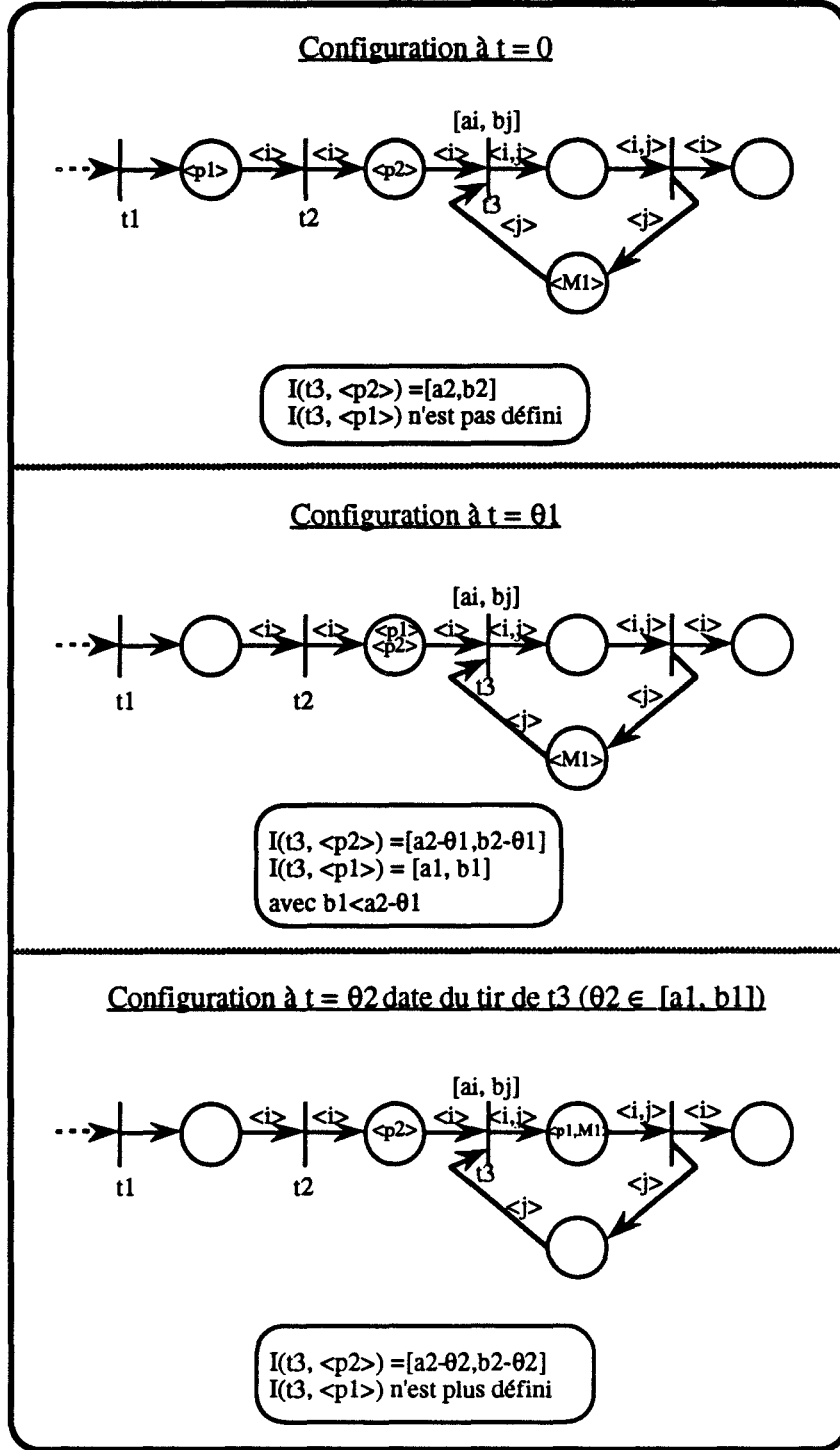


Figure I.50 : Résolution d'indéterminismes transition-substitution dus au flux de gamme.

Définition I.19 : Franchissement d'une transition.

Le tir d'une transition franchissable t_j à la date τ conduit à un nouvel état (m', I') avec :

$$* m'(p) = m(p) + \text{Pre}(p, t_j) - \text{Post}(p, t_j) \quad \forall p \in P$$

* $I'(t_j, s)$ est vide pour les transitions non sensibilisées par m' ,
 $I'(t_j, s) = [\max(0, a_j - \tau), b_j - \tau]$ pour les transitions qui étaient sensibilisées par m et qui sont restées sensibilisées.

$I'(t_j, s) = IO(t_j, s)$ pour toutes les transitions qui n'étaient pas sensibilisées par m et qui le sont maintenant.

Remarque :

$IO(t_j, s)$ est l'intervalle de sensibilisation initial de t_j pour la substitution s .

IV.5.4.2 Application au filtrage.

Le mécanisme est représenté par la figure I.51. Pour le mettre en œuvre on crée trois classes d'objets :

* **La classe FILTRE** dont les attributs sont :

- **identif** : identifiant d'un filtre donné,
- **opération** : spécifie le nom de l'action considéré comme Start-event pour le positionnement du filtre,
- **CR** : spécifie le nom du compte-rendu attendu,
- ΔT_m et ΔT_M sont les bornes de la fenêtre temporelle du filtre.

* **La classe OPERATION** dont les attributs sont :

- **identif** : identifiant d'une opération donnée.

* **La classe CR** dont les attributs sont :

- **identif** : identifiant d'un compte-rendu donné.

A partir des classes précédentes, nous définissons les variables :

- **a** de classe OPERATION,
- **c** de classe CR,
- **f** de classe filtre.

Les places du graphe de filtrage sont également typées :

- **PC** est de classe OPERATION ; elle sert d'interface avec la Partie Commande,
- **PRO** est de classe CR ; elle sert d'interface avec le Procédé,

- Toutes les autres places (de P1 à P6) appartiennent à la classe FILTRE ; on notera ici qu'aucun n-uplet n'est formé avec les objets de type OPERATION ou CR ; en fait ces objets interviennent uniquement dans le calcul des conditions de tir des transitions t1, t5, t6 et t7.

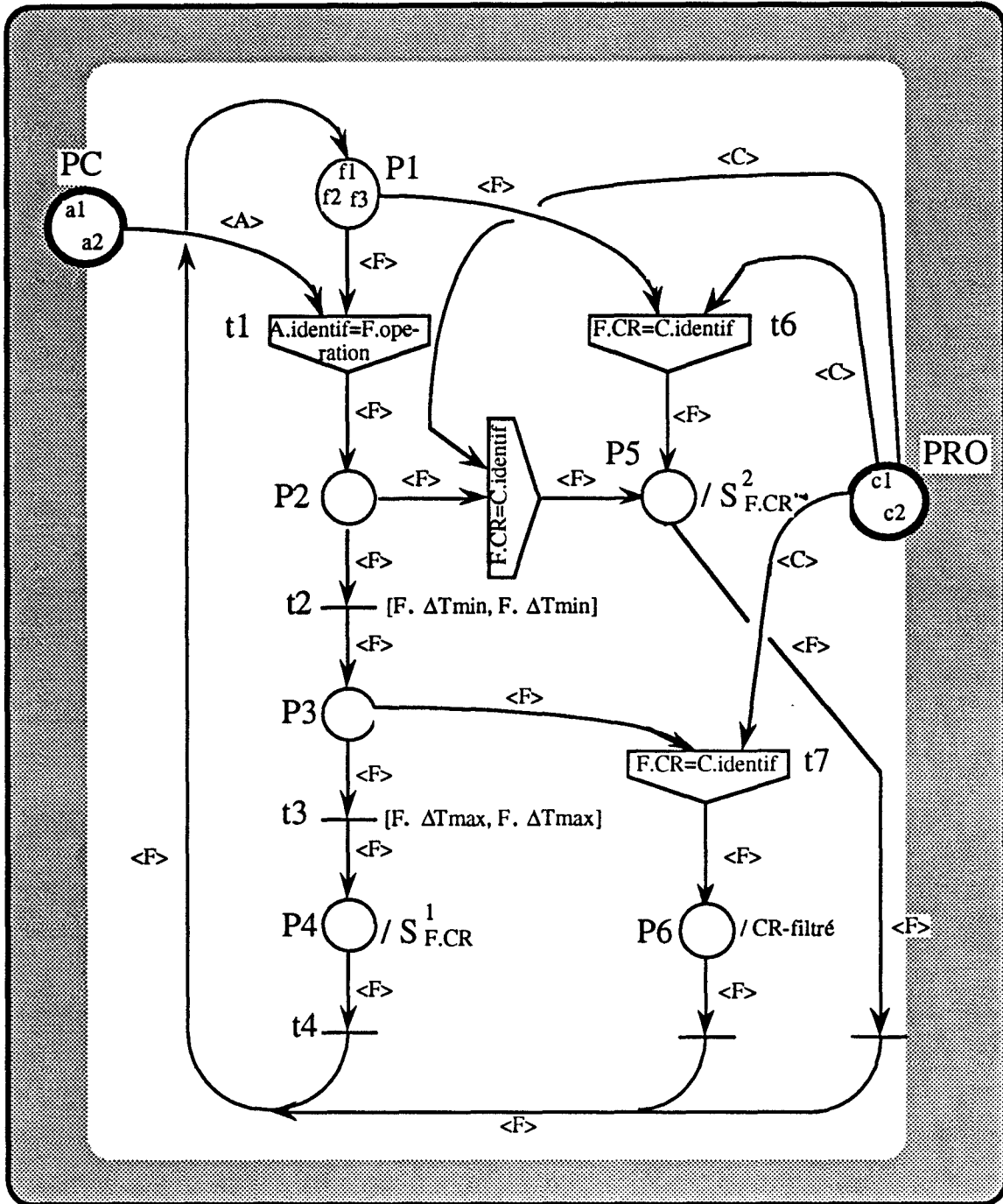


Figure I.51 : Mécanisme de filtrage par Réseau de Petri Temporel à Objets.

On notera que par rapport à la spécification avec les RdPI temporels, ici seules deux transitions sont temporisées. En outre, bien que nous utilisions le formalisme de Merlin, la temporisation est plutôt du type durée que du type intervalle.

Au niveau mise en œuvre, nous recommandons l'utilisation d'un langage comme l'ADA temps réel. Il dispose de primitives systèmes lui donnant son caractère temps réel. L'aspect objet quant à lui sera mis en œuvre grâce à la notion de "package".

IV.6 Classification des symptômes.

La classification des symptômes est basée sur la classification préalable des défaillances répertoriées. Le critère de classification utilisé ici est l'ampleur de la défaillance. Nous définissons deux classes :

*** Les défaillances significatives.**

C'est à dire que leurs conséquences remettent en cause la disponibilité d'une partie du système sans que la sécurité ne soit immédiatement altérée.

Par exemple la défaillance d'un aiguillage est un type de défaillance significative.

*** Les défaillances critiques.**

Elles altèrent de manière sensible et rapide la sécurité du système. Il faut réagir promptement pour mettre en position de repli (arrêt des mouvements ou arrêt d'urgence) les éléments matériels concernés.

Il est clair qu'une telle classification est facilitée par une analyse préliminaire du système du type APD ou AMDEC [VIL88].

La classification des symptômes suit celle des défaillances. Ainsi, **tout symptôme apparaissant dans la signature d'une défaillance critique est considéré comme critique.** En effet, même si en définitive le diagnostic l'attribue à une défaillance significative, on peut prendre le risque de laisser évoluer la partie concernée par la défaillance critique. Un symptôme ne sera que significatif s'il ne peut être attribué qu'à des défaillances significatives.

Bien sûr une telle classification est statique et n'intègre pas la connaissance supplémentaire relative aux symptômes déjà détectés. On pourrait être tenté d'utiliser un raisonnement probabiliste pour calculer des facteurs de vraisemblance permettant de quantifier

au fil du temps l'ampleur de chaque symptôme détecté. Une telle approche serait en fait un pré-diagnostic qu'il n'y a pas lieu d'établir dans la phase de détection. Cela pourrait remettre en question son caractère réactif.

Pour spécifier le mécanisme de classification, il faut enrichir les attributs de la classe **FILTRE**. On différencie l'ampleur dans un filtre par rapport au type du symptôme créé. On ajoute donc à la classe **FILTRE** les attributs :

- **ampleur(1)** : spécifie l'ampleur d'un symptôme de type I obtenu avec un filtre donné,
- **ampleur(2)** : spécifie l'ampleur d'un symptôme de type II obtenu avec un filtre donné.

Si **ampleur(i)** = critique, un message spécifiant, le **type** du symptôme (type I ou type II), l'**identifiant du CR** responsable de l'occurrence du symptôme, et la **date** de détection, est émis aux modules de diagnostic et de recouvrement. Si **ampleur(i)** = significatif, le message est seulement émis vers le module de diagnostic. Nous verrons au chapitre III qu'un tel message va permettre la création d'une instance datée de ce symptôme pour mettre en œuvre le diagnostic.

Notons ici qu'en cas de filtrage basé sur la spécification par RdPI temporels, la classification peut être mise en œuvre de manière séparée par un système à base de règles.

IV.7 Conclusion

Dans cette partie nous avons spécifié les différents mécanismes mis en œuvre dans le cadre de la détection. Notre but visait à dégager une démarche systématique, facilement applicable quel que soit le niveau de commande. La nature générique du modèle temporel et du mécanisme de filtrage-perception confère à notre démarche le caractère systématique et rigoureux attendu. Elle peut être résumée de la manière suivante :

- 1 - déterminer le modèle temporel associé à chaque couple opération/ compte-rendu,
- 2 - réaliser une classification des défaillances et des symptômes associés,
- 3 - choisir une des spécifications, du mécanisme de filtrage, proposées en fonction des calculateurs sur lesquels se feront l'implantation et de la complexité du problème.

CONCLUSION

Dans cette partie, nous nous sommes intéressés à présenter notre conception de la surveillance en ligne dans le contexte des ateliers flexibles. Cela nous a conduit à définir la structuration de la surveillance en trois fonctions de base : la détection, le diagnostic et le recouvrement. Nous avons montré comment en nous inspirant du concept de filtre introduit par le CRAN/LACN, nous positionnons le module de détection, en filtre des comptes-rendus du procédé. Cette notion de filtre est essentielle puisque le mécanisme interne de la détection est basé dessus.

Notre conception de la surveillance apparaît clairement ici, avec l'insertion d'un mécanisme de classification des symptômes, dès la phase de détection. Elle est basée sur la nécessité de prendre en compte en temps réel, toute défaillance, pour deux raisons principales :

*** En premier lieu pour une raison de sécurité.**

Il faut réagir tout de suite pour prendre les décisions adéquates engendrant un recouvrement immédiat en fonction des risques encourus.

*** En deuxième lieu pour une raison de recherche d'une disponibilité maximale des moyens de production.**

Il s'agit dans ce cas de ne pas prendre de décision hâtive, entraînant l'indisponibilité d'une partie du système.

Cette analyse rapide montre qu'en fait, la classification introduite dans la fonction détection, est une réponse à la même problématique que celle résolue par le test d'hypothèses dans le contexte de la surveillance des processus continus.

Mais notre approche n'est pas exclusivement caractérisée par la recherche d'un compromis entre sécurité et disponibilité, en raison de la contrainte de temps-réel. Au contraire, nous pensons que le temps est une source d'information supplémentaire, utilisable dans le cadre du diagnostic. En effet, l'observation du comportement dynamique d'un système, permet un diagnostic plus aisé que lorsqu'il est à l'arrêt. Nous exploiterons cette idée au chapitre III, en introduisant la notion de **Signature Temporelle Causale**.

CHAPITRE II

LE DIAGNOSTIC : EXPRESSION DES BESOINS

SOMMAIRE

I - Les formes classiques du diagnostic

II - Contraintes

III - Modélisation dans le cadre du diagnostic

IV - Modélisation fonctionnelle : analyse de l'existant

INTRODUCTION

Le diagnostic est la fonction centrale du système de surveillance tel que nous le concevons. Rappelons que notre démarche a pour but l'automatisation du diagnostic technique. Notre approche du diagnostic se situe dans le cadre de la maintenance curative d'un atelier flexible. Aussi dans nos raisonnements, nous supposerons toujours qu'il y a une panne effective (défaillance catalectique) dans le système. Dans ce contexte, le diagnostic a un double but : d'une part localiser et identifier les causes premières de défaillances et d'autre part permettre d'anticiper la sollicitation d'entités défaillantes par la commande. Nous faisons donc le choix de diviser la fonction diagnostic en deux sous-fonctions : l'identification et l'analyse des conséquences. Ce choix s'explique essentiellement par des considérations d'efficacité. Nous le justifierons dans le chapitre III.

La conception du système de diagnostic est basée sur une modélisation du système à surveiller. Cette approche a été nécessitée par les insuffisances des formes classiques du diagnostic vis à vis de contraintes issues de notre approche de la surveillance d'un atelier flexible.

Ce chapitre est consacré à l'expression des besoins pour le diagnostic en ligne. Il comprend essentiellement quatre parties.

La première partie est un rappel sur les formes classiques du diagnostic. A partir de ces rappels et du cahier de charges de la surveillance en ligne (cf. chapitre I), nous analysons l'ensemble des contraintes rencontrées pour la conception du diagnostic en ligne. C'est l'objet de la partie sur les contraintes. La partie suivante porte sur la définition de différents types de modèle pour le diagnostic. Nous y introduisons notamment les notions de modèle qualitatif et modèle de connaissance. Enfin dans la quatrième partie, nous présentons l'analyse de l'existant en matière de modélisation fonctionnelle.

I. LES FORMES CLASSIQUES DU DIAGNOSTIC

I.1 Introduction

Quelqu'en soit la forme, le diagnostic classique est caractérisé par une démarche en 'cascade' [YIM86]. La démarche est généralement initiée par la réception d'une alarme.

Ensuite, l'acquisition de symptômes permet de focaliser la recherche de la panne sur une partie du système. C'est ce que nous appellerons la **localisation** du sous-système défaillant. Cette localisation est grossière. Elle ne permet donc pas d'envisager la réparation du système dans la mesure où tous les constituants du sous-ensemble localisé, restent potentiellement incriminables. Il faut passer à une seconde étape du diagnostic consistant à identifier précisément l'origine de la défaillance : c'est l'étape d'**identification**. Cette identification nécessite l'acquisition de symptômes supplémentaires. Une fois la défaillance première identifiée on peut envisager la réparation (Fig. II.1).

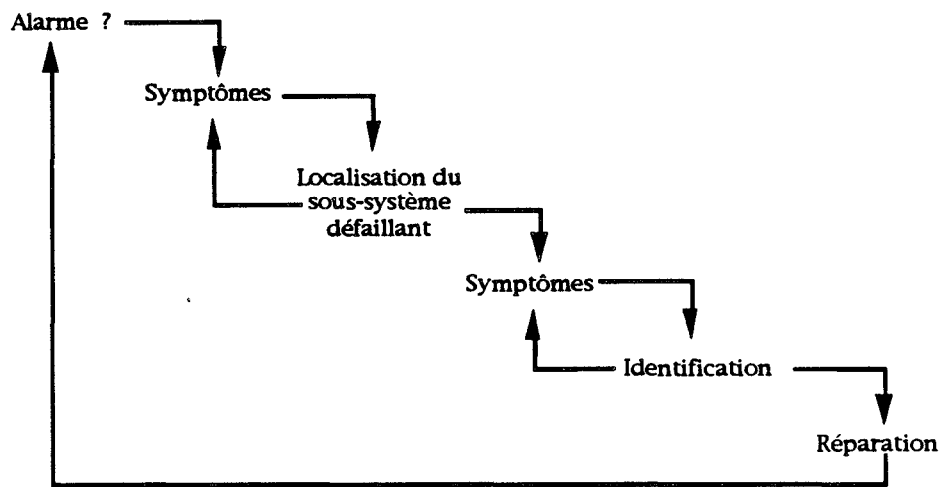


Figure II.1 : Schéma de principe du mécanisme de diagnostic

Dans certains cas, la démarche présentée aboutit à la localisation de plusieurs sous-ensembles et à l'identification de plusieurs pannes. On dit alors que l'on est dans un cas de défaillances multiples. Les défaillances multiples sont difficiles à diagnostiquer lorsque leurs symptômes ne sont pas indépendants. Cela arrive parfois dans les systèmes de production manufacturiers. Dans le développement de nos algorithmes, nous nous placerons toujours dans l'hypothèse de l'occurrence de défaillances multiples.

Au niveau diagnostic, nous pouvons distinguer deux formes classiques : le diagnostic procédural et le diagnostic à base de système expert.

I.2 Le diagnostic procédural

Cette forme du diagnostic est également connue sous les termes de test automatique ou ATE (Automatic Test Equipment). Elle est surtout rencontrée dans le diagnostic des systèmes

embarqués (aéronautique, chemin de fer etc...) et des systèmes informatiques. La série d'auto-tests effectués par tout système informatique lors du démarrage, constitue l'exemple type illustrant les avantages et les inconvénients de ce type de diagnostic. Son principal avantage c'est sa nature systématique. En effet, quelque soit la nature du déclenchement (type d'alarme, déclenchement automatique ou initié par un opérateur), la procédure reste immuable : tous les tests s'enchaînent dans le même ordre. Cet examen systématique de toutes les défaillances potentielles rend l'identification déterministe. Mais cette caractéristique est obtenue au prix d'un certain nombre d'inconvénients parmi lesquels on peut citer :

- La difficulté de concevoir une procédure prenant en compte toutes les pannes potentielles du système (problème de la complétude) ;
- La difficulté de diagnostiquer des pannes intermittentes ;
- La difficulté d' adapter le diagnostic à des modifications mineures du système surveillé (manque de flexibilité) ; il faut souvent repenser l'ensemble de la procédure ;
- La durée de l'ensemble de la procédure qui croît rapidement avec la complexité des systèmes surtout lorsque la maintenance impose des identifications très fines ;
- Le manque de convivialité pour les utilisateurs ;
- L'absence de réactivité ; en effet, de telles procédures sont généralement déclenchées par les opérateurs après constatation de l'occurrence d'une défaillance.

Ces inconvénients expliquent le manque de confiance des équipes de maintenance face à de tels systèmes, d'autant plus que les ATE sont incapables de justifier leur raisonnement.

De nombreux travaux menés ces dernières années ont permis d'atténuer quelque peu certains de ces inconvénients, notamment d'améliorer la vitesse d'identification de la panne. L'idée de base est que les tests portent une signature permettant de soupçonner ou de disculper certains composants ; il n'y a donc plus intérêt à figer l'ordre des tests. Des algorithmes sont utilisés pour calculer le meilleur test à réaliser à partir des résultats du précédent : c'est la technique de l'entropie minimale [KLE90], [AYG86].

1.3 Le diagnostic à partir des systèmes experts.

Ce type de diagnostic a connu un essor important ces dernières années, surtout dans le domaine médical (MYCIN, CASNET, ...). Schématiquement, un système expert comporte deux parties :

- la base de connaissance,
- le moteur d'inférence.

La base de connaissance sert à coder le savoir sur un domaine ou un système donné. Elle se subdivise en une connaissance statique et une connaissance dynamique appelée base de faits. Ainsi, dans le contexte du diagnostic, la base de faits rassemble l'ensemble des symptômes détectés à un instant donné. Au cours de l'inférence elle peut être enrichie par des déductions intermédiaires.

Les connaissances statiques représentent généralement le savoir sur un domaine (ou un système). Le formalisme le plus utilisé est celui des règles de production dont la syntaxe générale est :

<prémisses> ---> <conséquents>.

Les prémisses prennent leurs valeurs sur la base de faits. Les conséquents permettent de coder les déductions intermédiaires ou le diagnostic final. Considérons le procédé représenté par la figure II.2. Les CNi représentent des capteurs de niveau et les Vi des vannes.

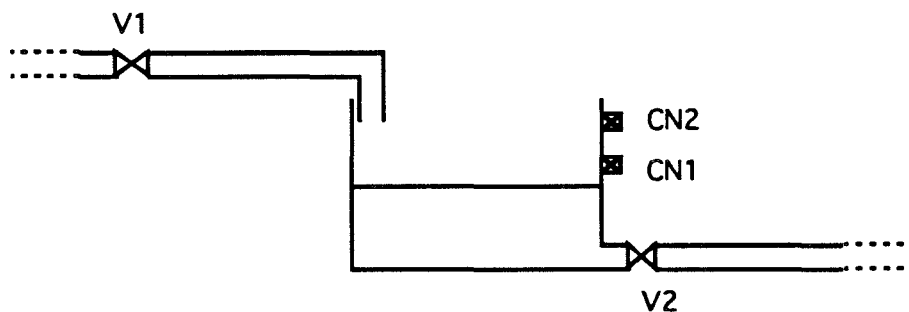


Figure II.2.

Nous supposons disposer d'un système de régulation permettant de régler le débit de la vanne V1 en fonction du niveau de fluide dans la cuve. Notamment si le niveau atteint CN1 (i.e. CN1=1), V1 doit être fermée. Si nous admettons que le système de régulation est correct (il peut avoir son propre système de surveillance), nous pouvons écrire les règles de diagnostic suivantes :

Si (CN1=1) alors (niveau-cuve > niveau-normal)

Si (niveau-cuve > niveau-normal) et (CN2=1) alors (V1 est défaillante)

Dans cet exemple (CN1=1) est un fait initial, (niveau-cuve > niveau-normal) une déduction intermédiaire, et (V1 est défaillante) une conclusion finale.

Il existe d'autres formes de représentation de la connaissance. Nous les évoquerons ultérieurement.

Le moteur d'inférence est une procédure utilisée pour interpréter la base de connaissance. Cette interprétation peut se faire selon trois modes :

- chaînage avant,
- chaînage arrière,
- chaînage mixte (combinaison des chaînages avant et arrières).

Le choix de l'un ou l'autre de ces chaînages dépend souvent du mode d'initialisation du diagnostic et du besoin de connaître les conséquences d'une panne autres que les manifestations qui l'ont révélée. Nous distinguons quatre cas :

* Cas 1 : L'opérateur soupçonne un composant d'être défaillant.

Le système sera déclenché en chaînage arrière à partir de l'hypothèse de défaillance du composant.

* Cas 2 : L'opérateur soupçonne un composant d'être défaillant. Si cette défaillance est confirmée, il veut connaître toutes les conséquences sur le système.

Le diagnostic sera déclenché chaînage arrière puis chaînage avant (chaînage mixte).

* Cas 3 : Le diagnostic est initié par la réception d'une alarme (diagnostic en cascade).

Dans un premier temps on effectue un chaînage avant pour la localisation du sous-système défaillant. Ensuite au niveau de ce sous-ensemble, on déclenche un chaînage arrière sur l'élément le plus susceptible d'être défaillant. Si l'évaluation échoue on en choisi un autre. Le choix peut être fait en utilisant la technique de l'entropie minimale afin de ne pas tomber dans le travers des ATE classiques.

* Cas 4 : Le diagnostic est initié par la réception d'une alarme et l'on désire connaître toutes les conséquences de la défaillance.

Dans ce cas on combine chaînage avant, puis chaînage arrière, et enfin de nouveau un chaînage avant.

But Initiateur	Identification	Identification et Analyse des conséquences
Opérateur	Chaînage arrière	Chaînage arrière puis Chaînage avant
Alarme	Chaînage avant puis Chaînage arrière	Chaînage avant1 puis Chaînage arrière et enfin Chaînage avant2

Figure II.3 : Tableau récapitulatif des techniques d'inférence utilisées, en fonction du mode d'initialisation et du but du diagnostic.

Un système expert se caractérise également par une interface homme-machine **conviviale** et **ergonomique**. Ces deux caractéristiques facilitent son utilisation par des non informaticiens, aussi bien en phase de conception (dans le cadre du transfert de la connaissance d'un expert) qu'en phase opérationnelle (pour le dialogue avec l'opérateur). D'autre part, le transfert d'expertise peut se faire progressivement, même si le système est déjà opérationnel (notion d'enrichissement). Cette flexibilité résulte de la nature fragmentée de la connaissance. La conséquence est que de tels systèmes sont facilement adaptables à des modifications du système surveillé.

Parmi les inconvénients de ces systèmes, nous citerons plus particulièrement la lenteur due principalement à leur aspect interactif (nombreuses entrées-sorties). Par ailleurs leur pouvoir explicatif est limité. Il se résume à l'énoncé des règles déclenchées. Ce type d'explication est plus utile aux concepteurs afin d'en contrôler le comportement, qu'aux opérateurs ou techniciens de maintenance. Ces derniers, pour se fier à un système de diagnostic, ont besoin qu'il leur explique comment la défaillance diagnostiquée s'est propagée dans le système pour produire les manifestations qui ont déclenché l'alarme...

I.4 Conclusion

Dans cette première partie nous avons évoqué les deux formes classiques du diagnostic : les ATE et le diagnostic à base de systèmes experts. Elles présentent toutes deux

un certain nombre d'avantages et d'inconvénients. Si historiquement les ATE sont apparus avant les systèmes experts, les nouveaux ATE tendent à acquérir certaines spécificités de ces derniers. Ainsi, nous avons vu que l'on retrouve dans ces deux familles l'utilisation de la technique de l'entropie minimale. Pour nous résumer, nous dirons qu'aujourd'hui les ATE se caractérisent par une plus grande autonomie qui garantit une certaine rapidité dans l'identification de la première panne (contexte de défaillances multiples). Par contre les systèmes experts se caractérisent par leur flexibilité et leur convivialité.

Rapidité, flexibilité et convivialité sont à notre sens des caractéristiques essentielles pour un bon système de diagnostic. Il existe en Intelligence Artificielle des techniques permettant de les mettre en oeuvre. Nous pensons notamment aux mécanismes de raisonnement, aux méthodes d'acquisition, représentation et maintenance de la connaissance. Notamment, concernant les ateliers flexibles, certaines techniques permettraient de tenir compte de l'aspect distribué de la connaissance. Notre but est de les utiliser pour concevoir des systèmes de diagnostic performants, adaptés au contexte de l'industrie manufacturière ...

II. CONTRAINTES

Dans cette partie, nous allons énoncer un certain nombre de contraintes qui permettront de mieux situer notre approche du diagnostic. Ces contraintes sont rapportées d'une part aux systèmes experts classiques et d'autre part à l'hypothèse du bouclage automatique énoncée dans le chapitre I.

II.1 Contraintes liées à une approche système expert classique.

Notre conception du diagnostic est fondamentalement différente de celle couramment admise. Ici, il ne s'agit plus d'orienter le technicien de maintenance dans l'acquisition de symptômes. Dans le contexte des ateliers flexibles, le rôle du technicien sera bien souvent tenu par l'opérateur. Vis à vis du diagnostic, il aura un rôle passif. Comme présenté dans le premier chapitre, les symptômes sont détectés automatiquement par le module de détection et émis vers le système de diagnostic. Celui-ci en déduit un certain nombre de conclusions sur l'état du procédé et les émet vers l'opérateur et le module de planification des opérations correctives. Cette forme de diagnostic est appelée **diagnostic en ligne**.

II.1.1 Manque d'expertise.

Notre approche se situe en phase même de conception du système de production à surveiller. Nous ne disposons donc pas d'expertise sur les défaillances du système. Cette situation ne correspond pas au contexte habituel de conception des systèmes experts de diagnostic. En conséquence, la méthodologie d'acquisition de la connaissance doit être différente. De manière générale, nous pensons que dans le cadre des systèmes industriels neufs, il faut partir d'une analyse systématique du fonctionnement du système à surveiller.

II.1.2 Contrainte temporelle.

Pour introduire le problème lié à la contrainte temporelle, nous allons reprendre l'exemple du convoyeur. Sur ce convoyeur sont disposés deux palettes : l'une en P5 et l'autre en P4. Le but de la commande est de réaliser les transferts P5 --> P1 et P4 --> P3. P3 représente une zone de dépalettisation des pièces par un robot pour chargement sur une machine d'usinage. P5 représente une zone de chargement des pièces sur le convoyeur.

A partir de P5 on peut atteindre les postes P1 (transfert P5 --> P1) ou P3 (transfert P5 --> P3). Un aiguillage permet de choisir l'une ou l'autre de ces directions. Supposons que la commande lance en parallèle les ordres "déblocage" de la butée-1 et "positionnement-d" de l'aiguillage. La palette étant libérée, si le positionnement de l'aiguillage échoue, la direction prise ne sera pas la bonne. Il y a alors risque de perturbation du transfert P4 --> P3. La seule possibilité de recouvrement d'erreur réside dans le blocage de la palette par la butée-2 (Cette dernière étant normalement en position non bloquant). Le diagnostic de défaillance de l'aiguillage et de sa conséquence directe (mauvaise orientation de la palette) doit donc être réalisé avant l'arrivée de la palette au niveau de la butée-2.

L'aspect en ligne induit donc une contrainte temporelle.

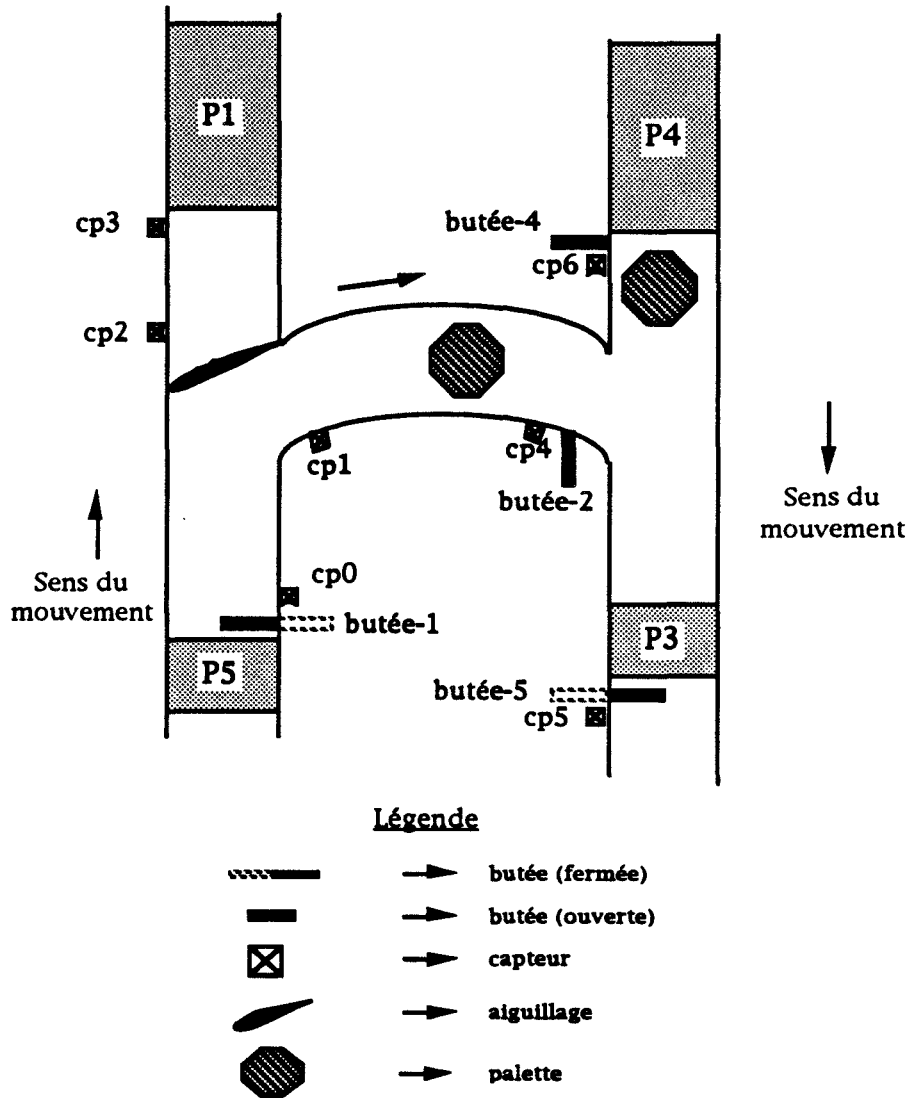


Figure II.4 : Convoyeur.

II.1.3 Données incomplètes.

Elles concernent aussi bien la base de règles que la base de faits.

II.1.3.1 Complétude de la base de règles

Le problème de la complétude est à relier à celui de la nature des connaissances codées. Dans la majeure partie des systèmes experts de première génération, ces connaissances sont empiriques. L'expérience accumulée permet à certains experts de mettre en relations des défaillances et leurs manifestations. Ces relations ne s'appuient sur aucune analyse du système. Elles sont uniquement basées sur des corrélations qualitatives.

II.1.3.2 Par rapport à la base de faits.

II.1.3.2.1 Causes

Deux causes permettent d'expliquer la nature incomplète de la base de faits. Ce sont l'**aspect en ligne** du diagnostic et le **niveau de perception** des symptômes. Pour les illustrer, nous reprenons l'exemple concernant la défaillance de l'aiguillage du convoyeur.

D'après les spécifications de la détection faites au chapitre II, les symptômes suivants sont détectés : $S^1_3, S^1_2, S^2_1, S^2_4$. Représentons l'ordre de détection de ces symptômes sur une échelle temporelle (Fig. II.5).

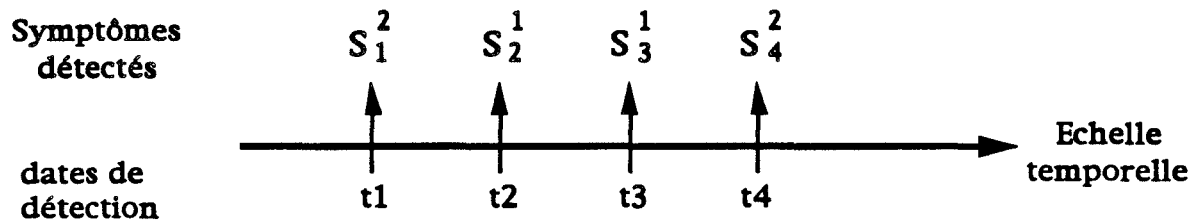


Figure II.5 : Séquence de détection de symptômes.

Si nous supposons que les capteurs ne peuvent être défaillants, les règles suivantes permettent de coder les défaillances potentielles des certains éléments structurels du convoyeur :

(def aiguillage)	-->	$\{S^1_2, S^1_3, S^2_1, S^2_4\}$
(def tapis)	-->	$\{S^1_2, S^1_3\}$

Cet exemple montre qu'avant la date $t = t_4$, nous ne pouvons pas identifier la cause des symptômes S^1_2 et S^1_3 .

Pour illustrer l'influence du niveau de perception sur le diagnostic, supposons à présent que nous disposons de capteurs de fin de course sur le vérin de l'aiguillage (Fig. II.6). Nous supposons par ailleurs qu'une défaillance du vérin est l'unique cause possible de défaillance de l'aiguillage. Dans le mode de fonctionnement normal, le capteur "ve-av" (respectivement à "ve-ar") a pour valeur 1 lorsque l'aiguillage est positionné à gauche (respectivement à droite).

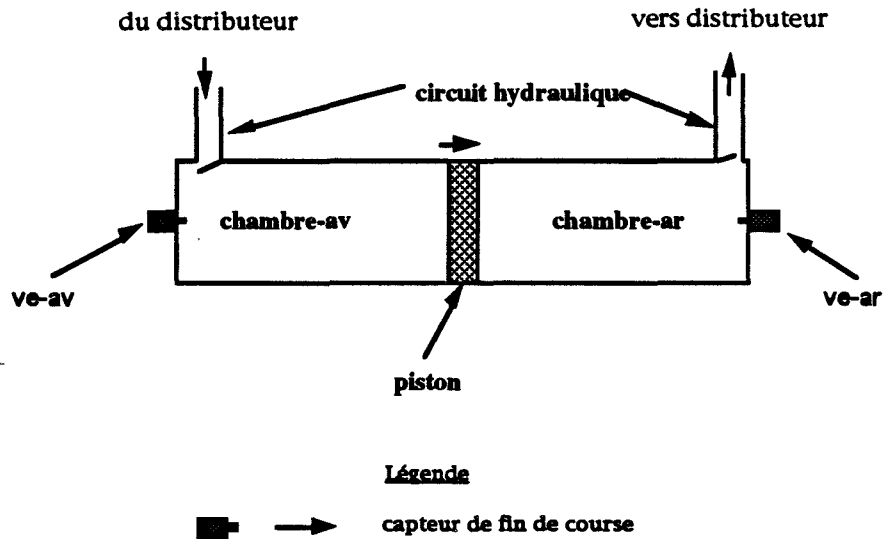


Figure II.6 : Schéma d'un vérin muni de capteurs.

Dans ce contexte la défaillance de l'aiguillage lors du positionnement à droite se traduirait par la détection du symptôme S^1_{ve-ar} . Comme l'opération de positionnement de l'aiguillage est plus rapide que l'arrivée d'une palette devant l'un des capteurs placés après lui, ce symptôme serait détecté à une date $t < t_1$.

D'un point de vue structurel, le vérin est un composant de l'aiguillage qui lui même est un composant du convoyeur. Ce résultat est donc intéressant. Il signifie que l'on identifiera plus vite une défaillance avec des symptômes perçus à un niveau inférieur à celui de l'aiguillage (i.e le niveau du vérin) qu'avec ceux d'un niveau supérieur (i.e le niveau du convoyeur). Ce résultat est d'ailleurs vrai quelque soit le domaine. Dans le domaine médical il correspond à la règle qui veut que les symptômes cliniques (fièvre, boutons, ...) soient toujours postérieurs à ceux biologiques (analyses médicales). Un exemple d'application de cette règle est le dépistage "systématique" du cancer du sein chez la femme.

L'analyse précédente nous permet de tirer deux conclusions importantes :

- Plus le niveau de perception est bas (au sens d'une décomposition hiérarchique), plus le diagnostic est rapide ;
- Plus le niveau de perception est bas, plus le diagnostic est précis En effet, dans l'exemple précédent, d'une part on diagnostique la défaillance de l'aiguillage, d'autre part on diagnostique celle d'un de ses composants : le vérin.

Il faut néanmoins souligner, que pour des raisons technologiques ou financières, il n'est pas toujours possible de positionner des capteurs aux niveaux de perception les plus intéressants.

II.1.3.2.2 Conséquences

Quelles sont les conséquences, de la disposition de données incomplètes par rapport aux mécanismes de raisonnement couramment utilisés dans les systèmes de diagnostic hors ligne ?

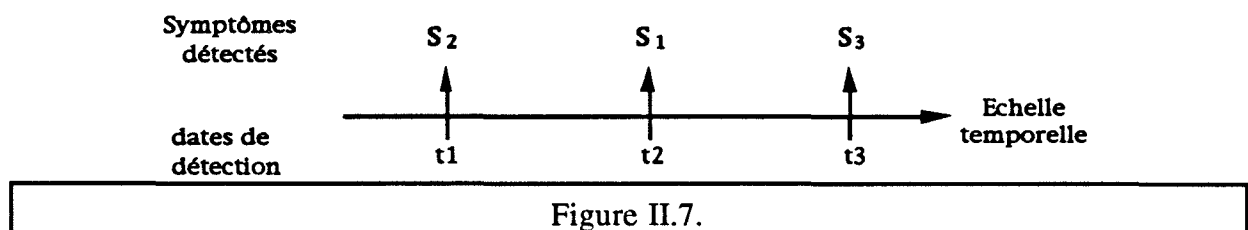
Les mécanismes de résolution classique reposent implicitement sur la disponibilité de tous les symptômes (exceptés ceux correspondant à des défaillances intermittentes). En effet, dans ce contexte, le système n'évolue plus dès la détection d'une défaillance. Les problèmes posés peuvent se résumer aux questions suivantes :

- Quels sont les critères pour déclencher le diagnostic à partir de la réception de symptômes,
- Quelle confiance accorder aux résultats si l'on admet la possibilité de défaillances multiples.

Afin d'illustrer cette problématique, prenons l'exemple donné par les règles de production suivantes :

$$\begin{array}{llll}
 \mathbf{r1 :} & \mathbf{C1} & \mathbf{-->} & \mathbf{S1 \wedge S2 \wedge S3.} \\
 \mathbf{r2 :} & \mathbf{C2} & \mathbf{-->} & \mathbf{S1 \wedge S2} \\
 \mathbf{r3 :} & \mathbf{C3} & \mathbf{-->} & \mathbf{S1 \wedge S3}
 \end{array}
 \tag{II.1}$$

La sémantique de ces relations est la suivante : en cas de défaillance ayant pour cause C_i , les symptômes associés(les S_j) seront détectés. Dans cette formulation les S_j ne sont pas typés. Par ailleurs l'opérateur " $\wedge$ " utilisé correspond au "et" habituellement utilisé dans les règles de production.



Supposons par ailleurs que les symptômes soient détectés selon le chronogramme suivant :

A priori, le moyen le plus équitable de déclenchement du diagnostic est celui consistant à le déclencher à chaque nouvelle occurrence de symptômes. C'est donc la procédure de déclenchement que nous appliquerons pour cet exemple.

Appliquons à la résolution un mécanisme éprouvé dans le diagnostic médical. C'est le mécanisme issu de la **Set Covering Theory** proposé par Reggia [REG83]. Dans cette approche, la base de règles relie les symptômes aux causes potentielles. Nous pouvons donc réécrire les règles (II.1) de la manière suivante :

$$\begin{array}{llll}
 \mathbf{R1 :} & \mathbf{S1} & \mathbf{-->} & \mathbf{C1 \vee C2 \vee C3} \\
 \mathbf{R2 :} & \mathbf{S2} & \mathbf{-->} & \mathbf{C1 \vee C2} \\
 \mathbf{R3 :} & \mathbf{S3} & \mathbf{-->} & \mathbf{C1 \vee C3}
 \end{array}
 \tag{II.2}$$

La sémantique des relations R_i est que si un symptôme S_j est détecté alors l'une des causes associées (les C_i) en est responsable. Dans cette écriture l'opérateur "ou" est noté " $\vee$ ".

La stratégie de contrôle consiste, à déterminer l'ensemble minimal de causes permettant d'expliquer la totalité des symptômes détectés.

Si l'on applique cette stratégie, en appelant L l'ensemble des hypothèses et V l'ensemble des solutions, nous avons les résultats suivants :

$$\begin{array}{lll}
 - \text{ à } t = t_1, & L = \{C1, C2\}, & V = \{ \}, \\
 - \text{ à } t = t_2, & L = \{ \}, & V = \{C2\} \\
 - \text{ à } t = t_3, & L = \{C1, C3\}, & V = \{ \} \text{ si on suppose vrai le résultat} \\
 & \text{à } t = t_2 \text{ sinon} & \\
 & L = \{ \}, & V = \{C1\} \text{ s'il est supposé faux.}
 \end{array}$$

Cette stratégie de contrôle n'est donc pas déterministe lorsqu'elle est appliquée au diagnostic en ligne. Nous pourrions en essayer d'autres, la conclusion serait la même.

II.2 Le diagnostic en ligne et l'incertain.

Dans les systèmes hors lignes, on est souvent amené à utiliser le raisonnement approximatif. Quelles sont les critères d'utilisation de cette forme de raisonnement ? Quelles sont les approches théoriques existantes ? Telles sont les questions auxquelles nous répondrons avant d'envisager si cette forme de raisonnement est applicable au diagnostic en ligne.

II.2.1 Nature de l'incertain dans les systèmes réels [TOR89].

Certains facteurs rendent difficiles l'utilisation de raisonnements catégoriques. Les principaux sont :

- L'incertitude inhérente aux connaissances même. En effet, souvent on ne peut pas établir des relations causales précises entre manifestations et les états d'un système. C'est le cas de systèmes très complexes comme le corps humain. On est alors obligé de coder des relations empiriques dérivées de l'expérience.
- L'incertitude liée à la détection ou recueil des symptômes. Cette incertitude est due au fait que les données initiales peuvent être bruitées (cas des signaux analogiques) ou manquer de précision. Par exemple, quelle valeur peut donner un médecin à la réponse d'un patient concernant l'intensité de ses céphalées ?
- Le flou issu de l'utilisation de termes linguistiques identiques pour décrire des situations complètement différentes.
- L'intérêt relatif de données différentes pour arriver au diagnostic (Cas où le diagnostic est obtenu à partir de plusieurs sources de données).

II.2.2 Différentes approches.

Il existe deux grandes familles d'approches pour modéliser l'incertain :

- les approches probabilistes,
- les approches basées sur l'utilisation de la logique floue.

La théorie probabiliste ([VIL88], [FRO90]) est la plus classique pour modéliser l'incertain. Mais elle présente d'importantes limites sur le plan mathématique. Nous en citerons essentiellement trois :

- La première concerne le contexte d'utilisation. Pour que cette théorie puisse être utilisée, il faut que d'une part l'ensemble des hypothèses soit exhaustif, et que d'autre part les hypothèses soient mutuellement indépendantes.
- La deuxième limite concerne la quantité de données nécessaire pour effectuer une classification Bayésienne.
- La troisième limite concerne la sensibilité du système par rapport à de petites erreurs sur les faits initiaux. Ces erreurs conduisent souvent à des écarts importants sur les résultats finaux.

La deuxième approche est basée sur l'utilisation de la logique floue [CAR90]. Son principal avantage réside dans une sémantique intuitive pour le concepteur. De plus, elle permet d'intégrer facilement des données relatives à des sources différentes.

Dans la pratique, peu de systèmes experts utilisent les approches précitées dans leur formalisme pur. La majorité des systèmes incluent des méthodes heuristiques dans leur mécanisme de résolution.

Comme systèmes utilisant le raisonnement incertain, on peut citer MYCIN, SAM [VOY87] ...

II.2.3 Contexte de la surveillance en ligne.

Dans le cadre de la surveillance en ligne, les symptômes sont générés automatiquement par le module de détection. Ils ont une sémantique précise (type et capteur) et correspondent à la violation du comportement temporel d'une partie du procédé. Il n'y a donc aucune incertitude à leur sujet si l'on admet que le système de détection est sain.

Dans les ateliers flexibles, l'utilisation de connaissances discrètes rend moins complexes les problèmes de modélisation. En fait tout est fonction du niveau d'abstraction auquel on descend. Ce niveau est lié à la précision souhaitée. Nous reviendrons en détail, sur cette question au chapitre III.

Parmi les objectifs du recouvrement de défaillances, nous pouvons citer le maintien d'un taux de production proche du taux planifié. Cela nécessite d'accroître la disponibilité moyenne des équipements. De ce fait, il est nécessaire d'avoir une gestion fine et rigoureuse de l'arrêt des

équipements qui présentent une anomalie. Par ailleurs l'identification en ligne, des composants défaillants des équipements en panne, permet de réduire les temps de réparation. Dans ce contexte quel intérêt aurait une conclusion du type : il y a 60% de chances que votre robot de chargement soit en panne et 40% que la machine d'usinage soit en panne ? A notre avis aucun. Donc les conclusions du diagnostic doivent être sûres.

Pour toutes ces raisons, nous pensons que l'utilisation de l'incertain ne se justifie pas dans le contexte de la surveillance en ligne des ateliers flexibles.

II.3 Conclusion.

La première conclusion concerne l'acquisition de la connaissance. Il faut une méthodologie différente basée sur une analyse systématique du système à surveiller. L'aspect en ligne impose de telles contraintes au niveau sécurité qu'on ne peut pas se permettre d'inférer à partir d'une base de règles incomplètes.

La deuxième conclusion est relative à l'omniprésence de la gestion du temps. Pratiquement tous les problèmes abordés se ramènent à cette question : comment gérer le temps ?

La troisième conclusion concerne les mécanismes de raisonnement. Notre étude nous a permis de mettre en évidence les limites des mécanismes classiques pour inférer un diagnostic certain dans le cadre de la surveillance en ligne. Même l'incertain est inapproprié aux problèmes que nous voulons traiter.

Faces à ces conclusions, nous allons proposer un certain nombre de solutions pour rendre possible le diagnostic en ligne. C'est l'objet des paragraphes suivants de ce chapitre.

III. MODELISATION DANS LE CADRE DU DIAGNOSTIC.

III.1 Introduction : objectif de la modélisation.

Toute modélisation d'un système est généralement dépendante de l'application dans laquelle elle sera utilisée. De ce fait on peut distinguer plusieurs modèles d'un même système :

- Des modèles pour la simulation. Ceux-ci diffèrent selon que l'on cherche à valider la commande [AMA90] ou à simuler le fonctionnement du procédé.
- Des modèles pour la détection. Ils sont bâtis sur le postulat que tout changement interne dans un procédé se manifeste par un changement de comportement. La démarche consiste donc à détecter les écarts entre le comportement du procédé réel et celui d'un modèle de référence. C'est cette approche que nous avons utilisée dans le chapitre II pour la détection de symptômes (modèle du comportement temporel).

Dans le cadre du diagnostic, on s'intéresse à l'origine des écarts perçus par la détection. De ce fait une modélisation pour le diagnostic, doit s'intéresser à la structure interne du procédé.

III.2 Classification des modèles de diagnostic.

Il existe plusieurs critères pour classer des modèles. On s'intéressera au degré de précision du modèle, à sa structure, et au type de comportement modélisé.

III.2.1 Modèle mathématique, modèle qualitatif.

Le degré de précision est à relier à la nature du modèle. On distingue alors deux types de modèles :

- les modèles mathématiques,
- les modèles qualitatifs.

Définition II.1 : Modèle mathématique [CHA89].

C'est une représentation abstraite du système, réalisée à partir d'outils mathématiques utilisés dans un cadre d'hypothèses précises.

L'utilisation des lois de la physique nous conduit à construire des modèles mathématiques.

Exemple : Modélisation d'un vérin hydraulique.

La commande du pré-actionneur représenté par le distributeur hydraulique permet le déplacement du piston dans le corps du vérin. Lorsqu'il est en fin de course, sa position est donnée par un capteur (Fig. II.8).

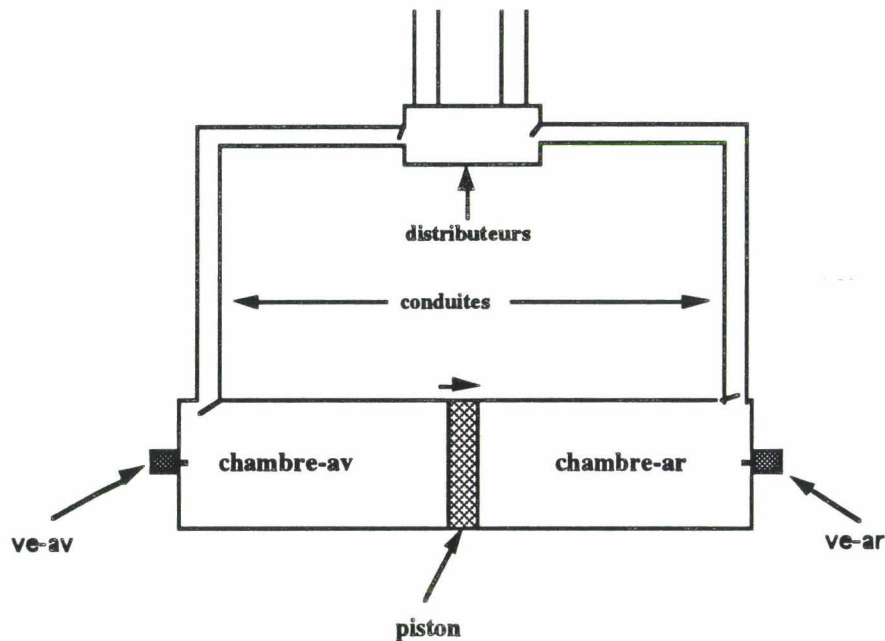


Figure II.8 : Synoptique d'un vérin hydraulique.

Si l'on fait l'hypothèse qu'il n'y a aucune fuite dans le circuit hydraulique (pas de perte de charge), l'application du principe fondamental de la dynamique permet de modéliser ce système. On obtient les équations suivantes :

$$m \, dV = (PS - fV) \, dt$$

$$dl = V \, dt$$

Remarque :

- fV représente la résistance due à la friction du piston sur le corps du vérin.

Un modèle mathématique offre une précision maximale.

Il n'y a pas de définition formelle des modèles qualitatifs. Nous pouvons dire que tout modèle non purement mathématique est un modèle qualitatif. De cette notion de modèles qualitatifs, est né le concept de physique naïve qui est à relier aux travaux sur les modèles de représentation mentale de l'homme [KLE80].

III.2.2 Modèle de représentation, modèle de connaissance [BRU90].

Définition II.2 : Modèle de représentation.

C'est un modèle qui traduit un comportement externe identique à celui du système réel sans que les mécanismes intérieurs aient été pris en compte.

Remarque :

Le modèle de représentation est donc un modèle d'abstraction qui se comporte comme une boîte noire dont on ne considère que les entrées et les sorties. On essaie alors de les relier par des équations.

Un exemple de modèle de représentation est la représentation d'état utilisée par les automaticiens.

Définition II.3 : Modèle de connaissance.

C'est un modèle qui analyse la structure du système, recherche la logique des mécanismes internes.

Remarque :

En automatique, un modèle de connaissance fait intervenir les variables internes.

III.2.3 Modèle du comportement normal, modèle du comportement défaillant.

Un modèle peut représenter soit le comportement normal, soit le comportement anormal. Il y a équivalence entre les deux approches si la représentation est complète. En effet, si le modèle modélise le comportement normal, cela signifie que tout ce qui ne correspond pas au comportement modélisé est par défaut anormal. Le raisonnement inverse peut être mené.

III.3 Conclusion.

La modélisation permet de résoudre les problèmes de complétude dans le cadre du diagnostic. Mais quel type de modélisation ?

L'intérêt de la modélisation mathématique, c'est sa précision. Mais avons-nous toujours besoin d'une telle précision dans le cadre du diagnostic ? Comme l'indique Chandrasekaran dans l'article intitulé " Deep models and their relation to diagnosis " [CHA89] la démarche des

experts humain est avant tout qualitative. Par des mécanismes qui peuvent même être heuristiques, ils focalisent leur recherche sur une partie du système. La complexité du problème de départ ayant été réduite, ils peuvent alors utiliser la modélisation mathématique pour terminer la résolution. Nous voyons donc que la modélisation qualitative est une solution pour concevoir un système complet temps réel.

Notre conception du diagnostic nous amène à identifier les défaillances et à en analyser les conséquences. Notre modélisation doit donc prendre en compte la structure interne du système. Notre démarche devra permettre de concevoir un modèle de connaissance du système surveillé.

Pour résoudre la contrainte temporelle, nous pensons qu'il est préférable de travailler sur un modèle du comportement anormal. En effet, sous cette forme, la connaissance apparaît comme pré-compilée. L'identification s'en trouve donc accélérée. Mais ce type de modélisation est plus délicat à mettre en oeuvre au niveau conceptuel. En effet, généralement, on connaît bien le comportement normal d'un système à commander. C'est celui spécifié dans le cahier des charges. Par contre il faudra analyser le système, pour spécifier son comportement défaillant.

Comment construire un modèle dont les caractéristiques correspondraient aux trois types cités précédemment ? Pour répondre à cette question, nous allons maintenant présenter la modélisation fonctionnelle.

IV. MODELISATION FONCTIONNELLE : ANALYSE DE L'EXISTANT.

IV.1 Introduction

La modélisation fonctionnelle est une forme de modélisation qualitative. Elle est à relier au concept de **modèles profonds** que nous nous proposons de définir.

Pour comprendre cette notion, il faut l'opposer à une autre qui a longtemps prévalu dans le cadre de la conception de systèmes experts : c'est le concept de **modèle de surface** [FIN85], [CHA89].

D'après Chandrasekaran [CHA89] la majeure partie des systèmes experts sont conçus à partir de modèles de surface (Fig. II.9). Ils sont caractérisés par une connaissance factuelle des

données ne leur permettant pas justifier leur raisonnement en se basant sur le fonctionnement des processus diagnostiqués. Les modèles de surface n'intègrent pas une représentation explicite de la structure et du comportement des composants du système : ce sont donc en fait des modèles de représentation.

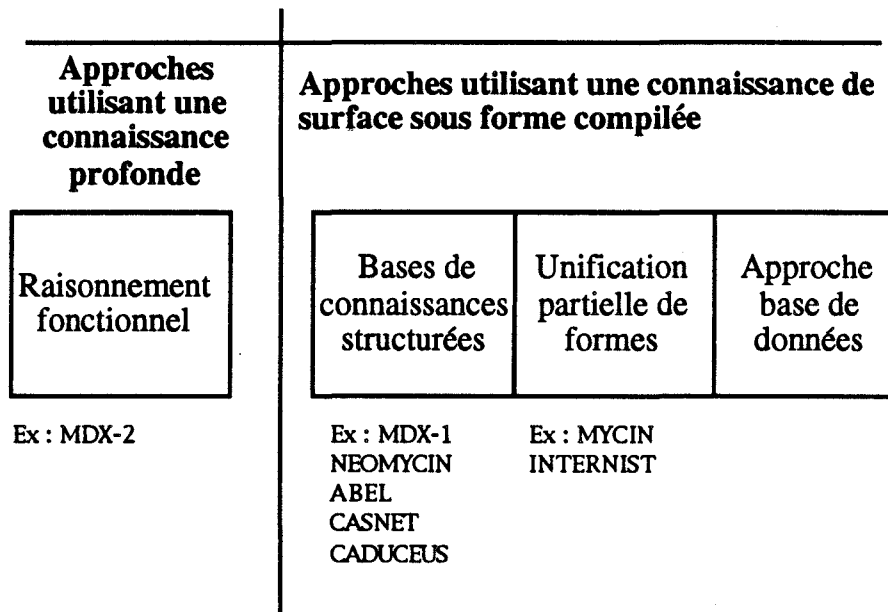


Figure II.9 : Classification des systèmes experts par approche de représentation de la connaissance.

En se référant toujours aux travaux de Chandrasekaran, nous pouvons caractériser les modèles profonds par les propriétés suivantes :

- ils contiennent des connaissances causales disponibles à des niveaux croissants de détails,
- ils contiennent des informations explicites caractérisant des liens causaux entre la structure et le comportement,
- ils fournissent des pointeurs vers d'autres types de connaissances exprimant des liens causaux.

Les modèles profonds sont donc des modèles de connaissance.

Afin d'affiner notre perception des modèles profonds et à travers elle la notion de modélisation fonctionnelle, nous allons présenter les principaux travaux relatifs à ces concepts.

IV.2 Analyse de l'existant.

IV.2.1 Raisonnements causaux ou associatifs : le système CHECK [TOR89].

IV.2.1.1 Réseau causal

Les réseaux causaux se présentent sous la forme de graphes constitués de noeuds et d'arcs. Mais ils ne doivent pas être confondus avec les graphes causaux. Ces derniers permettent seulement de coder des relations causales à différents niveaux d'abstraction (Fig. II.10).

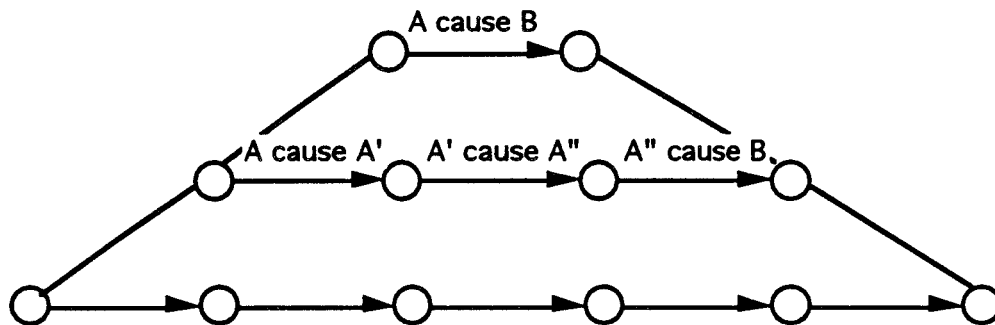


Figure II.10 : Graphe causal.

Des systèmes comme ABEL et CASNET utilisent ce type de représentation pendant leur mécanisme de résolution.

L'archétype du réseau causal est illustré par la modélisation des connaissances utilisées dans CHECK (Combining HEuristic and Causal Knowledge). Comme l'indique son nom, ce système combine des connaissances heuristiques et causales. Ici, nous allons nous intéresser seulement aux connaissances causales.

Dans CHECK, chaque noeud du réseau représente un concept abstrait. Dans ce cadre on distingue cinq types de noeuds : **HYPOTHESES**, **STATES**, **ACTIONS**, **INITIAL CAUSES** et **FINDINGS**.

Les noeuds du type STATE.

Ils servent à modéliser les états internes du système. Ces noeuds déterminent des états partiels (ou locaux). Ces états ne sont pas directement observables de l'extérieur.

Les noeuds du type ACTION.

Chaque noeud de ce type modélise une action associée à l'arc causal liant deux états internes. En fait une action ici est identifiée à l'ensemble des événements permettant la transition d'un état vers un autre.

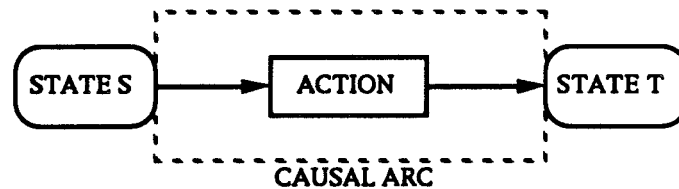


Figure II.11 : Structure d'un arc CAUSAL.

Les noeuds du type INITIAL CAUSE.

L'ensemble de ces noeuds modélise les causes premières, potentielles, des défaillances du système. Dans la théorie des graphes, ils correspondraient à des états puits. Ce sont donc des états particuliers.

Les noeuds du type FINDING.

Ils modélisent les manifestations externes produites par les défaillances du système. Ils sont donc équivalents à nos symptômes.

Les noeuds du type HYPOTHESIS.

Ils représentent les hypothèses de défaillances que la tâche de diagnostic pourrait être amenée à vérifier. On peut donc les assimiler à des manifestations internes. Dans CHECK, ces hypothèses sont normalement le résultat d'inférences menées par le niveau heuristique à partir de certains FINDINGS. Mais, plus généralement, elles permettent d'établir une ouverture sur des systèmes de localisation autre que le niveau heuristique de CHECK.

On distingue également plusieurs types d'arcs dans CHECK : CAUSAL, HAM, SUGGEST, DEFINED_AS, FORM_OF and LOOP.

Les arcs du type CAUSAL.

Ils modélisent des relations de cause à effet entre deux états ou un état et une cause initiale. En fait chaque arc CAUSAL est particularisé par un noeud ACTION donné (Fig. II.11).

Les arcs du type **HAM (Has As Manifestation)**.

Ils établissent un lien entre un état interne (noeud **STATE**) et sa manifestation externe (noeud **FINDING**).

Les arcs du type **SUGGEST**.

Ils mettent en relation des noeuds **STATES** ou **FINDINGS** avec des noeuds **HYPOTHESES**. Ils suggèrent d'examiner certaines hypothèses lorsque certains états internes (ou des manifestations externes) ont été établies. Les hypothèses associées à ce type d'arcs doivent être validées par d'autres systèmes que **CHECK**.

Les arcs du type **DEFINED_AS**.

Ils mettent en relation des noeuds **STATES** avec des noeuds **HYPOTHESES**. Les hypothèses considérées sont vérifiées par **CHECK** lui même. Il faut pour cela que les états internes correspondant aient été établis par l'inférence.

Les arcs du type **FORM_OF**.

Ces arcs établissent des relations entre les noeuds **HYPOTHESES**. Ces relations correspondent à une spécialisation entre les noeuds connectés. En pratique de tels arcs sont utilisés pour la génération d'explication.

Type de l'arc	Type du noeud origine	Type du noeud destination	Signification
Causal	State	Un State atteint à travers un Action	Relation de cause-effet
Ham	State	Finding	Manifestation
Defined-as	State	Hypothesis	Définition
Suggest	State ou Finding	Hypothesis	Suggestion
Loop	State	Un State atteint à travers un Action	Relation de cause-effet
Form-of	Hypothesis	Hypothesis	Spécialisation

Figure II.12 : Tableau récapitulatif des relations entre noeuds.

Les arcs du type **LOOP**.

Structurellement ce type d'arc est similaire aux arcs du type CAUSAL. C'est à dire qu'ils représentent des relations de cause-effets entre deux états internes. Mais la particularité de ce type d'arc est de permettre la représentation de circularités dans le modèle causal.

Définition II.4 : Un réseau causal.

C'est un ensemble de noeuds modélisant des concepts abstraits. Ces concepts sont liés par des relations typées appelées arcs.

IV.2.1.2 Résolution causale ou raisonnement associatif

Elle a pour fondement les relations causales. Elle consiste à partir des effets pour déterminer l'ensemble minimal de causes permettant de les expliquer.

Dans CHECK, la résolution causale est initiée par l'instanciation de noeuds HYPOTHESES. Cette instanciation est généralement le résultat d'une phase préliminaire de localisation, réalisée par une résolution heuristique. La résolution causale consiste alors à déterminer l'ensemble de causes initiales (INITIAL CAUSES) qui permettent d'expliquer ces hypothèses.

D'un point de vue formel, les auteurs expriment le problème du diagnostic par $P = \langle \text{NET}, \text{HYP}, \text{OBS} \rangle$. Dans cette formulation on a :

- NET : c'est le réseau causal du système modélisé,
- HYP = $\{ \langle H, \text{def}(H) \rangle / H \in \text{HYPOTHESES} \text{ et } \text{def}(H) = \text{DEFINED\_AS}^{-1}(H) \}$
- OBS = $\{ \text{OBS}^+, \text{OBS}^- \}$; OBS représente l'ensemble des observations potentielles pouvant être effectuées sur le système (FINDINGS). OBS^+ correspond aux observations relatives au cas traité. OBS^- est le complémentaire d' OBS^+ dans OBS.

La résolution causale s'effectue en deux étapes. La première étape consiste à déterminer l'ensemble des noeuds du type INITIAL CAUSE (IC) appartenant à des chemins étiquetés par des éléments de HYP. Cette détermination se fait en chaînage arrière en utilisant les relations de cause à effet dans le sens inverse.

La deuxième étape consiste à déterminer tous les chemins issus d'un noeud d'IC passant par un état interne ayant une manifestation externe présente dans OBS^+ . La détermination de ces chemins se fait par chaînage avant à partir des éléments d'IC. Si tous les états internes ayant

pour manifestation un élément d'OBS⁺, appartiennent à l'un de ces chemins, la résolution est terminée. L'ensemble des solutions est alors un sous-ensemble de IC.

IV.2.1.3 Conclusion

Dans leur formulation du problème du diagnostic, les auteurs de CHECK ont repris le formalisme introduit par Raymond Reiter [REI87] dans le cadre de la théorie logique. Mais la méthode proposée ici est basée sur le raisonnement abductif. Elle s'oppose donc aux approches de la théorie logique.

La méthode présentée dans CHECK n'en est pas moins intéressante. Elle propose une solution au diagnostic de systèmes complexes. Elle tient compte de l'hypothèse de défaillances multiples. Mais l'absence de raisonnement temporel ne permet pas de l'appliquer aux processus à évolution rapide comme les ateliers flexibles.

IV.2.2 La théorie logique.

IV.2.2.1 Introduction

Elle a été conceptualisée au début des années 80 par les travaux de Randal Davis et Johan de Kleer sur le raisonnement à base de modèles [KRA91], dans le cadre du diagnostic de circuits électronique. Mais elle doit également beaucoup à ceux de Genesereth sur l'utilisation de la logique dans le diagnostic automatique [REI87].

La théorie logique s'appuie sur une modélisation des systèmes selon deux axes : structurel et comportemental.

La décomposition structurelle consiste à considérer un circuit comme un ensemble de composants reliés physiquement et interagissant fonctionnellement les uns vis-à-vis des autres. Les relations entre composants sont modélisées par des contraintes.

Au niveau comportemental, la théorie logique exige que chaque composant soit décrit par des règles déterministes appelées contraintes. Ce principe tend à réduire le problème du diagnostic en un exercice formel de validation de théorème [KRA91]. Ce formalisme est proche des représentations utilisées dans l'analyse structurelle des systèmes à évolution continue [DEC89].

Nous allons présenter ici, les travaux de R. Reiter qui est l'auteur ayant formalisé cette théorie de la manière la plus rigoureuse. Pour une information plus complète sur cette théorie se reporter à l'annexe C.

IV.2.2.2 Approche de Reiter [REI87].[FRO90]

L'approche de Reiter est fondamentale dans cette présentation de la théorie logique. En effet, elle s'inspire des travaux des trois approches présentées en complément dans l'annexe C. Elle emprunte notamment, à Genesereth l'idée de modéliser le système à partir de formules logiques. Mais surtout, elle est la seule à avoir été formalisée avec rigueur.

IV.2.2.2.1 Modélisation.

Reiter définit formellement un système par la paire (SD, COMPONENTS) où :

- SD représente les descriptions du système. Ce sont un ensemble de primitives logiques. Elles modélisent les composants, la connectique du système et les comportements.
- COMPONENTS est un ensemble fini de constantes modélisant les composants du système.

Exemple :

Nous modélisons la carte électronique suivante :

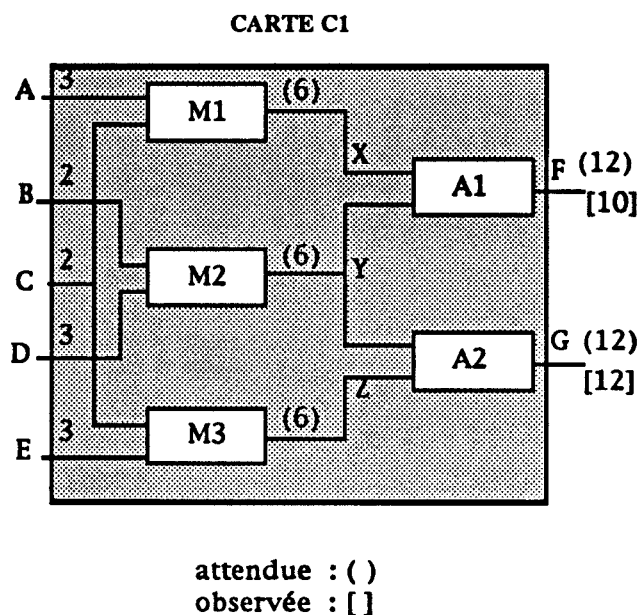


Figure II.13.

* COMPONENTS = { A1, A2, M1, M2, M3 }

* SD

- Typage des composants :

ADD(A1)
ADD(A2)
MULT(M1)
MULT(M2)
MULT(M3)

- Modélisation de la connectique :

out(M1)=in1(A1)
out(M2)=in2(A1)
out(M2)=in1(A2)
out(M3)=in2(A2)

- Modélisation du comportement :

$ADD(x) \wedge \neg AB(x) \Rightarrow out(x) = +(in1(x), in2(x))$
 $MULT(x) \wedge \neg AB(x) \Rightarrow out(x) = *(in1(x), in2(x))$

L'introduction du prédicat d'anormalité "AB" est une des originalités de cette approche. Il permet dans la modélisation de spécifier explicitement que le comportement d'un composant "x" est anormal (AB(x)).

La première règle de modélisation s'interprète de la manière suivante : si "x" est un additionneur (ADD(x)) et si "x" est non défaillant ($\neg AB(x)$) alors la sortie de "x" (out(x)) doit être égale à la somme de ses deux entrées (in1(x) et in2(x)).

IV.2.2.2.2 Résolution.

Le diagnostic d'un système est établi à partir d'un ensemble d'observation. Ces observations regroupent les valeurs en entrées et les valeurs de sorties. Dans le formalisme de Reiter, elles sont désignées par l'ensemble OBS.

Le but du diagnostic est de déterminer un sous-ensemble de "COMPONENTS" expliquant le comportement défaillant du système SD. Ce sous-ensemble doit vérifier le principe de Parcimonie.

Le principe de Parcimonie.

Un diagnostic est la détermination d'un ensemble minimal de composants défaillants.

Selon ce principe, Reiter pose alors formellement le problème du diagnostic selon la définition suivante :

Définition II.5 :

Un diagnostic pour (SD, COMPONENTS, OBS) est un ensemble minimal Δ

tel que :

$$SD \cup OBS \cup \{ AB(c) / c \in \Delta \} \cup \{ \uparrow AB(c) / c \in COMPONENTS - \Delta \}$$

soit cohérent.

Exemple :

Nous reprenons la carte électronique donnée par la figure II.13.

Pour cet exemple, l'ensemble des observations est défini par :

OBS = { in1(M1)=3, in2(M1)=2, in1(M2)=3, in2(M2)=2, in1(M3)=3, in2(M3)=2, out(A1)=10, out(A2)=12 }.

Avec ces observations la solution est donnée par la réunion des sous-ensembles Δ suivants :

$\Delta_1 = \{ A1 \}$, $\Delta_2 = \{ M1 \}$, $\Delta_3 = \{ M2, M3 \}$ et $\Delta_4 = \{ M2, A2 \}$.

Dans les paragraphes suivants, nous précisons comme est obtenue cette solution.

La résolution est formalisée en utilisant les notions de **conflits et d'ensembles ciblés**. La notion de conflits est la même que celle utilisée par de Kleer [KLE84], [KLE87].

Définition II.6 : Un ensemble de conflits.

Un ensemble de conflits pour (SD, COMPONENTS, OBS) est un ensemble $\{c1, c2, \dots, ck\}$ COMPONENTS tel que :

$$SD \cup OBS \cup \{ \uparrow AB(c1), \uparrow AB(c2), \dots, \uparrow AB(ck) \}$$

soit incohérent.

Définition II.7 : Ensemble ciblé.

Soit F un ensemble d'ensembles. Un **ensemble ciblé** de F est un ensemble H tel que $H \subseteq \bigcup_{S \in C} S$, avec $H \cap S \neq \{ \}$ pour tout S de F .

H est minimal si l'on ne peut pas trouver de sous-ensemble ciblé de H .

Le mécanisme de résolution est basé sur le théorème suivant :

Théorème II.1 :

$\Delta \subseteq \text{COMPONENTS}$ est un diagnostic de $(SD, \text{COMPONENTS}, \text{OBS})$ si Δ est un ensemble ciblé minimal pour l'ensemble des ensembles de conflits de $(SD, \text{COMPONENTS}, \text{OBS})$.

Pour mettre en oeuvre ce théorème, la résolution est divisée en deux étapes :

1 - La première étape consiste à déterminer l'ensemble des conflits noté F . La construction utilise un algorithme appelé **TP** (Theorem Prover). Pour des raisons d'efficacité, F est construit de manière implicite au cours de la deuxième étape.

2 - La deuxième étape consiste à construire un arbre de résolution appelé **HS-tree** (i.e. l'arbre des ensemble ciblés) et à en extraire les ensembles ciblés minimaux. Cette étape utilise **DIAGNOSE**, un algorithme récursif (**DIAGNOSE**($SD, \text{COMPONENTS}, \text{OBS}$)) faisant lui même appel à **TP**.

Un **HS-tree** est un arbre dont les noeuds sont étiquetés par des sous-ensembles de **COMPONENTS**, et les arcs par des éléments de **COMPONENTS**. La construction des noeuds est réalisée de proche en proche à partir de la racine.

1 - La racine est étiquetée par l'appel de **TP** appliquée à $(SD, \text{COMPONENTS}, \text{OBS})$. Si aucun conflit n'est trouvé, le résultat est l'ensemble vide (étiquette "nil") et la résolution est terminée. Sinon on détermine un ensemble de composants inclus dans **COMPONENTS**.

2 - Soit " n " un noeud quelconque. On définit $H(n)$ comme l'ensemble des étiquettes du chemin allant de la racine au noeud " n ". Notons Σ le sous-ensemble de **COMPONENTS** étiquetant " n ".

Si $\Sigma = \text{nil}$ alors " n " n'a pas de successeur dans l'arbre.

Si $\forall \sigma \in \Sigma$, "n" a un noeud successeur " n_σ ", relié à "n" par un arc étiqueté par σ . Notons S l'étiquette de " n_σ ". S est un sous-ensemble de COMPONENTS devant vérifier deux équations :

$$- S = TP(SD, COMPONENTS - H(n_\sigma), OBS)$$

$$- S \cap H(n_\sigma) = \{ \}$$

Si l'on ne peut pas trouver un élément de F vérifiant ses deux équations alors $S = \text{nil}$.

Le HS-tree est construit en largeur de manière à ré-utiliser les résultats des appels antérieurs de TP. Des algorithmes sont utilisés pour réduire la combinatoire en coupant certaines branches de l'arbre (noeuds marqués par une étoile).

Les ensembles $H(n)$ étiquetant les chemins reliant la racine à un noeud étiqueté par "nil" sont des ensembles ciblés minimaux. Le diagnostic est donné par l'ensemble composé de ces ensembles ciblés minimaux.

Si nous reprenons l'exemple du circuit électronique de la figure II.13, on pourrait trouver que $TP(SD, COMPONENTS, OBS) = \{A1, M1, M2\}$ (Le résultat n'est pas unique. L'ensemble $\{M3, A2, M1, A1\}$ est également une solution. De la racine on tire trois arcs qui seront étiquetés respectivement par A1, M1, M2 (Fig. II.14). Ainsi on trouve que $TP(SD, COMPONENTS - \{A1\}, OBS) = \text{nil}$, $TP(SD, COMPONENTS - \{M2\}, OBS) = \{M3, A2, M1, A1\}$... Le développement de l'arbre se poursuit jusqu'à ce que tous les noeuds soient marqués ou que l'on tombe sur une branche coupée (noeud étiqueté par une "*") par un algorithme de simplification.

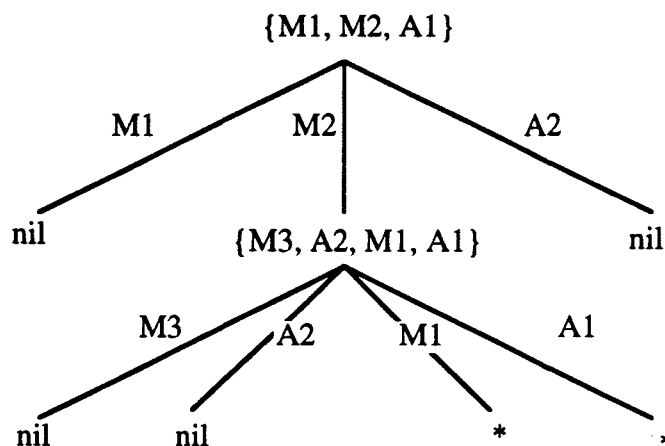


Figure II.14 : Un HS-tree simplifié pour le circuit électronique [REI87].

IV.2.2.3 Conclusion

La majeure partie des travaux actuels sur la théorie logique ont adopté le formalisme de Raymond Reiter. Mais aujourd'hui, de nombreux auteurs cherchent à étendre ces concepts de base. Notamment, le diagnostic ne consiste plus à étiqueter chaque composant comme normal ou anormal. Certains travaux utilisent des approches probabilistes ou floues pour qualifier l'état des composants de manière plus nuancée [KRA91].

Bien qu'intéressante, la théorie logique présente quelques limites du point de vue du diagnostic en ligne des systèmes manufacturiers.

La première limite tient au formalisme utilisé qui nécessite que le comportement des systèmes puisse être modélisé par des prédicats logiques. Les systèmes électroniques et digitaux s'y prêtent bien. Ce n'est pas le cas de systèmes complexes comme ceux que l'on rencontre dans l'industrie manufacturière.

Par ailleurs, l'ensemble des travaux développés jusqu'à présent dans cette théorie sont fondés sur l'hypothèse d'un système statique. C'est à dire que les observations (OBS) n'évoluent pas au cours du temps. Cette hypothèse est évidemment restrictive dans le contexte de la surveillance des systèmes manufacturiers (cf. § II). L'utilisation d'une logique temporelle [BES89], [AUD90] pourraient être le moyen d'adapter la théorie logique aux contextes des systèmes évolutifs.

La troisième limite concerne la forme des résultats. Toutes les approches donnent la solution sous la forme d'une disjonction de sous-ensembles minimaux représentant les défaillances de composants dont la conjonction pourrait expliquer le comportement observé du système. La résolution n'est donc pas déterministe.

IV.2.3 Fonctionnel, structurel et comportemental.

Dans cette partie nous allons présenter les travaux de S. Amar et ceux de B. Chandrasekaran sur la modélisation fonctionnelle. L'intérêt de ces travaux est qu'ils distinguent trois composantes dans une modélisation fonctionnelle :

- la composante fonctionnelle,
- la composante structurelle,
- la composante comportementale.

IV.2.3.1 Approche de S. Amar [AMA90].

L'approche de S. Amar se situe dans le cadre de la validation par simulation des différentes composantes d'un système de production flexible (PC, PO). Son but est d'assister la conception d'un **modèle dynamique** du procédé. Contrairement à toutes les autres approches présentées dans cette analyse de l'existant, elle ne s'inscrit donc pas dans le contexte de la surveillance. Notre intérêt pour cette approche est qu'elle représente les premiers travaux de modélisation entrepris dans le cadre de la méthodologie CASPAIM. De ce fait notre, approche a été fortement influencée par ces travaux.

Du point de vue du concepteur, la démarche commence par une modélisation structurelle du procédé. Elle se poursuit de proche en proche jusqu'à la localisation des composants de base représentés par les actionneurs. A ce stade, nous disposons d'un modèle statique représentant l'organisation hiérarchique des composants du système.

Pour prendre en compte la dynamique du système, le concepteur dispose de deux bibliothèques :

- la bibliothèque fonctionnelle,
- la bibliothèque comportementale.

La bibliothèque fonctionnelle a été établie à partir d'une analyse générale des fonctions existant dans un système manufacturier. Les fonctions sont vues dans ce contexte sous l'angle de la commandabilité (exception faite du stockage). Ainsi, ces travaux ont permis de dégager un certain nombre de fonctions génériques (positionnement, transfert, serrage, stockage ...) organisées en hiérarchie fonctionnelle (Fig. II.15).

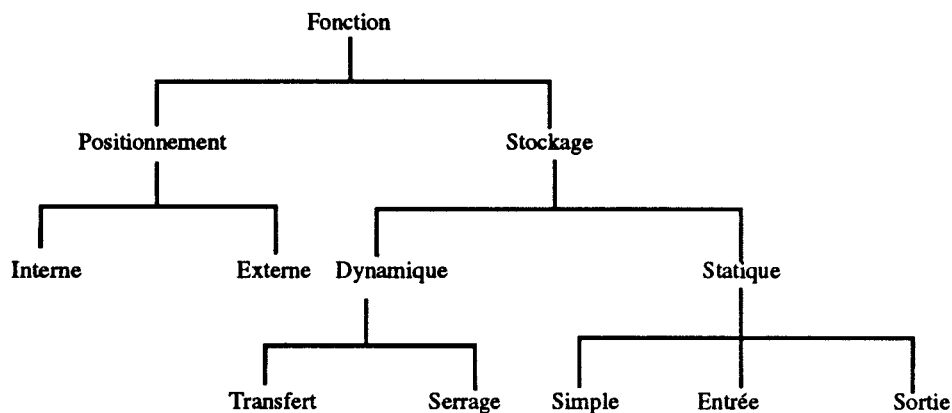


Figure II.15 : Classification fonctionnelle d'après [AMA90].

Les feuilles de la hiérarchie ainsi obtenue modélisent les fonctions de base de tout système flexible.

La tâche du concepteur est alors de "mettre en relation" des composants du modèle structurel avec des fonctions de base (Fig. II.16). Le modèle obtenu est un modèle structuro-fonctionnel caractérisant les entités commandables élémentaires du système i.e. toutes les entités possédant en propres au moins un actionneur.

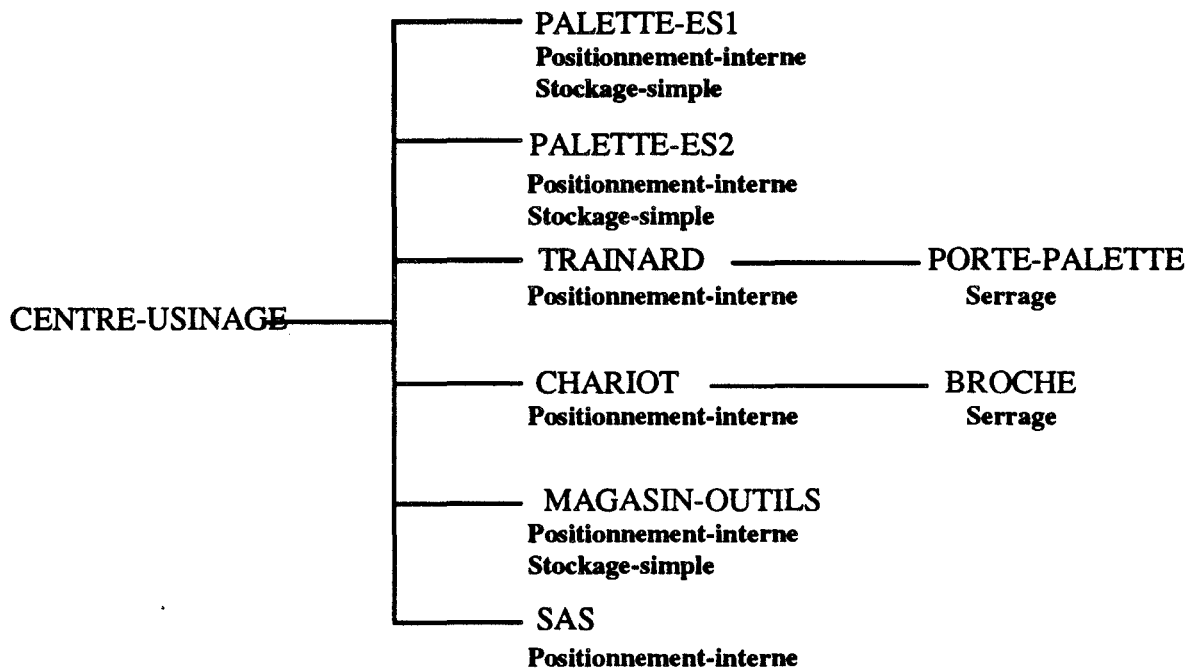


Figure II.16 : Association de fonctions aux composants d'un centre d'usinage d'après [AMA90].

La bibliothèque comportementale a également été établie à partir d'études sur la généricité des comportements des organes de la Partie Opérative. Un certain nombre d'actionneurs types ont été identifiés (moteurs, vérins ...). Le comportement de chaque actionneur est modélisé par un automate à états finis (Fig. II.17).

Chaque modèle comportemental élémentaire caractérise une ou plusieurs fonctions de base d'un modèle structuro-fonctionnel.

Pour obtenir le comportement global d'une fonction de niveau supérieur, le concepteur doit intégrer des **contraintes de fonctionnement**. Par exemple, on peut interdire le positionnement du bras d'un robot pendant une action de serrage de sa pince et inversement.

Cette interdiction se fait en contraignant mutuellement les modèles de comportement de ces deux entités. Le graphe résultant modélise le comportement normal de la ressource robot.

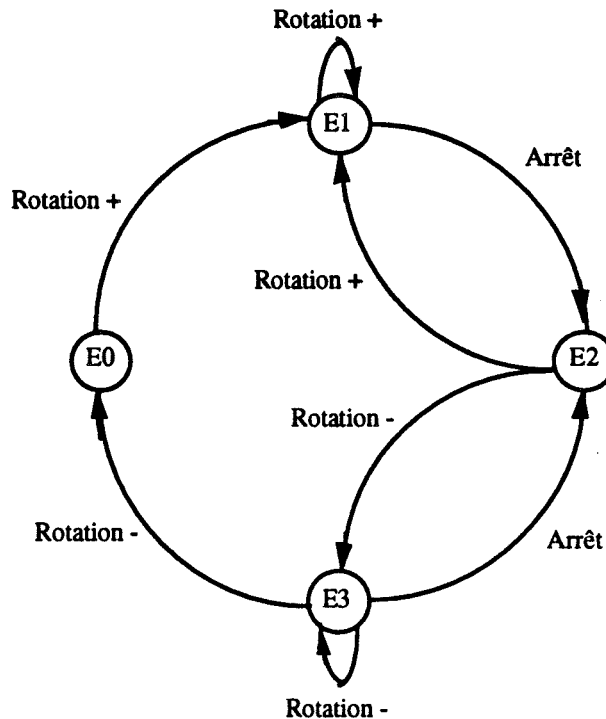


Figure II.17 : Modèle comportemental d'un moteur à double sens de marche.

IV.2.3.2 Approche de B. Chandrasekaran [CHA83/a], [CHA83/b], [CHA86], [CHA89].

Les travaux de Chandrasekaran se situent dans le cadre du diagnostic hors ligne de systèmes complexes. Deux idées essentielles permettent de les comprendre.

La première idée est qu'il faut trouver un compromis entre d'une part la précision et la complétude des connaissances d'un système et d'autre part l'efficacité du système de diagnostic. Il est admis que la représentation sous forme de règle de production ("connaissance compilée") est celle garantissant le plus d'efficacité à un système expert. Par contre les connaissances codées sous cette forme sont difficilement maintenables. D'où l'idée de représenter tout système à l'aide d'un formalisme orienté objet. Le modèle est ensuite compilé afin de générer des règles de production.

La deuxième idée qui sous-tend ces travaux est la nécessité de gérer les explications données à l'opérateur, par le système de diagnostic. La génération d'explication nécessite

l'utilisation d'une modélisation profonde du système diagnostiqué ; d'où l'intérêt d'une modélisation fonctionnelle.

Comme dans l'approche de S. Amar, le système est d'abord modélisé structurellement.

Définition II.8 : Structure d'un système [SEM86]

La structure spécifie les relations entre composants d'un même niveau, et l'abstraction à un niveau donné des composants du niveau inférieur.

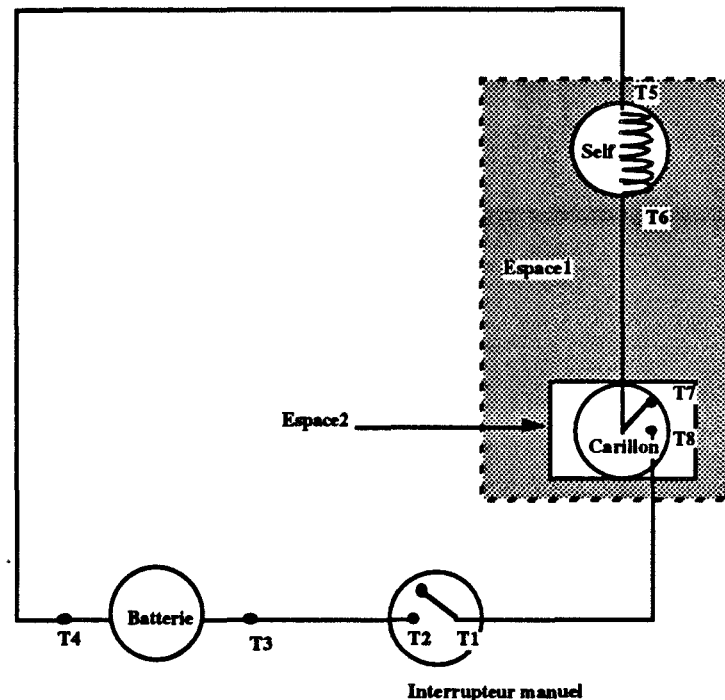


Figure II.18 : Sonnerie électronique.

Considérons par exemple, le système décrit par la figure II.18. Il représente une sonnerie électronique dont les principaux composants sont :

- un interrupteur manuel,
- une batterie,
- une self,
- un carillon.

L'ensemble de ses composants sont connectés en série.

Le modèle structurel de la sonnerie est le suivant :

DEVICE : sonnerie

STRUCTURE :

COMPONENTS :

interrupteur(T1,T2), batterie(T3,T4),
self(T5, T6, espace1), carillon(T7,T8, espace2).

RELATIONS :

en-série(interrupteur, batterie, self, carillon),
inclus(espace1,espace2).

ABSTRACTIONS-OF-COMPONENTS :

COMPONENT : carillon (t1, t2, espace)

STATES : elect-connecté(t1, t2),
répétition-de-coups(carillon)

FUNCTIONS : mécanique, acoustique,
magnétique.

COMPONENT :

STATES :

FUNCTIONS :

END STRUCTURE

Dans cette description STATES spécifient les états d'un composant connu par le système, et FUNCTIONS ses fonctions.

Le modèle structurel est ensuite transformé en un modèle structuro-fonctionnel, en intégrant les fonctions de chaque entité (système et composants). Dans la suite de ce paragraphe, nous nous contenterons de développer la spécification de l'entité "sonnerie".

Au sens de Chandrasekaran, une fonction spécifie la réponse d'un système ou d'un composant à un stimulus. Donc toute entité physique possède au moins une fonction. Les fonctions sont réalisées par des comportements (Fig. II.19).

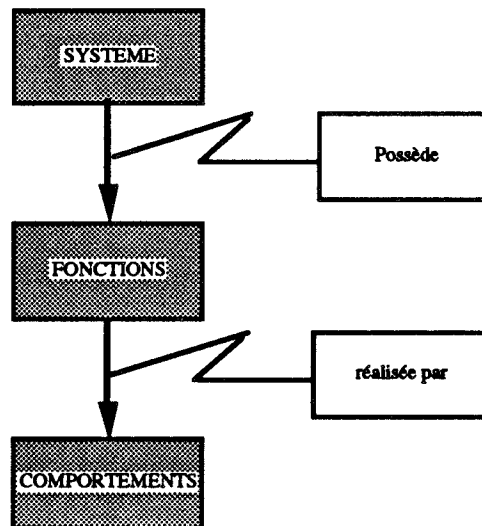


Figure II.19 : Composantes d'une représentation fonctionnelle d'après Chandrasekaran.

Dans l'exemple de la sonnerie électrique, les fonctions sont données de la manière suivante :

DEVICE : sonnerie

FUNCTIONS :

sonner : **TOMAKE** sonnant(sonnerie)

IF appuyé(interrupteur)*

PROVIDED hypothèse1

BY comportement1

arrêter-de-sonner :

END FUNCTIONS

STRUCTURE :

END STRUCTURE

Dans cette spécification (appuyé(interrupteur)*) et (sonnant(sonnerie)) représentent des états de la sonnerie vue de son environnement. Les termes **TOMAKE**, **IF**, **PROVIDED**, **BY** sont des expressions du langage de spécification qui seront utilisées lors de la compilation sous forme de règles de production. Nous avons représenté sur la figure II.20, la fonction "sonner" de la sonnerie. Pour en faciliter la compréhension, nous avons utilisé les Réseaux de Petri

Interprétés (RdPI) comme outil de modélisation. Dans cette représentation, les places modélisent des états du système. Au niveau des transitions sont représentées les connaissances permettant de passer d'un état dans un autre.

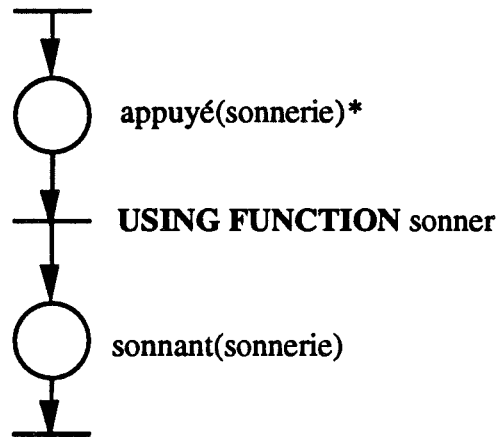


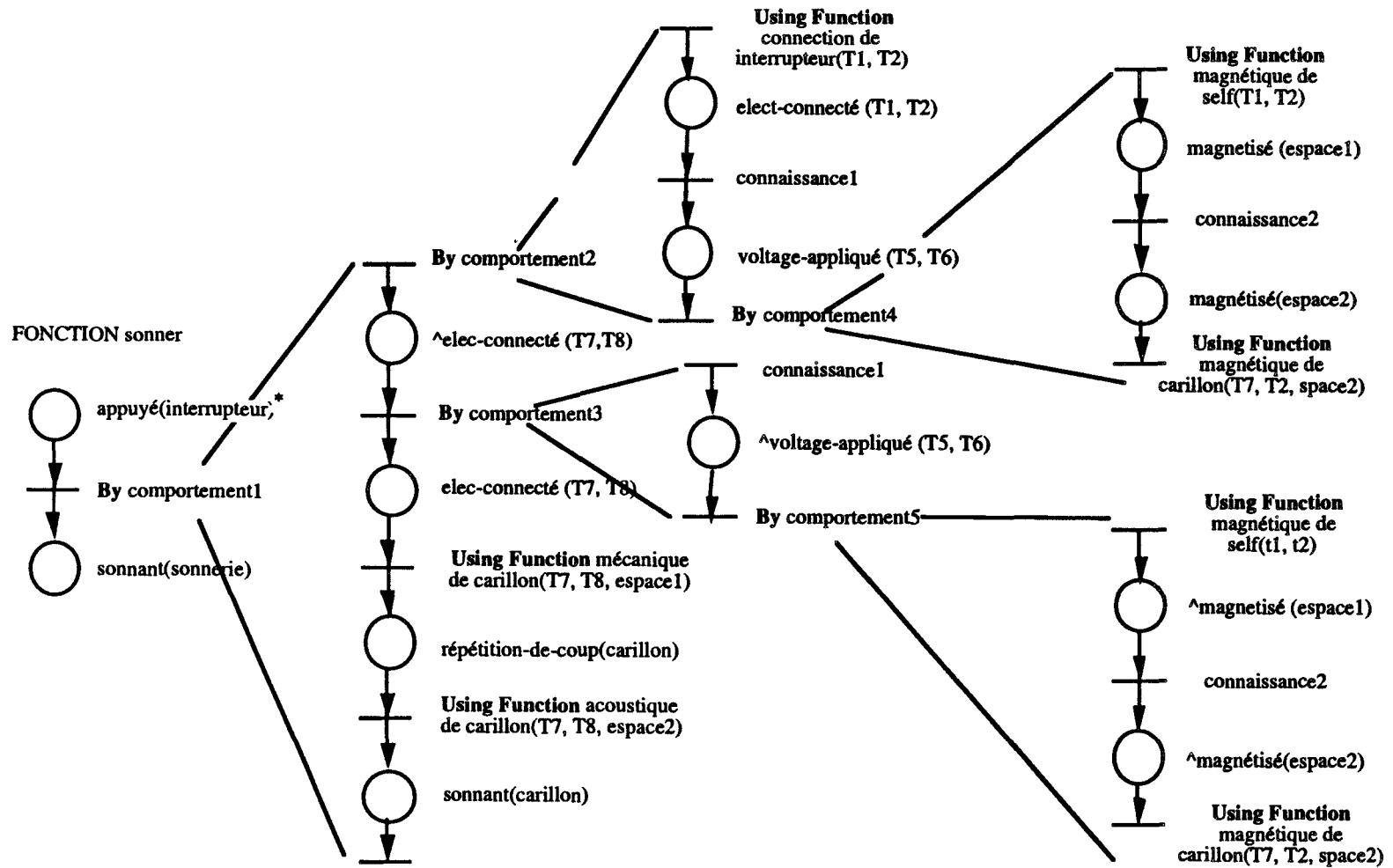
Figure II.20 : Modélisation de la fonction "sonner".

L'auteur définit un comportement comme une succession d'états internes. Les états internes du système sont en fait des états de ses composants spécifiés dans la description structurelle. Les transitions entre ces états internes sont établis à partir de trois sources de données :

- les fonctions de ses composants ; elles sont introduites au niveau de la spécification par **USING FUNCTION** ;
- des connaissances génériques tels que les lois de Kirshoff en électricité ; dans notre exemple la "connaissance2" identifie la relation causale entre la magnétisation de l'espace1 et celle de l'espace2.
- des hypothèses explicites sur le fonctionnement du système. Un exemple d'hypothèse (hypothèse1) est que soumis au champ magnétique de la self, la force magnétique induite sur le frappeur (composant du carillon) est supérieur à la force de rappel du ressort chargé de maintenir le contact sur la borne T8 (cf. Figure II.18).

Sur la figure II.21, nous avons représenté par RdPI comment le comportement1 s'exprime en fonction des états et des fonctions des composants de la sonnerie.

Figure II.21 : Modélisation par RdPI du "comportement1" relatif à la fonction *sonner* de la sonnerie.



IV.2.3.3 Conclusion

Les deux approches que nous avons présentées sont intéressantes dans la mesure où elles introduisent toutes deux une distinction nette entre les concepts de fonction et de comportement d'une entité physique. Au niveau de la sémantique des concepts de base et de la mise en oeuvre de la modélisation, elles présentent toutefois des différences.

Dans l'approche de S. Amar, les fonctions sont vues uniquement sous l'angle de la commandabilité ou des capacités de stockage. Cela explique que des fonctions tels que celles du roulement de broche d'un centre d'usinage, ne seront pas prises en compte dans sa modélisation fonctionnelle. Pourtant du point de vue des défaillances potentielles du système, ce sont des fonctions importantes (cf. chapitre 1).

Par ailleurs contrairement à l'approche de Chandrasekaran, il n'est pas explicitement associé un comportement à chaque fonction. Seules les fonctions de base ont un comportement. Celui des fonctions des niveaux supérieurs est hérité en raison de la relation qui lie toute fonction de haut niveau, à des fonctions du niveau inférieur.

Dans la modélisation de Chandrasekaran, les fonctions sont indépendantes de l'environnement qui les sollicite. De ce fait, toute entité physique remplit au moins une fonction. Par contre les fonctions non élémentaires sont créées par la combinaison de fonctions des constituants qui composent cette entité : c'est la notion de comportement. Dans cette démarche, les fonctions de plus bas niveaux n'ont pas de comportement. Elles sont vues comme des fonctions génériques ; c'est à dire des connaissances indivisibles.

L'ajout dans la description structurelle de connaissances du type "en-série (interrupteur, batterie, ...)", nous semble en contradiction avec la notion de comportement. En effet, c'est une donnée redondante car en fait elle exprime une relation fonctionnelle entre composants d'un même niveau.

Les différences contenues dans les modèles issues des deux approches s'expliquent également par des démarches différentes. S. Amar introduit les fonctions dans le modèle structurel, selon une démarche ascendante. On part des feuilles de la hiérarchie structurelle. Dans l'approche de Chandrasekaran, c'est le contraire. On part de la racine et on affine au fur et à mesure jusqu'aux composants dont les fonctions sont considérées comme génériques.

CONCLUSION

Dans cette partie, nous avons présenté l'état de l'art sur les approches de modélisation fonctionnelle. Nous avons également exposé les mécanismes de résolution des approches orientées diagnostic.

Ces approches présentent des inconvénients qui ne nous permettent pas de les exploiter tel quel. Le principal inconvénient réside dans l'absence d'intégration des aspects temporels. Par ailleurs certaines approches comme celles de la théorie logique semblent trop spécifiques à un domaine d'application. C'est la raison pour laquelle, tout en nous inspirons du formalisme de Reiter, notre approche se situera plus dans le sens d'une démarche abductive.

L'analyse de ses différentes approches permet de dégager les composantes nécessaires à une modélisation fonctionnelle. Il s'agit principalement des composantes :

- structurelle,
- fonctionnelle,
- comportementale.

CHAPITRE III

LE DIAGNOSTIC : MODELISATION ET RAISONNEMENTS

SOMMAIRE

I - Modélisation fonctionnelle pour le diagnostic en ligne.

II - Exploitation du modèle fonctionnel.

III - Spécifications informatiques.

INTRODUCTION

Dans le chapitre précédent, nous avons montré que la modélisation fonctionnelle est une forme de modélisation qualitative parfaitement adaptée à la résolution des contraintes induites par l'aspect en ligne. L'analyse de l'existant ne nous a toutefois pas permis de trouver des méthodes de modélisation intégrant toutes ces contraintes. Ce chapitre sera consacré à la proposition d'une méthode adaptée à notre problématique. Il comprend essentiellement trois parties :

- * La première partie concerne la présentation de notre méthode de modélisation fonctionnelle. Elle conduit à différencier modèle fonctionnel et modèle structurel.
- * La deuxième partie est axée sur l'exploitation du modèle fonctionnel. Nous proposons notamment un algorithme de positionnement des capteurs pour assurer l'observabilité du procédé surveillé. A partir des caractéristiques structurelles du modèle fonctionnel, nous proposons plusieurs mécanismes de raisonnement intelligent, pour mettre en œuvre le diagnostic : ce sont le diagnostic local, l'identification, la validation et le pronostic.
- * Dans la troisième partie nous proposons les spécifications pour la mise en œuvre informatique des mécanismes proposés précédemment.

I. MODELISATION FONCTIONNELLE POUR LE DIAGNOSTIC EN LIGNE.

I.1 Définitions des concepts de base.

L'analyse de tout système physique permet de mettre en évidence deux types de relations inter-composants :

- des **relations verticales** exprimant la structure matérielle opérative selon une relation descendante,
- des **relations horizontales** exprimant les coopérations et échanges entre composants.

Les relations verticales expriment des liens de composition. Ainsi, on dira que le système est constitué d'un certain nombre de systèmes élémentaires. A leur tour, ces systèmes

élémentaires peuvent être subdivisés en sous-ensembles. On peut répéter le processus de décomposition jusqu'au niveau des pièces, constituants de base du système (Fig. III.1).

(pièce) $\subset$ (composant) $\subset$ (sous-système) $\subset$ (système-élémentaire) $\subset$ (système)

Figure III.1.

Ainsi, les relations verticales permettent de caractériser un atelier flexible par les machines et moyens de transport qui le constitue.

Les relations horizontales expriment les inter-actions entre composants d'un même niveau d'abstraction (au sens des relations verticales).

Ces inter-actions sont généralement d'origine énergétique : flux électrique, flux hydraulique, flux pneumatique ou flux de matière (pièces et outils). Si l'on considère un convoyeur à bandes, les relations verticales permettent de distinguer des composants de trois natures :

- les **postes** qui servent de zone de travail ou de stockage temporaire,
- les **tronçons** qui sont des zones de transit mettant en relation des postes,
- les **redirecteurs** (aiguillages et jonctions) qui gèrent les inter-actions entre tronçons.

Ces différents composants interagissent par le biais des pièces circulant sur le convoyeur.

Si l'on prend l'exemple du convoyeur décrit figure II.4 (Chapitre II), les relations horizontales permettent de les représenter par la figure III.2.

Les inter-actions entre composants d'un même niveau peuvent également être d'origine mécanique. C'est par exemple le cas lorsque deux robots coopèrent dans le cadre d'un assemblage.

La figure III.2 s'apparente aux schémas blocs couramment utilisés en automatique. On peut donc identifier les relations horizontales à des liens fonctionnels.

Remarque :

Les relations verticales sont statiques tandis que les relations horizontales permettent de caractériser la dynamique du système.

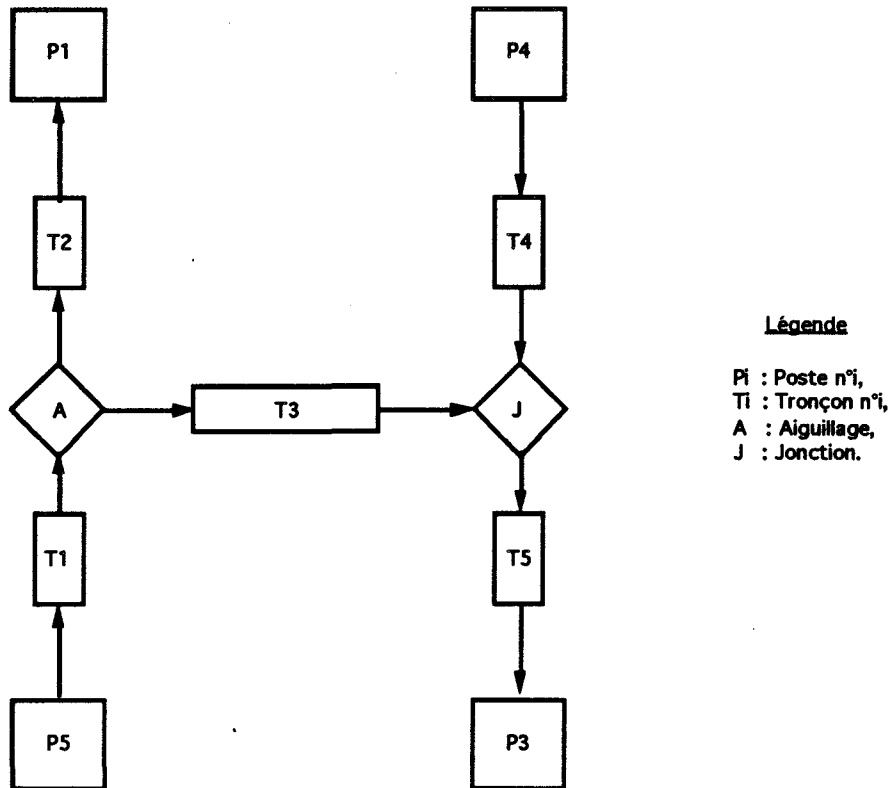


Figure III.2.

Sur la base de l'analyse précédente, nous pouvons donner les définitions suivantes :

Définition III.1 : Lien structurel.

Un lien structurel est une relation verticale entre deux composants d'un système appartenant à des niveaux d'abstraction adjacents.

Définition III.2 : Lien fonctionnel.

Un lien fonctionnel est :

- une relation horizontale entre deux composants d'un système appartenant au même niveau d'abstraction,
- une relation finale entre le système (ou un composant d'un niveau supérieur) et l'un de ses composants, positionné en fin d'une composition de relations horizontales.

Les liens structurels et fonctionnels établissent une double hiérarchie au sein d'un système (Fig. III.3).

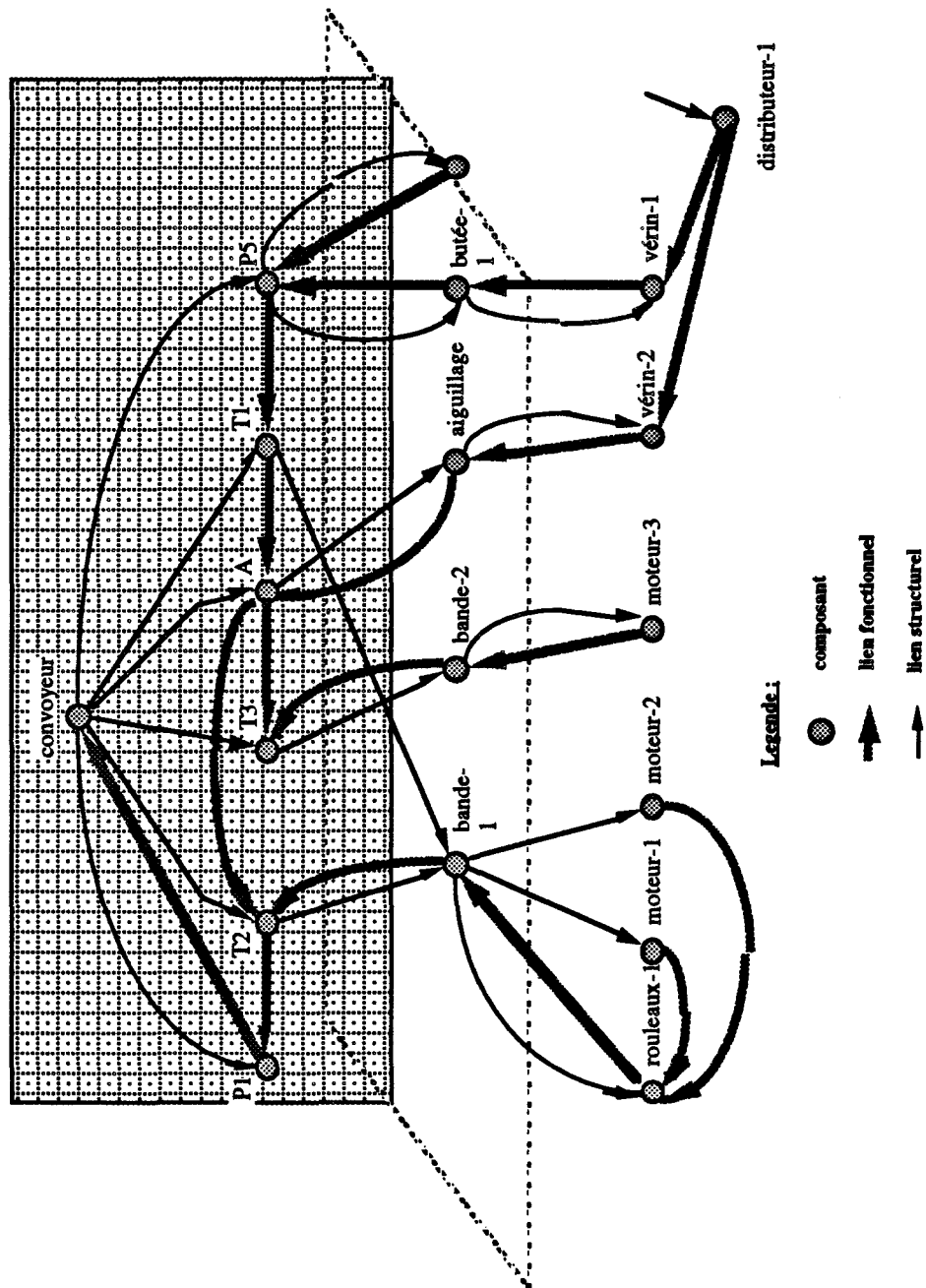


Figure III.3 : Sémantique des liens inter-composants.

I.2 Modèle fonctionnel et modèle structurel : définitions et différenciation.

Lorsqu'on effectue la synthèse d'un système on commence toujours par créer un modèle fonctionnel (Schémas fonctionnels). Le modèle fonctionnel permet de concevoir les fonctions

du système à partir de fonctions élémentaires. Prenons l'exemple en logique de la synthèse d'un additionneur 1 bit. Fonctionnellement, il est équivalent à un XOR. Mais on pourra le réaliser avec cinq portes NAND si l'on ne dispose pas de XOR. Cet exemple permet de tirer deux conséquences :

- il y a primauté du modèle fonctionnel sur le modèle structurel (Principe de Primauté),
- il n'y a pas unicité du modèle structurel mettant en œuvre un modèle fonctionnel.

Dans le contexte de la modélisation de systèmes existant, l'analyse s'effectue toujours en partant de la structure. On a alors tendance à définir les fonctions, composantes du modèle fonctionnel, à partir d'une structure donnée. Cela est en contradiction avec le principe de primauté. De Kleer et Brown [KLE80] ont proposé un principe fondamental à observer lors de la construction d'un modèle fonctionnel.

Principe d'indépendance (No-function-in structure).

Aucune fonction ne doit être spécifiée en se référant à la structure ou au comportement du système environnant.

Le principe d'indépendance permet de définir pour toute entité physique un type particulier de fonctions : les fonctions génériques. La détermination de fonctions génériques par famille de composants permet d'assister la démarche de modélisation fonctionnelle à l'aide d'outils de CAO. Les travaux de S. Amar [AMA90] menés dans le cadre du projet CASPAIM ont pour but la conception de tels outils.

Définition III.3 : Modèle fonctionnel.

Le modèle fonctionnel spécifie les différentes fonctions d'un système à partir de la composition de fonctions élémentaires. Cette composition est réalisée en définissant des liens fonctionnels entre fonctions.

Remarques :

- Une fonction est élémentaire du point de vue de la maintenance.
- Les liens fonctionnels établis entre fonctions du modèle créent une structure hiérarchique. Mais cette structure hiérarchique est différente de celle établie dans le modèle structurel équivalent.

Définition III.4 : Modèle structurel.

La structure spécifie les composants d'un système et leurs liens structurels.

I.3 Méthode de conception d'un modèle fonctionnel

Cette méthodologie est basée sur la notion de fonction. Dans un premier temps, nous allons donc présenter notre interprétation du concept de fonction. Ensuite nous donnerons les différentes étapes de la méthodologie. Nous illustrerons ces étapes par le traitement d'un exemple.

I.3.1 Notion de fonction.

Ce concept est le fondement de notre approche. Pour l'introduire, nous allons partir de la définition d'une fonction selon le point de vue de B. Chandrasekaran.

Définition III.5 : Une fonction [SEM86].

C'est la réponse du système ou d'un composant à un stimulus interne ou externe.

Cette définition, si elle constitue un point d'entrée pour l'interprétation de la notion de fonction, n'en présente toutefois pas toute la sémantique. Nous allons la préciser par des exemples concrets.

Quels sont en pratique les fonctions d'un système ? Ce sont les services rendus par le système. Ainsi, la fonction d'un atelier flexible est de **produire** différents types de produits en quantités et délais précis ... Aussi, une définition intéressante est celle donnée par A. Dean.

Définition III.6 : Une fonction [DEA91].

C'est un service rendu par un système ou un composant lorsqu'il fonctionne normalement.

Qu'appelle-t-on un service? Prenons l'exemple d'un vérin. Dans un système de production flexible, un vérin peut être utilisé de différentes manières. Par exemple pour bloquer des pièces sur un poste de perçage. On dit alors qu'il a une fonction de blocage. Mais un vérin peut également être utilisé pour assurer le positionnement d'un aiguillage (fonction positionnement). Alors si l'on considère un vérin, quelle est sa fonction ? Ou bien peut-il avoir plusieurs fonctions ?

A notre avis, les fonctions blocage et positionnement ne sont pas des fonctions génériques d'un vérin. Ce sont les fonctions des systèmes (la machine de perçage et l'aiguillage) utilisant un vérin en tant que composant. Associer ces fonctions au vérin, c'est donc violer le **principe d'indépendance**. A notre avis, le vérin a une seule fonction générique qui est la fonction avance/recul. Cette fonction peut être utilisée pour réaliser les fonctions blocage ou positionnement. On pourrait également utiliser la fonction rotation d'un moteur (Fig. III.4). On établit alors un lien fonctionnel entre la fonction blocage (respectivement positionnement) et la fonction avance/recul ou rotation. Nous retrouvons ici le fait que dans tout système, il s'établit une hiérarchie fonctionnelle (notion de niveaux).

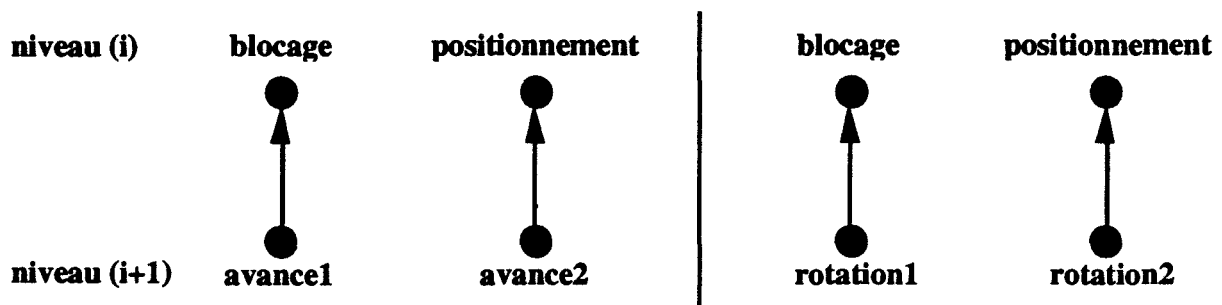


Figure III.4

Nous allons utiliser la représentation de la figure III.4 pour spécifier la structure de nos modèles fonctionnels. Dans cette représentation, les noeuds symbolisent les fonctions et les arcs orientés les liens de dépendance fonctionnelle. Ainsi, on dira que la fonction blocage de la machine de perçage utilise la fonction avance (avance1) d'un composant de type vérin.

Définition III.7 : Une fonction.

C'est la réponse d'un système ou d'un composant à un stimulus donné, lorsqu'il fonctionne normalement, indépendamment de l'environnement dans lequel il est sollicité.

Cette dernière définition rejoint la notion de fonction couramment utilisée en automatique continue. La fonction apparaît comme une boîte noire qui pour une entrée donnée (indépendante de l'environnement), donne toujours la même sortie. Nous préciserons cette idée lors du développement du concept de comportement.

I.3.2 Les différentes phases de la méthode.

Notre méthodologie concerne la construction du modèle fonctionnel d'un système existant dans le cadre du diagnostic. Elle s'applique donc aux systèmes dont le schéma fonctionnel global de conception n'est pas donné à priori. Elle comprend trois phases selon le schéma décrit figure III.5.

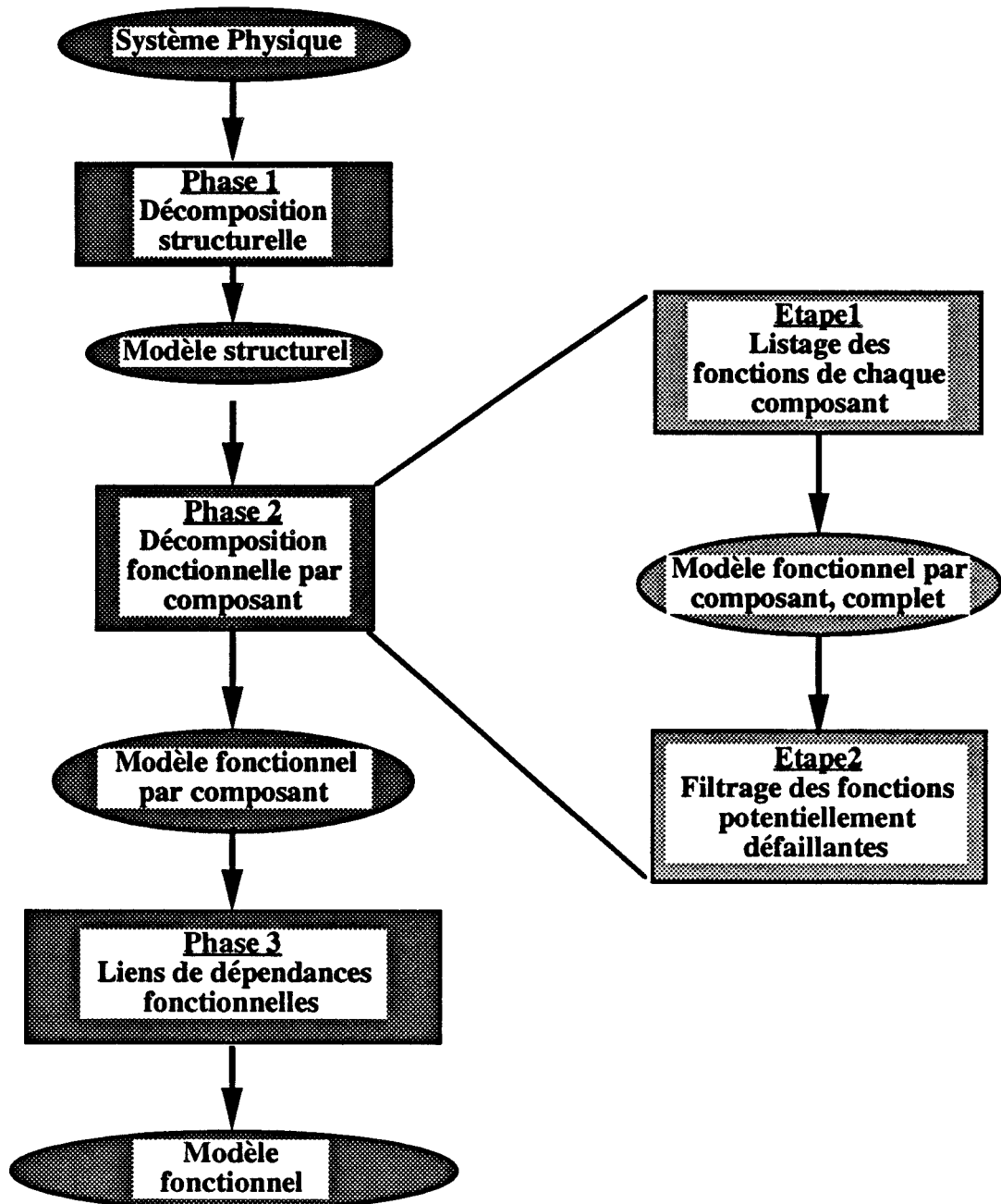


Figure III.5 : Méthodologie de conception du modèle fonctionnel d'un système existant.

Phase 1 : Décomposition structurelle.

Cette phase correspond à la décomposition du système en composant. On obtient alors un ensemble de composants de premier niveau, caractérisant le système. Puis chacun de ces composants est décomposé à son tour. Ce processus de décomposition est répété par niveau successif jusqu'à l'obtention des composants de base. Ils correspondent à deux types de composants :

- des composants potentiellement observables par des capteurs directs et dont les constituants ne peuvent l'être,
- des composants pour lesquels il n'y a pas lieu de poursuivre la décomposition en raison des objectifs de maintenance.

Au terme de cette phase, on obtient le modèle structurel de base dans lequel on aura modélisé les liens de dépendance structurelle.

Phase 2 : Décomposition fonctionnelle par composant.

Le but de cette phase est d'établir les fonctions utiles de chaque entité du système. Elle utilise donc les spécifications fonctionnelles de conception de ces entités. Elle comprend deux étapes :

Etape 1 : Elle consiste à établir la liste des fonctions de chaque composant. On obtient alors le modèle fonctionnel complet par composant.

Etape 2 : Elle consiste à éliminer systématiquement des listes précédentes, les fonctions qui ne sont pas susceptibles d'être défaillantes pendant la durée de vie moyenne du composant. Cette élimination est faite selon des critères de sûreté de fonctionnement. On calcule le taux de défaillance λ_f de chaque fonction à partir de la formule :

$$\lambda_f = \left(\prod_1^n \lambda_i \right)^{1/n} \quad (\text{III.1})$$

Les λ_i sont les taux de défaillance des constituants du composant auquel est rattaché la fonction ; constituants coopérants à sa création. Ces taux sont disponibles dans [RAC91].

Au terme de cette phase, on obtient un modèle fonctionnel par composant qui est un modèle réduit de celui obtenu à l'étape précédente. Cette réduction est nécessaire pour éviter d'augmenter inutilement la complexité du diagnostic.

Phase 3 : Modélisation des liens de dépendance fonctionnelle.

Cette phase permet d'établir les liens de dépendance fonctionnelle entre les fonctions de composants. La construction se fait de proche en proche, depuis les fonctions du système jusqu'à celles des composants de base. Le modèle fonctionnel par composant sert de support pour la conduite de la construction. Pour chaque fonction, le concepteur doit se poser deux questions :

- 1 - Quelles sont les fonctions permettant de produire cette fonction (fonctions filles),
- 2 - Quelles sont les fonctions utilisant cette fonction (fonctions mères).

Le résultat final est un graphe orienté dont les noeuds modélisent les fonctions, et les arcs les liens de dépendance. L'orientation permet d'indiquer le sens de la dépendance.

I.3.1 Exemple : modèle fonctionnel d'un centre d'usinage.

L'exemple retenu est la construction du modèle fonctionnel d'une machine outil à commande numérique. Il s'agit du CU60 de Graffenstaden. C'est un centre d'usinage quatre axes à broche horizontale. Les quatre axes correspondent au positionnement vertical de la broche (axe Y) et au positionnement de la table (axe X, axe Z et axe B). Tous les axes sont entraînés par des moteurs à courant continu. Les mouvements linéaires sont assurés par des vis sans fin. En outre, la broche a un mouvement de rotation par rapport à l'axe Z. L'actionneur est également un moteur à courant continu dont la rotation est transmise à la broche par un système de deux axes parallèles mis en relation par un ensemble baladeur-courroie. La courroie transmet le mouvement de l'axe primaire à l'axe secondaire et le baladeur permet de sélectionner une gamme de vitesse. Les pièces à usiner sont transférées sur la table par deux convoyeurs à palettes appelés échangeurs. Ces échangeurs sont actionnés par des vérins hydrauliques. Les palettes une fois sur la table sont bridées par deux vérins hydrauliques disposés en T sous la table. Selon le type de pièce chargé, un outil d'usinage est sélectionné dans le magasin. Ce magasin possède deux mouvements : un mouvement de positionnement angulaire (rotation incrémentale) afin de positionner l'outil dans le plan vertical au fourreau de la broche, et un mouvement de basculement vertical pour charger l'outil. Le positionnement angulaire est réalisé par un moteur à courant continu et le basculement par un vérin hydraulique. La ressource d'usinage de la machine est isolée de l'extérieur par un carénage comprenant un SAS (Système d'Accès et de Sécurité) permettant l'accès des pièces palettisées.

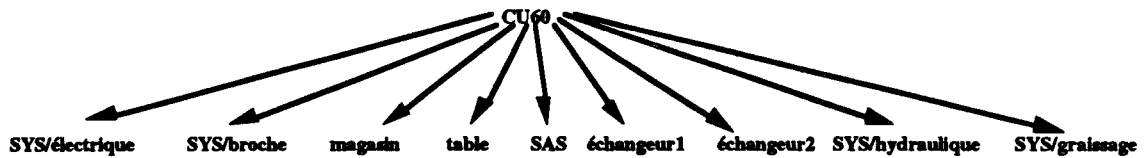


Figure III.6.

Dans un premier temps nous décomposons structurellement le CU60 en fonction de ses composants de premier niveau (Fig. III.6).

Dans le schéma précédent, on peut remarquer que dans cette décomposition, nous ne nous sommes pas limités aux seules entités mécaniques. Ainsi le noeud SYS/hydraulique représente le système hydraulique vu dans sa globalité au premier niveau.

Les arcs de ce graphe modélisent le lien "a-pour-composant". Ce lien a été choisi plutôt qu'un lien du type "est-composé-de" car il correspond au sens de la démarche de décomposition.

Nous répétons le processus à chacun des composants du premier niveau. Le SAS se décompose en une porte et un vérin (Fig. III.7). Le vérin peut être à son tour décomposé en piston et corps (corps-v/SAS).

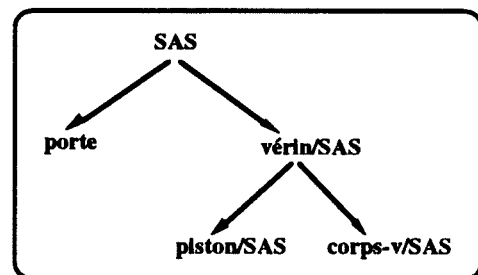
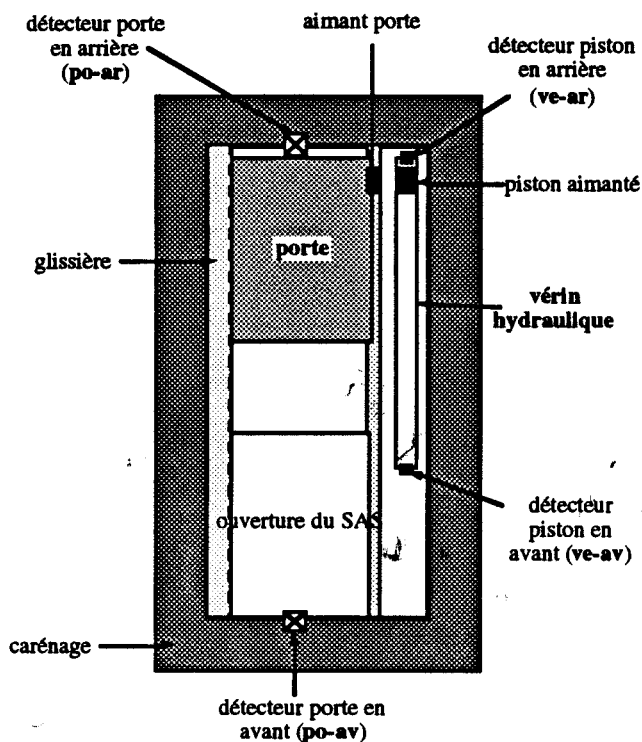


Figure III.7 : Schéma du SAS et modèle structurel associé.

On notera que les capteurs ne sont pas modélisés dans le modèle structurel.

La figure III.8 nous donne la décomposition obtenue pour les principaux organes mécaniques de la machine. Bien que le modèle réel soit plus complexe, nous nous limitons ici à ces organes, afin de conserver au graphe sa lisibilité.

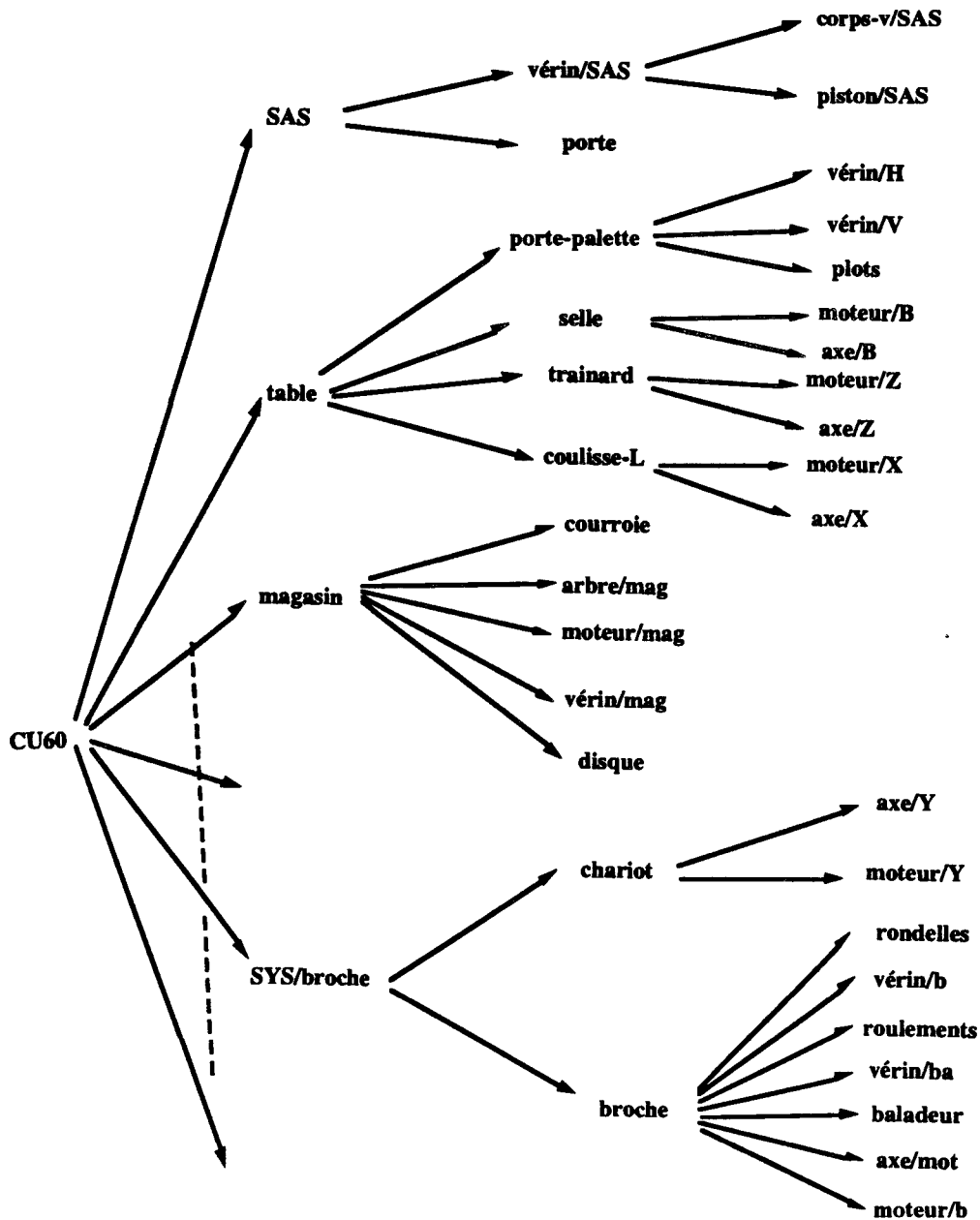


Figure III.8.

Pour l'ensemble des composants du modèle structurel, nous établissons la liste des fonctions utiles (Fig. III.9).

Toute opération d'usinage est caractérisée par deux objets : une pièce et un outil. Sur le CU60, la pièce à usiner est fixée sur la table. L'outil initialement stocké sur le disque du magasin, est emmanché dans la broche. L'usinage est obtenu par la conjonction des mouvements de la table et de la broche (donc de la pièce et de l'outil).

Nom	Niveaux structurels	Désignation	Fonction
CU60	0	Centre d'usinage	usinage
SAS	1	Système d'Accès et de Sécurité	accès, sécurité
Table	1	Table servant de support à la pièce usinée	stockage, positionnement et bridage de la pièce
SYS/broche	1	Système broche	stockage, positionnement et rotation de l'outil pendant l'usinage
Magasin	1	Magasin d'outil	stockage et positionnement de l'outil
Porte	2	Porte	avance
Vérin/SAS	2	Vérin du SAS	avance
Coulisse-L	2	Coulisse longitudinale	positionnement linéaire du trainard
Trainard	2	Coulisse transversale	positionnement linéaire de la selle
Selle	2	Selle	positionnement angulaire et bridage du porte-palette
Porte-palette	2	Porte-palette	Stockage pièce (palette)
Broche	2	Broche	stockage, rotation de l'outil
Chariot	2	Support broche	positionnement linéaire de la broche
Disque	2	Disque	stockage outils, basculement, positionnement circulaire
Vérin/mag	2	Vérin du magasin	avance
courroie/mag	3	Couuroie de transmission	Transmission
moteur/mag	3	Moteur	rotation
corps-v/SAS	3	Corps du vérin du SAS	glisse

Figure III.9.

A partir de la connaissance du comportement global de la machine, nous pouvons établir les fonctions de chacun de ses composants. Ainsi, le CU60 a une fonction d'usinage. Au niveau 1, le SAS a deux fonctions : une fonction d'accès et une fonction de sécurité. La porte du SAS a une fonction d'avance etc ... Toutes ces spécifications sont rassemblées dans le tableau de la figure III.9.

Pour construire le modèle fonctionnel, on est parfois obligé de créer des fonctions qui n'appartiennent pas à un composant donné. C'est par exemple le cas de la fonction

transmission/SAS. Cette fonction résulte de la force magnétique créée par le piston du vérin et l'aimant collé au bâti de la porte. Elle n'appartient donc ni à la porte ni au vérin. En fait, elle appartient à un composant virtuel dont les deux aimants seraient des constituants. Cela à un sens, dans la mesure où cette fonction transmission/SAS aurait pu être mise en œuvre par une autre entité matérielle bien réelle : par exemple une barre métallique.

En tenant compte de ces fonctions supplémentaires, on obtient le graphe de la figure III.10.

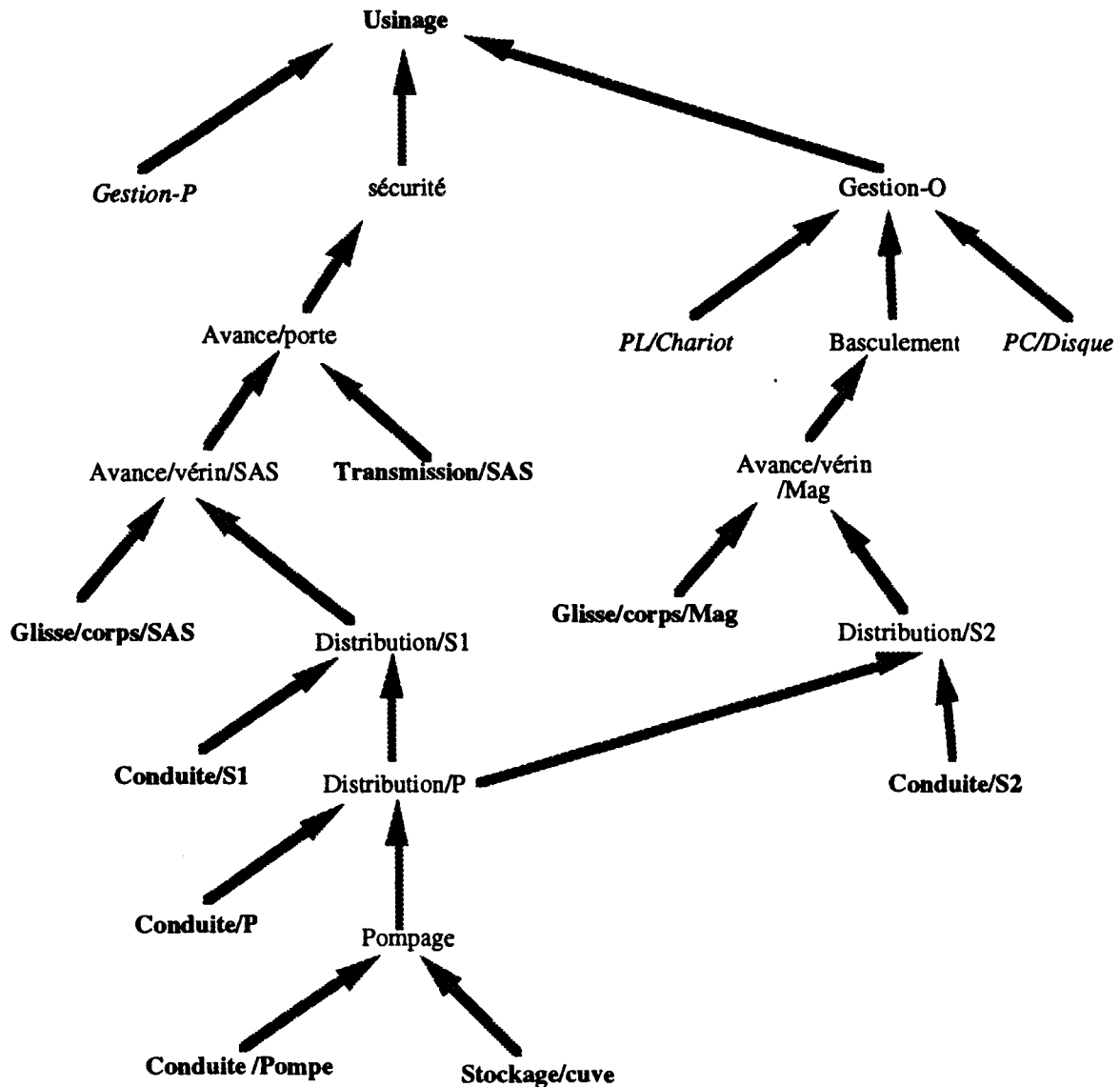


Figure III.10.

Sur la figure précédente, nous avons représenté en gras les fonctions initiales et en italiques les fonctions n'ayant pas été développées. Le terme PL (respectivement PC) désigne une fonction du type positionnement linéaire (respectivement positionnement circulaire).

I.3.4 Terminologie liée aux fonctions.

Définition III.8 : Fonction initiale.

On appelle fonction initiale toute fonction du graphe fonctionnel qui n'est dépendante d'aucune autre fonction.

Dans la figure III.10 , des fonctions comme **GLISSE/CORPS-SAS**, **CONDUITE/P** ou **STOCKAGE/CUVE** sont des fonctions initiales.

Définition III.9 : Fonction principale.

On appelle fonction principale toute fonction du graphe fonctionnel dont n'est dépendante aucune autre fonction.

Dans l'industrie manufacturière, tout processus, tout moyen de production à une fonction principale unique, accessible en tant que service par son environnement. Par exemple, la fonction principale du GF d'un centre d'usinage est la fonction **USINAGE**. De même la fonction principale d'un système de production est la fonction **PRODUCTION**.

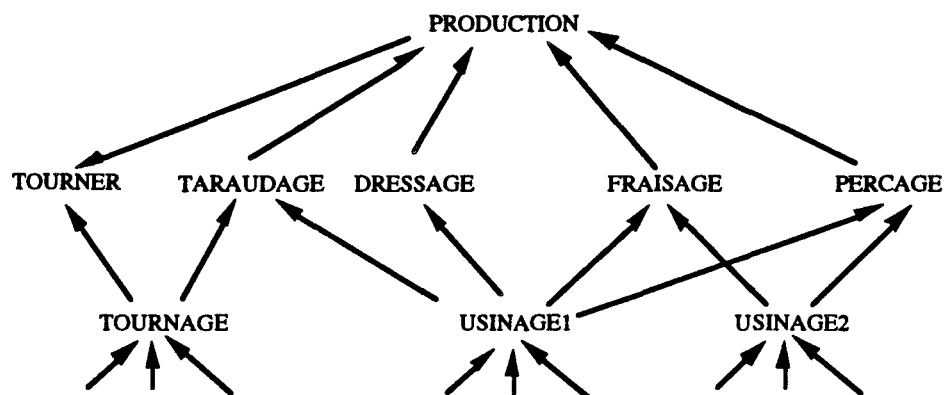


Figure III.11 : Extrait du GF d'un système de production.

I.4 Modélisation comportementale.

I.4.1 Introduction

Pour introduire la notion de comportement, nous empruntons à Chandrasekaran la définition suivante.

Définition III.10 : Un comportement [SEM86].

Il spécifie comment le système répond à un stimulus donné.

Pour être plus explicite, un comportement spécifie comment une fonction du système est mise en œuvre à partir des fonctions de ses composants. Cette spécification peut être causale ou temporelle. Ainsi si dans le modèle fonctionnel, nous nous sommes contentés de spécifier à partir de quelles fonctions filles, une fonction de niveau supérieur est créée, grâce à la modélisation comportementale, nous pouvons spécifier l'ordre d'appel. Considérons par exemple la fonction P5->P1/TRANSFERT (transfert de palettes du poste P5 vers le poste P1) du convoyeur présenté dans la figure II.4. Cette fonction de P5->P1/TRANSFERT fait appel aux fonctions butée1-BLOPAGE, tapis-AVANCE et aiguillage-POSITIONNEMENT (Fig. III.12). Avec les RdPI temporisés, nous pouvons spécifier le comportement de P5->P1/TRANSFERT de la manière suivante :

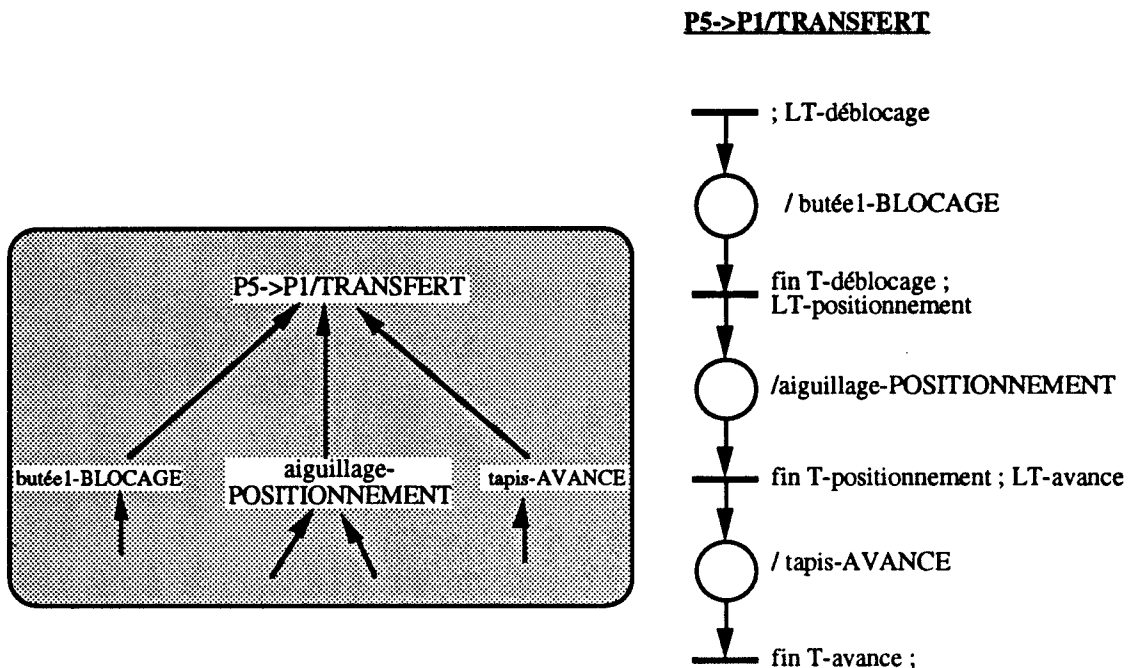


Figure III.12 : Modélisation du comportement d'une fonction de transfert.

Dans la pratique, nous ne pouvons pas toujours modéliser le comportement normal de manière discrète. En effet, plus on est bas dans le modèle fonctionnel, plus le comportement du système devient continu. On est alors obligé d'utiliser d'autres outils de modélisation. Par exemple la représentation par équations d'états ou des équations logiques. Nous avons déjà exposé les limites concernant l'utilisation de ce type de représentation dans le cadre du diagnostic. Par contre la panne d'un système correspond à un état discret. Cet état peut être lui même dissocié en sous-états correspondants aux pannes de sous-ensembles du système. Nous avons donc choisi dans le cadre du diagnostic de modéliser le comportement anormal du procédé.

I.4.2 Principe

Notre but est de construire un modèle de connaissance, qualitatif, modélisant le comportement défaillant du procédé. Compte tenu de notre système de détection, le synoptique du système de diagnostic est le suivant :

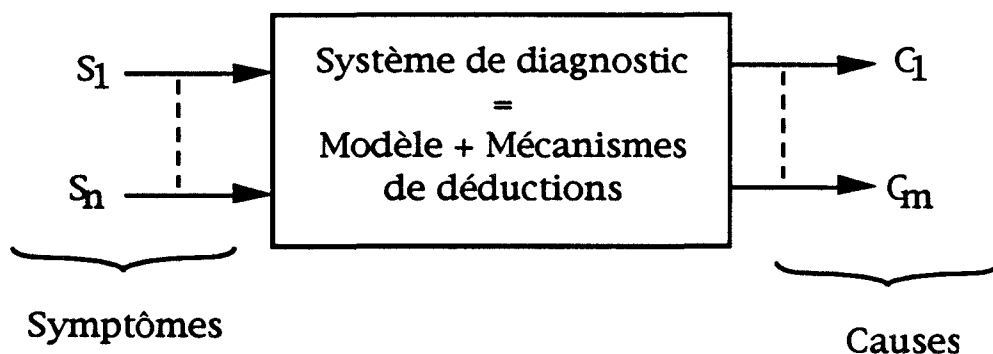


Figure III.13.

Ce synoptique montre clairement le rôle du modèle du procédé : établir des relations entre les entrées (symptômes issus de la détection) et les sorties (causes des symptômes donc les défaillances).

De nombreuses méthodes permettent de lier symptômes et causes de défaillances ; nous pouvons citer la MAC [VIL88] ou la méthode nommée **Action-Errors-Features Data Set** proposée par S.J. Chang et F. Diceseare [DIC89], [TOG90]. Mais toutes ces méthodes modélisent les comportements défaillants des systèmes sous une forme difficilement utilisable dans le cadre du diagnostic en ligne. Lors de l'expression des contraintes liées à cette forme de diagnostic, nous avons souligné l'incomplétude liée au raisonnement abductif sans apport de don-

nées supplémentaires. Plutôt que d'utiliser des capteurs supplémentaires nous avons pensé que le temps pouvait apporter ce supplément d'information. D'où l'introduction de la notion de **Signature Temporelle Causale (STC)**.

I.4.3 Le concept de la redondance temporelle [TOG91].

Pour introduire la notion de STC, nous avons défini le concept de **redondance temporelle**. Il est bâti sur un concept plus connu qui est la redondance analytique.

I.4.3.1 La redondance analytique [BRU90].

La redondance est un concept bien connu dans le cadre de la conception des systèmes automatisés. Il existe deux grands types de redondance : la **redondance matérielle** et la **redondance analytique**.

La redondance matérielle est la plus connue. Elle consiste à multiplier les entités matérielles vitales d'un processus. Par exemple, pour assurer l'entrée des pièces dans un système de production on pourra doubler le robot chargé de cette opération. Si les deux robots sont utilisés en parallèle, on dit qu'on a une redondance active. Si l'un des robots est laissé inactif dans le mode de fonctionnement normal, on a réalisé une redondance passive. De même, pour accroître la fiabilité du système de commande, on peut doubler voir tripler les capteurs à chaque point de mesure.

Quant à la redondance analytique, elle consiste en "**l'exploitation de la redondance informationnelle contenue implicitement dans un ensemble de mesures**" [BRU90]. L'utilisation de la redondance analytique nécessite l'exploitation d'un modèle permettant de relier ces mesures.

I.4.3.2 Enoncé du concept.

Reprenons l'exemple du convoyeur de la figure II.4. Supposons que nous voulons réaliser le transfert d'une palette de P5 vers P1. Supposons par ailleurs qu'un mauvais positionnement de l'aiguillage dirige la palette vers P3. Que va-t-il se passer au niveau des capteurs ? Au niveau de cp0, on ne détectera rien d'anormal. Par contre, les mesures de cp1, cp2, cp3 et cp4 ne seront pas conformes au comportement attendu du système. Ainsi cp2 et cp3 ne détecteront le passage d'aucune palette alors que cp1 et cp4 détecteront un passage. Ce comportement ne correspond pas à celui modélisé pour chaque capteur au niveau du module de détection (Fig.

III.24). De ce fait le module de détection va générer les symptômes S^2_{cp1} , S^1_{cp2} , S^2_{cp4} et S^1_{cp3} .

Les symptômes précédents apparaissent toujours répartis dans le temps, mais toujours dans le même ordre. Il existe donc des causes communes, les liants les uns aux autres. C'est ce que stipule le concept de la redondance temporelle.

Définition III.11 : La redondance temporelle [TOG91].

C'est l'exploitation d'informations temporelles comprises dans l'occurrence ou la non occurrence de signaux répartis dans le temps mais dépendant d'une cause commune.

I.4.4 Les Signatures Temporelles Causales (STC).

Dans ce paragraphe, nous allons formaliser l'écriture de STC. Pour cela, nous allons nous appuyer sur l'utilisation du modèle générique du comportement normal d'un capteur et de l'exploitation de structures temporelles classiques.

I.4.4.1 Introduction au concept de STC.

Dans l'exemple d'introduction au concept de la redondance temporelle, la cause commune est la défaillance de la fonction POSITIONNEMENT de l'aiguillage lors de l'opération P5->P1/TRANSFERT. Aussi, l'application de la redondance temporelle permet d'écrire la relation (III.2) :

$$(P5 \rightarrow P1/TRANSFERT) * S^2_{cp1} * S^1_{cp2} * S^2_{cp4} * S^1_{cp3} \Rightarrow (\text{def aiguillage-POSITIONNEMENT}) \text{ (III.2)}$$

ou plus généralement :

$$(P5 \rightarrow P1/TRANSFERT) * (ct1 S^2_{cp1}) * (ct2 S^1_{cp2}) * (ct3 S^2_{cp4}) * (ct4 S^1_{cp3}) \Rightarrow (\text{def aiguillage-POSITIONNEMENT}) \text{ (III.3)}$$

Dans (III.3) les "cti" modélisent des **contraintes temporelles** liant **deux à deux** les événements caractéristiques de la défaillance de la fonction aiguillage-POSITIONNEMENT. Il est important de noter ici que l'introduction des contraintes temporelles permet d'établir une équivalence entre cause et événements caractéristiques. C'est ce qui nous permet de changer le sens classique de l'implication.

L'opérateur * utilisé dans notre formalisme ne traduit pas une simple conjonction de fait. C'est un opérateur temporel qui traduit la relation de séquence "est-suivi-de".

Définition III.12 : Signature Temporelle Causale ou STC.

Nous appelons STC toute relation liant une séquence d'événements caractéristiques, à une ou plusieurs causes données.

I.4.4.2 Structure temporelle.

I.4.4.2.1 Choix d'une structure temporelle.

On distingue principalement deux types de structure pour la manipulation du temps :

- les structures de points,
- les structures d'intervalles.

Les structures de points sont les plus souvent utilisées. Elles permettent en effet une relation aisée avec les systèmes de nombre [BES89]. Par contre elles ne permettent pas de prendre en compte facilement, une notion comme la durée. La durée se modélise naturellement avec une structure d'intervalle.

Notre approche de détection des symptômes est basée sur la modélisation du comportement normal des capteurs à partir d'intervalles. Aussi avons-nous choisi d'avoir une approche basée sur une structure d'intervalles.

I.4.4.2.2 Définition des entités temporelles associées.

Une structure d'intervalle nous permet de définir trois types d'entités temporelles pour l'écriture des contraintes :

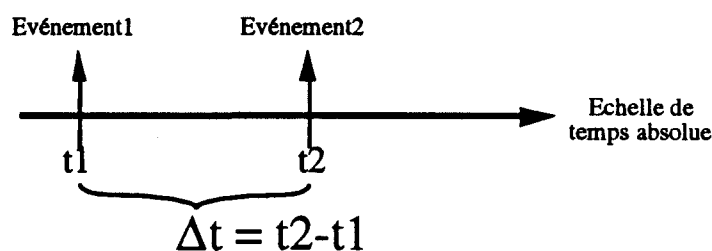


Figure III.14.

*** L'instant noté Δt .**

Il permet de modéliser de manière certaine, le temps relatif entre deux événements (Fig. III.14).

*** La période notée p .**

Elle permet de modéliser avec un degré d'incertitude le temps séparant l'occurrence de deux événements. Elle est représentée par un intervalle : la borne inférieure indiquant le temps minimal qui sépare les deux événements et la borne supérieure le temps maximal.

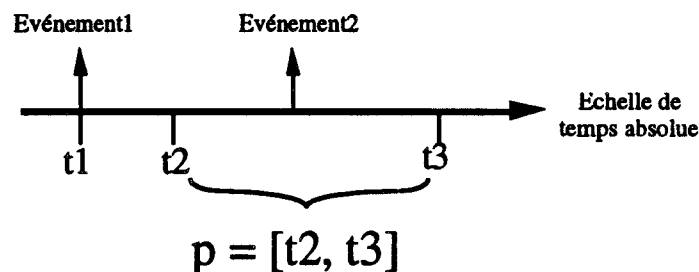


Figure III.15

La relation entre l'événement n°1 et l'événement n°2 est vraie si l'occurrence du deuxième événement est dans p .

*** La durée notée d .**

Elle est modélisée par un doublet $(\Delta t, v)$ constitué d'une date relative à l'événement précédent, et d'un intervalle. V est un réel qui indique le temps pendant lequel l'événement contraint doit rester vrai à partir de la date.

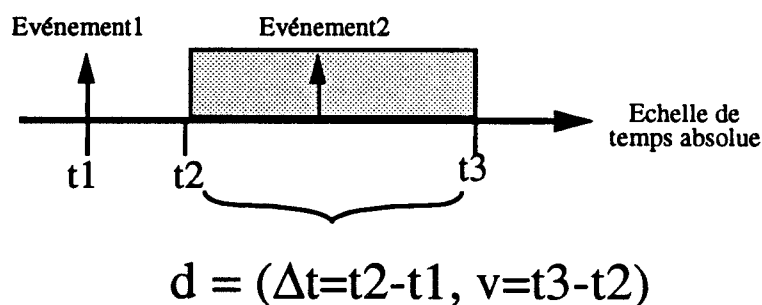


Figure III.16.

La relation entre l'événement n°1 et l'événement n°2 est vraie si ce dernier apparaît à Δt et persiste pendant v . La définition de " v " en tant que réel nous permettra d'utiliser des durées négatives.

I.4.4.3 Opérateurs et prédicats.

Pour écrire les STC, nous utilisons trois opérateurs.

Au niveau de la partie antérieure, les opérateurs sont :

- * l'opérateur de séquençement,
- $\neg$ l'opérateur de négation ; ainsi $\neg S_i$ signifie que le symptôme S_i n'a pas été détecté.

Au niveau postérieur, le seul opérateur utilisé est la conjonction de faits notée $\wedge$.

En plus de ces opérateurs, nous utilisons les prédicats suivants :

* Dans la partie antérieure :

occ qui spécifie la prise en compte d'un Start-Event à sa date d'occurrence.

* Dans la partie postérieure :

def qui qualifie l'état d'une fonction ou d'un capteur,

H1-def qui introduit une hypothèse sur l'état d'une fonction ou d'un capteur ; au paragraphe § II.2.3.1 nous introduirons un autre prédicat, **H2-def** servant à qualifier des hypothèses de défaillance avec une modalité différente, dans le cadre d'un raisonnement hypothétique.

I.4.4.4 Sémantique d'une STC.

Dans notre formalisme, les événements pris en compte lors de l'écriture d'une STC sont :

- l'événement de départ ou "Start Event",
- et les symptômes.

Un Start Event affirmé est pris en compte au début de la STC en utilisant le prédicat **occ**. Il modélise l'événement servant de référence temporelle à l'interprétation d'une séquence de symptômes en entrée du module de diagnostic. Il n'est donc pas contraint. Cet événement correspond en général à une opération émise par la Partie Commande (cf. Chapitre I).

Les symptômes sont contraints deux à deux en utilisant les entités temporelles définies précédemment. La syntaxe générale d'une STC est la suivante :

$$\text{STC}_i : (\text{occ Start-Event}_j) * (\Delta t \text{ S1}) * (p \text{ S2}) * (d \text{ S3}) \\ \implies (\text{def fonction}_k) \text{ (III.4)}$$

Dans l'écriture de la relation (III.4), nous avons fait abstraction de la nature des symptômes (type I ou type II, valeur affirmée ou niée). Dans le paragraphe suivant nous verrons comment le type de symptôme influe sur le type d'entité utilisée pour modéliser une contrainte temporelle.

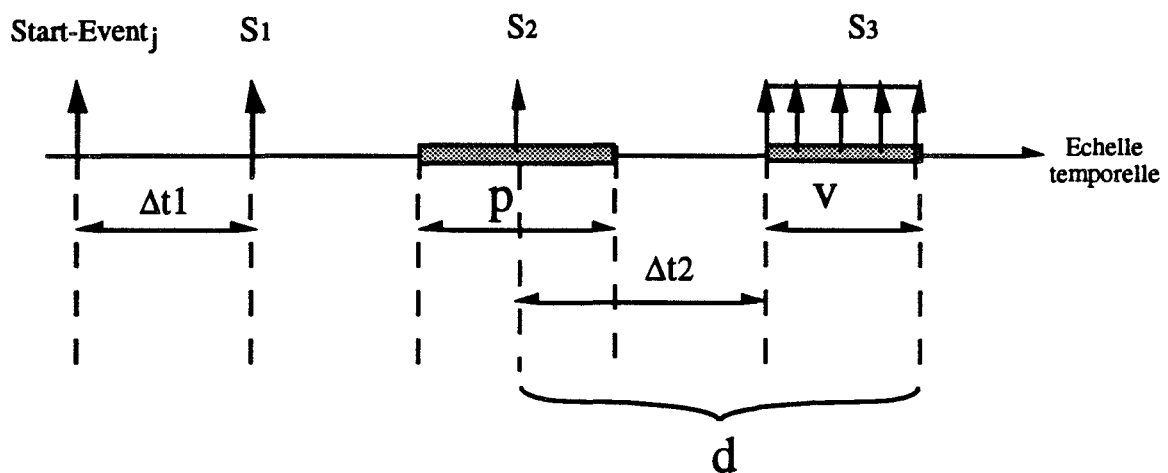


Figure III.17 : Interprétation des contraintes.

Il est important de noter que les STC ne sont pas des règles de production. En effet, la partie antérieure d'une STC sera vérifiée au fur et à mesure que les symptômes seront détectés et émis vers le module de diagnostic.

I.4.5 Ecritures des STC d'après les relations d'Allen.

I.4.5.1 Présentation des relations d'Allen [ALL81].[BES89].

Allen a formalisé 13 relations binaires permettant de spécifier les positions relatives de deux intervalles. En réalité nous avons 7 relations fondamentales, les autres étant des relations inverses de celles-ci (Excepté l'égalité dont la relation inverse est égale à elle même). Nous avons synthétisé ces relations dans le tableau suivant :

Relation	Relation Inverse	Représentation
I égale J (=)	égale (=)	
I avant J (<)	après (>)	
I touche J (m)	touché-par (mi)	
I recouvre J (o)	recouvert-par (oi)	
I débute J (s)	débuté-par (si)	
I contenu-dans J (d)	contient (di)	
I termine J (f)	terminé-par (fi)	

Figure III.18 : Relations d'Allen.

P.J. Hayes a montré que toutes ces relations peuvent être définies par la seule relation "touche" (m) [ALL89].

I.4.5.2 Expression des contraintes temporelles.

Afin d'exprimer les contraintes temporelles nous allons utiliser la relation "touche" appliquée à quatre configurations de symptômes.

Cas 1 : Deux symptômes de type I.

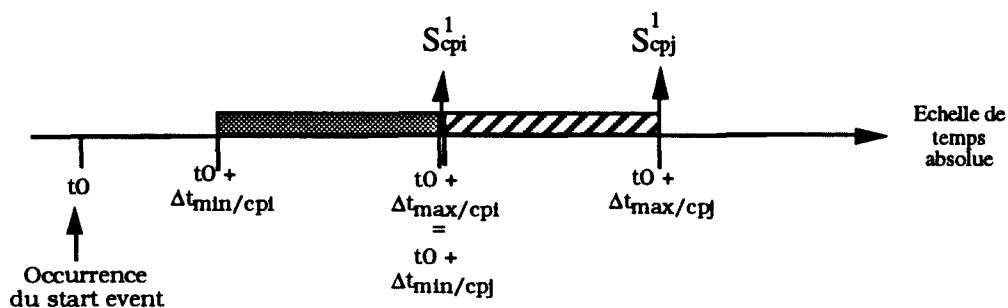


Figure III.19.

Les symptômes de type I apparaissent toujours à la borne supérieure des intervalles de validation I_{cpi} , la contrainte s'exprime simplement par un instant :

$$\dots S^1_{cpi} * (\Delta t = \Delta t_{max}/cpj - \Delta t_{max}/cpi) S^1_{cpj} \dots$$

Cas 2 : Un symptôme de type I suivi par un symptôme de type II.

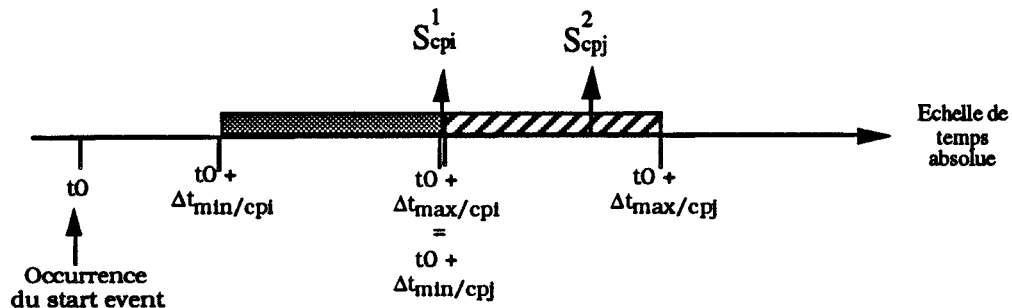


Figure III.20.

Le symptôme de type II peut apparaître n'importe quand dans l'intervalle $[t_0 + \Delta t_{max}/cpj, t_0 + \Delta t_{max}/cpi]$. De ce fait la contrainte sera exprimée par une période dont I est égal à la différence entre les bornes de cet intervalle et la date d'occurrence du symptôme de type I :

$$\dots S^1_{cpi} * (P = [0, \Delta t_{max}/cpj - \Delta t_{max}/cpi]) S^2_{cpj} \dots$$

Cas 3 : Deux symptômes de type II.

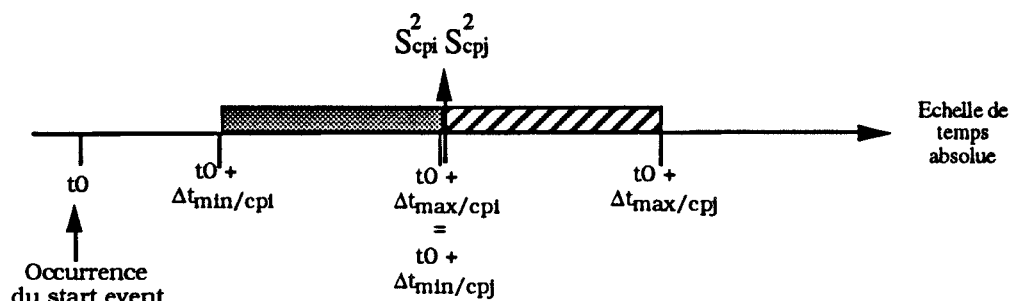
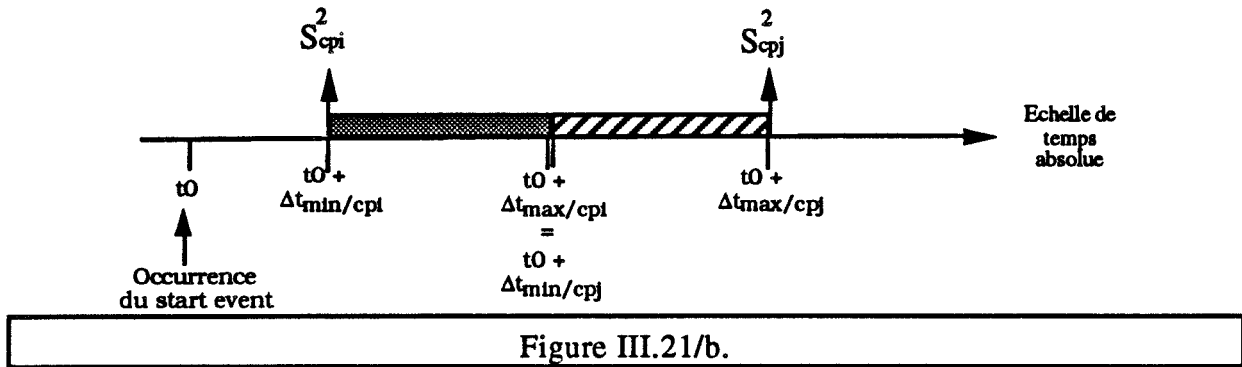


Figure III.21/a.



Lorsque deux symptômes de type II sont pris en compte cela revient à additionner leurs incertitudes. La contrainte sera donc de nouveau exprimée par une période déterminée à partir des configurations extrêmes.

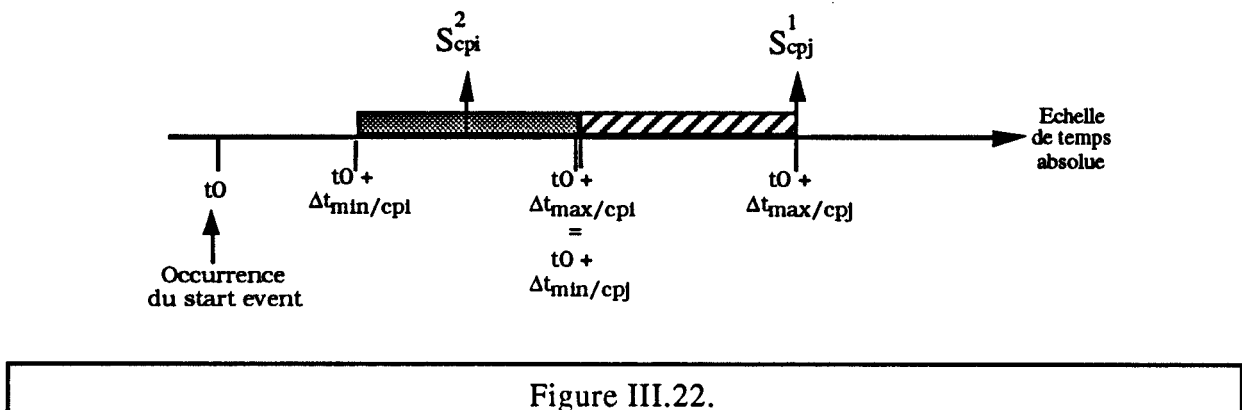
La première de ces configurations est donnée par la figure III.21/a. Les symptômes apparaissent à la borne supérieur du premier intervalle et à la borne inférieure du second. Comme les deux intervalles se touchent, l'expression est nulle.

La deuxième configuration extrême est donnée par la figure III.21/b. C'est le contraire de l'autre configuration. L'expression obtenue est $\Delta t_{\max}/cpj - \Delta t_{\min}/cpi$.

D'où l'expression finale de la contrainte :

$$\dots S^2_{cpi} * (P = [0, \Delta t_{\max}/cpj - \Delta t_{\min}/cpi]) S^2_{cpj} \dots$$

Cas 4 : Un symptôme de type II suivi par un symptôme de type I.



En appliquant une analyse similaire à celle du cas 2 on trouve que la contrainte s'exprime comme suit :

$$\dots S^1_{cpi} * (P = [\Delta t_{\max}/cpj - \Delta t_{\max}/cpi, \Delta t_{\max}/cpj - \Delta t_{\min}/cpi]) S^2_{cpj} \dots$$

I.4.5.3 Extensions et discussions.

Dans les quatre cas d'expressions des contraintes énumérés dans le paragraphe précédent nous n'avons en aucun cas utilisé l'entité durée. La raison en est très simple. En effet, dans chacun de ces cas, les occurrences des symptômes étaient affirmées. Or, il se trouve qu'on est parfois obligé de tester la non-occurrence d'un symptôme, pour apporter un supplément d'information à la discrimination des causes. Lorsque l'un des deux symptômes concernés est de type II, la non-occurrence doit être vérifiée sur tout l'intervalle d'occurrence potentielle. D'où l'introduction des durées.

Il est important de noter ici que nous nous plaçons dans l'hypothèse du monde clos. C'est à dire que tout symptôme qui n'a pas encore été détecté au cours d'une résolution est nié.

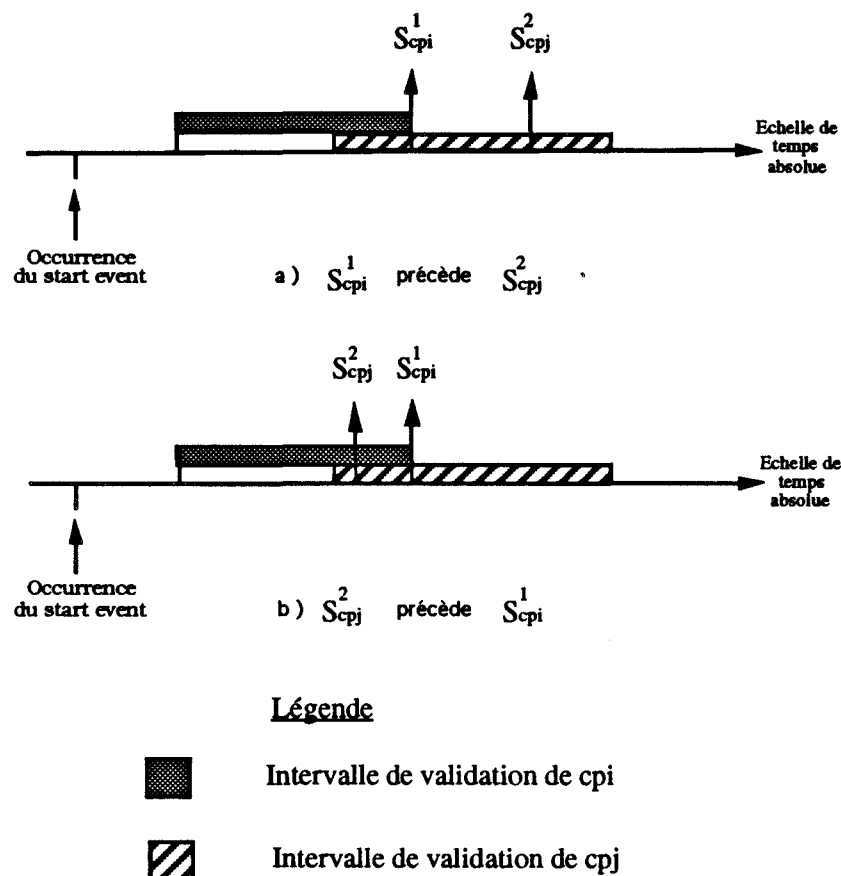


Figure III.23 : Occurrence d'un symptôme de type I à l'intérieur de l'intervalle de validation d'un symptôme de type II.

Le formalisme des STC est basé sur la notion de séquence, donc de précédence (et de succession). Or, lorsqu'on examine les relations d'Allen, on constate que seules les relations

"m" et "<" (et leurs relations inverses) vérifient la précédence entre intervalles. Quelle en est l'incidence sur le séquençement des symptômes ?

L'analyse de nos quatre cas de base, montre une difficulté à établir un ordre strict entre symptômes dans deux cas :

- lorsqu'on a deux symptômes de type II (cas 3),
- lorsqu'un symptôme de type I apparaît dans l'intervalle de validation d'un symptôme de type II (cas 2).

Dans ces cas, l'incertitude liée à l'occurrence d'un symptôme de type II rend impossible l'établissement d'un ordre strict. En effet, dans le cas 2, le symptôme de type II peut apparaître dans la zone de chevauchement (donc avant le symptôme de type I) ou après celle-ci (donc après le symptôme de type I). Pour prendre en compte cette incertitude, nous pourrions faire évoluer notre formalisme en utilisant une logique temporelle modale. C'est à dire introduire d'autres opérateurs que la précédence et la négation, traduisant des notions comme **il-est-possible-que**, **il-y-a-des-chances-que** ... Les expériences de mise en œuvre de logiques modales temporelles, nous ont dissuadé d'explorer cette voie. Cela d'autant plus que sur le plan pratique, la signification donnée aux intervalles de validation (cf. chapitre I), interdit le chevauchement entre deux intervalles de symptômes de type II. D'autre part, dans le cas n°2, le concepteur peut positionner différemment les capteurs afin d'éviter ce problème.

I.4.5.4 Exemple : Retour sur le convoyeur de la figure II.4.

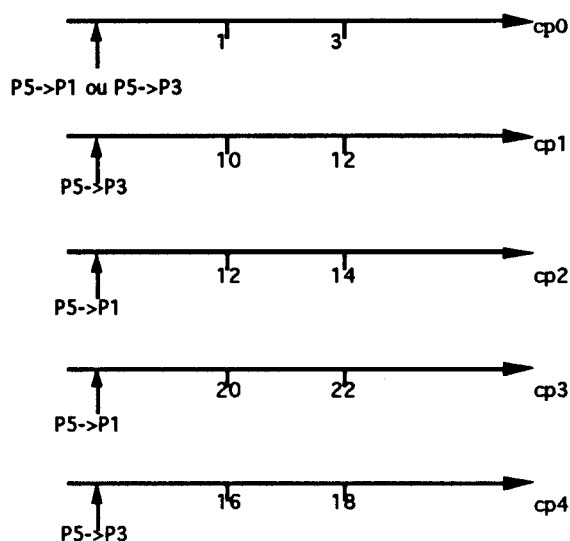


Figure III.24 : Modèle temporel utilisé par le module de détection.

A partir des spécifications de la figure III.24, nous pouvons établir STC1 en appliquant les cas 2 et 4 du § I.4.5.2 :

$$\text{STC1} : (\text{occ } P5 \rightarrow P1) * (p=[10,12] S^2_{cp1}) * (p=[2,4] S^1_{cp2}) * (p=[2,4] S^2_{cp4}) * (p=[2,4] S^2_{cp3}) \Rightarrow (\text{def aig-POSITION})$$

On peut écrire de même :

$$\text{STC2} : (\text{occ } P5 \rightarrow P3) * (\Delta t=12 S^1_{cp1}) * (p=[2,4] S^2_{cp2}) * (p=[2,6] S^1_{cp4}) * (p=[2,4] S^2_{cp3}) \Rightarrow (\text{def aig-POSITION})$$

STC2 est la règle duale de STC1.

$$\text{STC3} : (\text{occ } P5 \rightarrow P3) * (\Delta t=12 S^1_{cp1}) * (d=(2,2) \neg S^2_{cp2}) * (d=(2,4) \neg S^1_{cp4}) \Rightarrow (\text{H1-def tapis-AVANCE}) \wedge (\text{H1-def (collage cp1 0)})$$

Remarque : (H1-def (collage cp1 0)) signifie qu'il est possible que le capteur cp1 soit collé à zéro.

$$\text{STC4} : (\text{occ } P5 \rightarrow P3) * (\Delta t=3 S^1_{cp0}) * (\Delta t=9 \neg S^1_{cp1}) * (d=(0,2) \neg S^2_{cp2}) \Rightarrow (\text{def (collage cp0 0)})$$

$$\text{STC5} : (\text{occ } P5 \rightarrow P3) * (p=[10,12] S^2_{cp1}) * (d=(2,2) \neg S^1_{cp2}) * (d=(2,2) \neg S^2_{cp4}) \Rightarrow (\text{def (collage cp1 1)})$$

$$\text{STC6} : (\text{occ } P5 \rightarrow P3) * (\Delta t=3 S^1_{cp0}) * (\Delta t=9 S^1_{cp1}) * (d=(0,2) \neg S^2_{cp2}) \Rightarrow (\text{def butée1-BLOCAGE})$$

$$\text{STC7} : S^2_{cp0} * (d=(0,-3) \neg (\text{occ } P5 \rightarrow P3)) * (d=(10,2) \neg S^2_{cp1}) * (d=(0,2) \neg S^2_{cp2}) \Rightarrow (\text{def (collage cp0 1)})$$

STC1, STC2 et STC3 illustre l'utilisation des STC pour diagnostiquer des défaillances de fonctions filles. STC4 et STC5 permettent de coder la connaissance nécessaire au diagnostic

de défaillances de capteurs. STC3 illustre le fait que dans certains cas on ne peut que réduire l'ensemble d'hypothèses. Ici on passe de 8 défaillances potentielles (celles des 3 fonctions filles et des 5 capteurs) à 2. Quant à STC7, elle illustre le diagnostic du collage à 1 d'un capteur, en l'absence de toute commande ($\neg(\text{occ } P5 \rightarrow P3)$).

I.4.6 Reconnaissance de séquences dans une approche centralisée [TOG90], [FAH91].

Le principe est d'extraire d'une séquence d'entrée, les symptômes caractérisant des STC. Le mécanisme de résolution est basé sur la construction d'un arbre de résolution dont chaque sommet symbolise un **contexte de résolution ou monde**.

Contrairement au mécanisme de PROLOG, l'arbre est développé en largeur d'abord. Ce choix est basé sur l'hypothèse que le temps nécessaire au développement de nouveaux mondes à partir des feuilles de l'arbre à un instant donné, est inférieur au temps séparant l'occurrence de deux symptômes.

Au cours de la résolution, la division d'un monde en plusieurs fils correspond à émettre autant d'hypothèses qu'il y a de possibilité d'affectation d'un symptôme à des STC. Aussi, plus le nombre de symptômes communs à plusieurs STC est important, plus le risque d'explosion combinatoire est grand. Pour limiter cette explosion combinatoire, le contrôle est basé sur l'exploitation des contraintes temporelles. Dans chaque monde, on a plusieurs STC en cours de vérification : nous disons que ces STC sont lancées. Chaque **STC lancée**, est en attente d'un symptôme. Le symptôme attendu doit vérifier la contrainte temporelle le liant à son prédécesseur dans la STC considérée. Aussi son occurrence est contrôlée par un chien-de-garde déclenché dès la vérification du symptôme prédécesseur. La valeur du chien-de-garde est :

- Δt , si la contrainte est un instant,
- la borne supérieure si la contrainte est une période,
- $(\Delta t + v)$, si la contrainte est une durée.

Dès qu'un chien-de-garde expire, la branche du monde auquel il est associé est supprimée.

A un niveau i donné, le $j^{\text{ème}}$ monde $M_{i,j}$ est défini par les ensembles suivants :

- $R_{i,j}$: c'est l'ensemble des STC qui ne sont pas en cours de vérification,
- $L_{i,j}$: c'est l'ensemble des STC lancées, i.e. le complémentaire de $R_{i,j}$ par rapport à la base de règles,
- $V_{i,j}$: c'est l'ensemble des causes inférées.

$M_{0,0}$ est le contexte initial donc la racine de l'arbre. Il est défini par :

$R_{0,0}$ = base des STC , $L_{0,0} = \{ \}$ et $V_{0,0} = \{ \}$.

La résolution est terminée quand il reste un monde $M_{i,j}$ unique et que $L_{i,j} = \{ \}$.

Si $V_{i,j} = \{ \}$ alors il faut prévenir l'opérateur en lui donnant tous les symptômes de la séquence d'entrée. En effet cela signifie que la base de connaissance est incomplète.

Sinon, le diagnostic est donné par $V_{i,j}$.

Afin d'illustrer ce mécanisme, considérons la base de STC suivantes :

$$\text{STC1} : S_1 * (\Delta t_{1,1} S_2) * (\Delta t_{1,2} S_3) \Rightarrow C_1$$

$$\text{STC2} : S_2 * (\Delta t_{2,1} S_1) \Rightarrow C_2$$

$$\text{STC3} : S_3 * (\Delta t_{3,1} S_1) \Rightarrow C_3$$

avec la séquence suivante, d'entrée de symptômes dans le module de diagnostic :

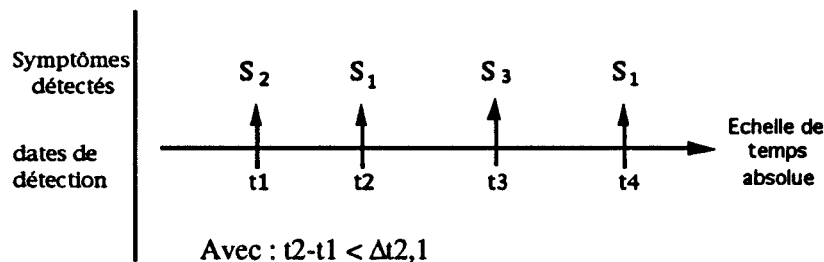


Figure III. 25 : Séquence d'entrée.

Cet exemple est identique à celui utilisé au chapitre II (cf. § II.1.3.2.2) pour illustrer les limites des systèmes experts classiques. La résolution s'identifie à l'arbre donné par la figure III.26, jusqu'à la date $t_3 + \Delta t_{2,1}$. Comme à cette date, l'ensemble des solutions candidates est réduit à $V_{4,1} = \{ C_2 \}$; il en résulte que la solution est C_2 . Néanmoins, il faut poursuivre la résolution car $L_{4,1}$ n'est pas vide. Deux cas sont alors possibles :

* A $t_3 + \Delta t_{3,1}$ on a expiration du chien de garde de STC3, alors on supprime la dernière branche de l'arbre et on avertit l'opérateur.

* A t_4 (avec $t_4 < t_3 + \Delta t_{3,1}$) on a de nouveau S_1 , alors la solution finale du diagnostic est $\{ C_2, C_3 \}$ (cas illustré par la figure III.26).

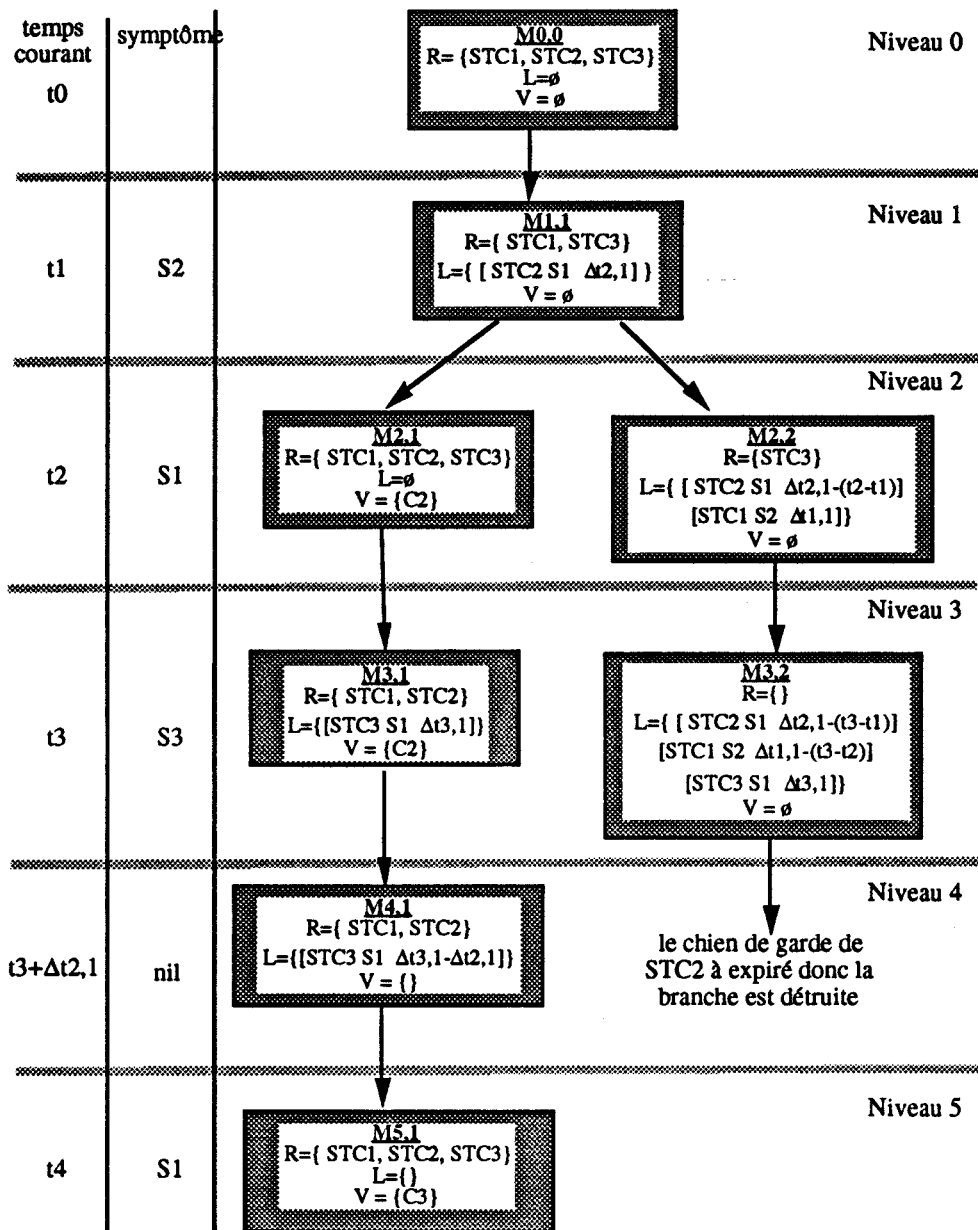


Figure III.26 : Reconnaissance de séquences.

I.4.7 Conclusion

Dans cette partie, nous avons présenté la notion de STC concept fondamental dans notre approche du diagnostic en ligne. En effet, les STC nous permettent de faire du raisonnement temporel condition nécessaire à la mise en œuvre de tout système temps réel basé sur des tech-

riques d'intelligence artificielle. Dans notre démarche nous avons essayé de rester pragmatique, afin de conserver au système son caractère temps réel. C'est la raison pour laquelle nous n'avons pas étendu notre système à l'utilisation de logiques modales temporelles. Le formalisme présenté ici est suffisamment riche pour résoudre les problèmes que nous voulons traiter.

Dans cette présentation nous avons présenté le mécanisme de résolution dans le cadre d'une approche centralisée. Historiquement, c'est le contexte dans lequel nous avons commencé le développement de ce formalisme. Dans ce cadre, le nombre de règles partageant les mêmes symptômes est plus grand. D'où l'utilité de limiter l'explosion combinatoire en coupant certaines branches de l'arbre de résolution.

Il faut néanmoins souligner que dans le contexte de l'approche globale de diagnostic, l'utilisation des STC ne concernera que la phase dite de "**diagnostic local**". Cette phase est locale à certains noeuds du modèle fonctionnel. De ce fait le nombre, de symptômes et de règles concernés font que bien souvent l'approche sera bien trop riche par rapport à la complexité du problème traité. D'autre part, le mécanisme de résolution sera mis en œuvre selon une approche orientée objet conforme aux orientations de l'approche globale.

II. EXPLOITATION DU MODELE FONCTIONNEL.

Dans la partie précédente, nous avons établi une distinction nette entre deux types de modèles : les modèles structurel et fonctionnel.

Cette partie, sera centrée sur l'exploitation du modèle fonctionnel selon deux axes :

* Le premier axe concerne le problème du choix et du positionnement des capteurs. C'est un problème que nous avons occulté dans la partie sur la détection car nous ne disposions pas des données nécessaires pour le résoudre. La question du positionnement des capteurs est essentielle dans le cadre du diagnostic en ligne. En effet, la qualité du diagnostic est directement liée à la qualité de l'observation que l'on a sur le procédé compte tenu du fait qu'il s'effectue automatiquement. Les points d'observations nécessaires et suffisants doivent avoir été prévu dès la conception.

* Le deuxième axe d'utilisation du modèle fonctionnel concerne la spécification de mécanismes de raisonnements adaptés à notre cadre de diagnostic. Bien entendu, ces mécanismes devront exploiter les caractéristiques du graphe fonctionnel.

II.1 Positionnement des capteurs.

II.1.1 Propriétés du graphe fonctionnel.

Nous proposons dans cette partie d'établir les propriétés du graphe fonctionnel. Dans un premier temps nous allons rappeler quelques concepts de base de la théorie des graphes [GON79].

Définition III.13 : Notion de graphe orienté.

Un graphe orienté $G=[X,U]$ est défini par la donnée :

- de l'ensemble X des noeuds du graphe,
- d'un ensemble U dont les éléments sont des couples ordonnés de noeuds appelés des arcs ($u \in U \iff \exists (i, j) \in X^2 / u = (i, j)$).

Les noeuds d'un graphe sont également appelés sommets.

Nous allons introduire les notions de chemins, de chaînes et de cycles.

Définition III.14 : Notion de P-graphe.

Un p -graphe est un graphe dans lequel il n'existe jamais plus de p arcs de la forme (i, j) entre deux noeuds i et j .

Propriété III.1 :

Le modèle fonctionnel d'une ressource de production est un 1-graphe simple.

Dans le modèle fonctionnel, les noeuds modélisent des fonctions, et les arcs des liens fonctionnels. Il y a au plus un lien de dépendance fonctionnelle entre deux fonctions. Le modèle fonctionnel est donc un 1—graphe.

Définition III.15 : Notion de chemin.

Dans un 1-graphe, un chemin est décrit par la succession des noeuds rencontrés.

Définition III.16 : Notion de chaîne.

Une chaîne de longueur q est une séquence de q arcs :

$$L = \{ u_1, u_2, \dots, u_q \}$$

telle que chaque arc u_r de la séquence ($2 \leq r \leq q-1$) ait une extrémité commune avec l'arc u_{r-1} ($u_{r-1} \neq u_r$) et l'autre extrémité commune avec l'arc u_{r+1} ($u_{r+1} \neq u_r$).

Remarque :

Contrairement à la notion de chemin, celle de chaîne fait abstraction de l'orientation des arcs. Les arcs sont dans ce cas identifiés à des arêtes.

Définition III.17: Notion de cycle.

Un cycle est une chaîne dont les extrémités coïncident.

Avec la notion de chaîne, nous pouvons introduire la connexité de la manière suivante :

Définition III.18 : Connexité.

Un graphe est dit connexe si pour tout couple de noeuds i et j , il existe une chaîne joignant i et j .

Propriété III.2

Le graphe fonctionnel de tout système électromécanique est un graphe connexe. On l'appellera GF.

Définition III.19

Un arbre est un graphe connexe sans cycles.

Cette définition entraîne qu'un graphe fonctionnel n'est pas forcément un arbre. En effet, dans l'exemple de GF donné par la figure III.10, nous pouvons extraire le cycle suivant :

{ USINAGE, ACCES-SECURITE, AVANCE/PORTE, AVANCE/VERIN/SAS, DISTRIBUTION/S1, DISTRIBUTION/P, DISTRIBUTION/S2, AVANCE/VERIN/MAG, BASCULEMENT, GESTION-OUTIL, USINAGE }.

II.1.2 Identification et observabilité.

A présent, nous allons introduire des définitions qui nous permettront de poser formellement le problème de l'identification des défaillances d'un système. Afin de ne pas entraîner de confusion par rapport au cas particulier des arbres, nous allons définir notre propre terminologie.

II.1.2.1 Terminologie liée à l'identification.

Définition III.20 : Noeud principal.

Le noeud principal p d'un GF correspond au noeud modélisant la fonction principale de GF.

Lorsque le GF est un arbre, le noeud principal correspond à sa racine.

Définition III.21 : Fonction successeur Γ .

La fonction successeur Γ est une fonction définie de X dans X telle que pour un noeud quelconque i de X différent de p , il existe j de X tel que $j = \Gamma(i)$. ($j = \Gamma(i) \Leftrightarrow \exists u \in U / u = (i, j)$)

Remarque :

$$\Gamma(p) = \emptyset$$

Définition III.22 : Noeud terminal.

On appellera noeud terminal, tout noeud $a \in X / \nexists i \in X$ avec $a = \Gamma(i)$.

Les noeuds terminaux sont rencontrés dans un GF, au terme d'un processus d'identification initié à un niveau inférieur. Ils modélisent les fonctions initiales.

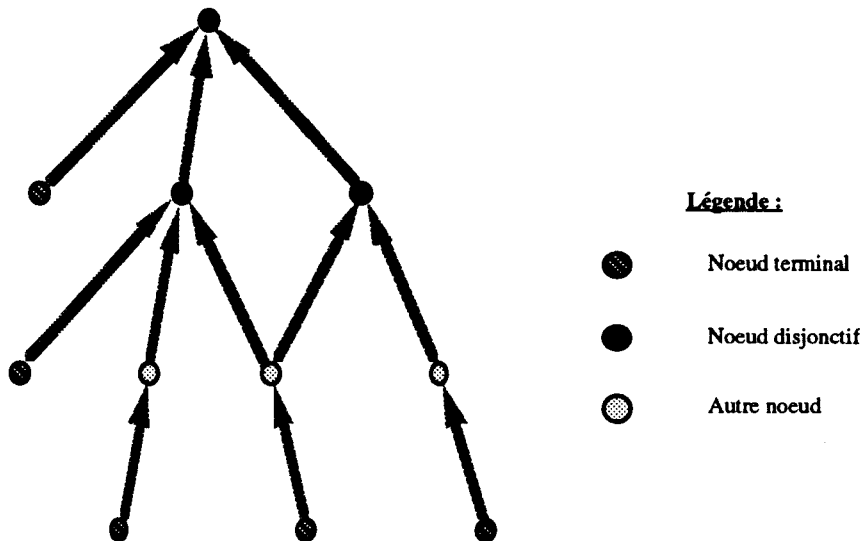


Figure III.27.

Dans un arbre, les noeuds terminaux correspondent aux feuilles de l'arbre.

II.1.2.2 Formalisation du problème de l'identification

En raison de la sémantique des arcs d'un graphe fonctionnel (liens de dépendance fonctionnelle), les noeuds terminaux modélisent les causes initiales des défaillances (défaillance première). Le processus d'identification à un instant donné, consiste donc à déterminer le sous-ensemble de noeuds terminaux permettant d'expliquer les symptômes détectés à cet instant.

Formellement on a $I(t) = \langle GF, TER, S(t) \rangle$, où :

GF est le graphe fonctionnel d'un système donné,

TER désigne l'ensemble des noeuds terminaux de GF,

S(t) représente l'ensemble des symptômes détectés à l'instant t ; ces symptômes sont affectés à des noeuds situés à des niveaux supérieurs aux noeuds terminaux.

Nous appelons TER^+ la solution du processus d'identification. TER^+ est le sous-ensemble minimal des sous-ensembles candidats comme solution de $I(t)$.

II.1.2.3 Lien entre identification et observabilité

Comme nous venons de le voir, le but de l'identification est de parcourir le graphe fonctionnel pour déterminer les noeuds terminaux responsables des défaillances. La perception de ces défaillances s'effectue par des noeuds situés à un niveau hiérarchiquement supérieurs à celui des noeuds terminaux. Cela peut sembler en contradiction avec la nécessité d'assumer la contrainte du temps-réel (cf chapitre II). En effet, le cas idéal serait que cette perception s'effectue directement au niveau des noeuds terminaux. Dans ce contexte, il n'y aurait pas d'identification. Le diagnostic serait limité à la phase d'analyse des conséquences, ce qui permettrait un confinement plus rapide des défaillances. Dans la pratique ce scénario idéal est impossible. En effet, la perception d'une défaillance au niveau d'une fonction est liée à la possibilité d'observation de son comportement : c'est ce que nous appelons l'**observabilité**. Dans le contexte de la surveillance en ligne, cette observabilité est acquise par le biais de capteurs. Les problèmes technologiques et les coûts induits ne permettent pas toujours de placer des capteurs pour surveiller les fonctions initiales (noeuds terminaux). Ainsi, il est généralement plus facile de surveiller les fonctions d'un système que celles de ses composants. Prenons l'exemple de la fonction roulement de tête de broche de notre centre d'usinage. En cas d'usure, le roulement émet un bruit. D'un point de vue technique, il est impossible de placer un capteur sur le roulement pour surveiller ce bruit. Par contre un capteur peut être placé sur le montant (bloc sur lequel repose la broche). Mais dans ce cas, le capteur va détecter également les bruits dus à



d'autres composants de la machine : usure de l'outil en cours d'usinage, chocs de l'outil sur la pièce, usure d'un des axes du système de rotation de la broche etc ... Dans ce contexte on surveille non pas la fonction roulement, ni même la fonction broche, mais plutôt la fonction usinage.

II.1.2.4 Définitions liées à l'observabilité.

Dans tout ce qui suit, nous associerons la propriété d'observabilité des fonctions aux noeuds qui les modélisent.

Définition III.23 : Observabilité d'un noeud.

Un noeud (une fonction) est observable si à tout instant, l'on peut établir son état.

Un noeud est "**directement observable**" si son **état défaillant** peut être établi par les symptômes qui lui sont liés.

Un noeud est "**indirectement observable**" si son état défaillant peut être indirectement déduit grâce à la causalité du graphe fonctionnel.

Un noeud est "**non observable**" s'il n'a ni la propriété de l'**observabilité directe** ni celle de l'**observabilité indirecte**.

Définition III.24 : Système totalement observable.

Un système sera dit totalement observable, si tous les noeuds du GF associé, sont directement ou indirectement observables.

Dans le cas contraire le système est partiellement observable.

Nous noterons **OBSD**, l'ensemble des noeuds directement observables d'un graphe fonctionnel. Les symptômes détectés à un instant donné permettent de caractériser un sous-ensemble de noeuds directement observables devant servir de support à l'identification. Nous appellerons **OBSD⁺** ce sous-ensemble.

Avec le concept d'observabilité, nous pouvons préciser la problématique de l'identification.

Proposition III.1 :

Si un système est totalement observable alors toute défaillance est identifiable.

Démonstration :

Cette propriété est directement déductible des définitions précédentes. En effet, si un système est totalement observable, tous ses noeuds sont observables. Il vient donc que ses noeuds terminaux sont observables. On peut donc à tout instant, établir TER^+ . Donc toute défaillance est identifiable.

II.1.2.5 Propriétés à vérifier du graphe fonctionnel dans le cadre de l'identification.

Quelles propriétés le graphe fonctionnel doit-il vérifier pour que tous ses noeuds soient observables ?

D'un point de vue pratique cette question pose le problème équivalent de la détermination du nombre minimal de capteurs nécessaires, et de leur positionnement sur le procédé.

Nous allons à présent définir et établir un certain nombre de propriétés qui nous permettront de donner une solution à ce problème.

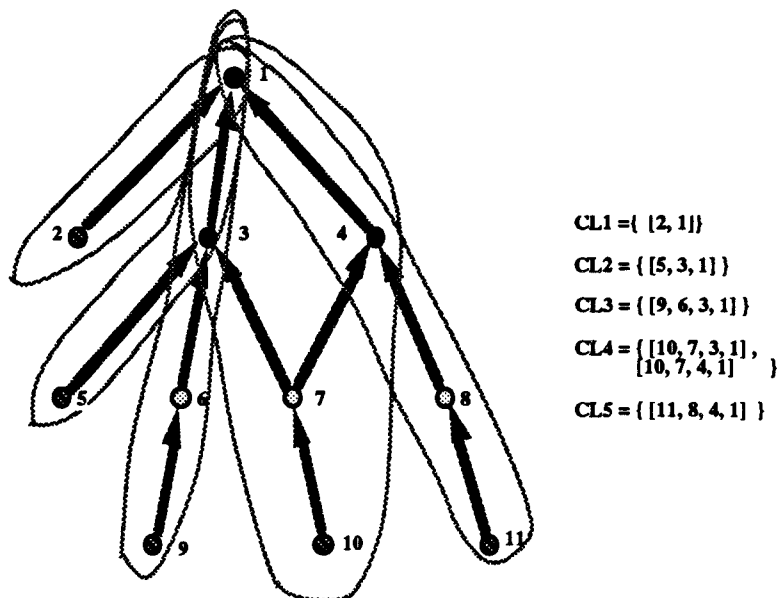


Figure III.28.

Toute défaillance apparaît au niveau d'un noeud terminal et se propage dans le GF jusqu'au noeud principal. Il y a un cheminement ascendant des défaillances à travers le graphe. Cela nous amène à introduire la définition de **chemin complet** suivante :

Définition III.24 : Chemin complet C_i .

C'est un chemin du graphe fonctionnel vérifiant les conditions suivantes :

- il a pour extrémité initiale, un noeud terminal,
- il a pour extrémité terminale, le noeud principal.

Sur la figure III.28, les ensembles [2, 1], [5, 3, 1] ... sont des chemins complets. Par contre [9, 6, 3] n'est pas un chemin complet car le noeud principal "1", n'est pas élément de cet ensemble.

Définition III.25: Noeud disjonctif.

C'est un noeud par lequel passe au moins deux chemins complets (cf. figure III.28).

Soit $\mathcal{C}$, l'ensemble des chemins complets du graphe fonctionnel. Nous définissons sur $\mathcal{C}$ la relation $\mathcal{R}$ suivante :

Définition III.26 : Relation entre chemins complets.

C_i est en relation $\mathcal{R}$ avec C_j s'ils ont la même extrémité initiale (ini) .

$$C_i \mathcal{R} C_j \Leftrightarrow \begin{cases} C_i = C_j \\ \text{ou} \\ \text{ini}(C_i) = \text{ini}(C_j) \end{cases}$$

Propriété III.3 :

$\mathcal{R}$ définit une relation d'équivalence sur $\mathcal{C}$.

La propriété précédente est directe car $\mathcal{R}$ est uniquement définie à partir de l'égalité.

La relation $\mathcal{R}$ induit donc une partition de $\mathcal{C}$ en classes d'équivalence notée CL_i . Dans la figure III.28, nous dénombrons quatre classes d'équivalence. Par exemple, la classe CL_4 comprend les chemins [10, 7, 3, 1] et [10, 7, 4, 1] .

Propriété III.4 :

Il y a autant de classes dans $\mathcal{C}$ qu'il y a de noeuds terminaux .

La propriété III.4 est une conséquence directe de la définition III.26 . En effet, chaque classe d'équivalence est caractérisée par un noeud terminal ($\text{ini}(C_i) = \text{ini}(C_j)$).

Du point de vue de l'observabilité, on peut définir une relation d'équivalence entre noeuds d'une même classe de chemins complets.

Propriété III.5 :

Si un noeud terminal est observable (directement ou indirectement) alors tous les noeuds de la classe auquel il appartient sont observables.

La propriété III.5 résulte de la causalité du graphe fonctionnel.

Par contre la réciproque est fausse. Le contre-exemple est le cas d'un noeud disjonctif observable appartenant à deux classes distinctes.

Proposition III.2 :

S'il existe un noeud directement observable, **distinct** par classe de chemins complets, alors le système est totalement observable.

Remarque :

Le qualificatif "distinct" signifie ici que le noeud appartient à une et une seule classe de chemins complets.

Démonstration :

Nous allons démontrer la contraposée. C'est à dire que si deux classes ont un et un seul noeud directement observable, qui leur est commun, alors les noeuds de ces deux classes ne sont pas observables.

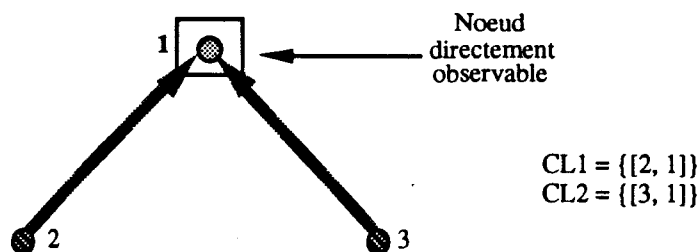


Figure III.29.

Dans l'exemple de la figure précédente, le noeud 1 est un noeud directement observable. Nous supposons donc qu'à tout instant, nous connaissons son état. Si cet état est défaillant, que pouvons-nous déduire sur l'état respectif des noeuds 2 et 3 ? La causalité du graphe ne nous permet aucune déduction si nous sommes limités à ce degré d'observabilité.

Proposition III.3 :

Deux classes de chemins complets sont observables s'il existe au moins un noeud directement observable par classe et si l'un de ces noeuds est un noeud distinct.

Démonstration :

La démonstration de cette proposition est évidente. Considérons deux classes C_i et C_k , telles que "i" soit un noeud directement observable appartenant en propre à C_i , et que "k" soit un noeud disjonctif, directement observable, partagé par les deux classes.

D'un point de vue statique, il suffit d'affecter "k" à C_k , et on retombe dans le cadre de la Proposition III.2.

D'un point de vue dynamique, si "k" localise une défaillance qui n'est pas perçue par "i", c'est que le noeud terminal caractérisant C_k est défaillant.

Proposition III.4 :

Un système est observable si ses classes de chemins complets sont observables deux à deux.

La Proposition III.4 découle de la Proposition III.3. Son intérêt est qu'elle permet de choisir comme noeud directement observable, certains noeuds disjonctifs.

II.1.3 Algorithme de positionnement n°1.

L'algorithme de positionnement des capteurs repose sur les propositions (Proposition III.2 et Proposition III.4) définies dans le paragraphe précédent. Pour l'énoncer, nous définissons les notations suivantes :

$C[i]$ est la $i^{\text{ème}}$ classe de C .

$C[i][j]$ est le $j^{\text{ème}}$ chemin complet de $C[i]$.

$C[i][j][k]$ est le $k^{\text{ème}}$ noeud de $C[i][j]$.

$OBS[i]$ désigne le noeud directement observable de $C[i]$.

m est le nombre de chemins complets de $C[i]$.

q le nombre de noeuds de $C[i][j]$.

En plus de ces variables, nous définissons les fonctions suivantes :

initiale : donne le noeud terminal d'une classe,

noeud-disjonctif : indique si un noeud est de type disjonctif ou pas,

successeur : donne le successeur du noeud courant sur le chemin complet parcouru.

L'algorithme s'énonce de la manière suivante :

begin

(1) Construction des n classes de C

(2) (* Initialisation *)

$i \leftarrow 1$

$stop \leftarrow false$

(3) While ($i \leq n$) and (not stop) do

begin

(3.1) (* Initialisation *)

$OBS[i] \leftarrow O$; (* OBS est le vecteur des noeuds directement observable du graphe*)

$j \leftarrow 1$; (* indice d'un chemin complet dans une classe *)

$noeud-référence \leftarrow initiale(C[i])$

(3.2) While ($OBS[i] = 0$) and ($j \leq m$) do

begin

(3.2.1) $noeud-courant \leftarrow noeud-référence$

$k \leftarrow 1$; (* indice d'un noeud sur un chemin donné *)

(3.2.2) While (not $d-observable(OBS[i])$) and ($k \leq q$) do

begin

(3.2.2.1) **If** *noeud-disjonctif* ($noeud-courant$)

then $noeud-référence \leftarrow noeud-courant$

(3.2.2.2) $k \leftarrow k + 1$

(3.2.2.3) $noeud-courant \leftarrow successeur(C[i][j])$


```

        end
    (3.2.3) If k < q+1 then OBS [i] <- noeud-courant else j <- j+1
        end
    (3.3) If OBS[i]=0 then stop <- true else i <- i+1
        end
end

```

II.2 Mécanismes de raisonnement.

II.2.1 Principe de résolution.

Notre approche du diagnostic repose ainsi sur un principe très simple : étant donné l'ensemble des symptômes détectés à un instant donné, comment en déduire le maximum d'information sur l'état du système (d'un point de vue fonctionnel). Le processus est répété à travers le modèle fonctionnel jusqu'à l'identification d'une ou plusieurs fonctions initiales (noeuds terminaux). Le diagnostic peut être vu à deux niveaux :

- * Un niveau local à chaque noeud du modèle. Nous dirons alors que le diagnostic est local. Le **diagnostic local** consiste à déduire l'état du noeud à partir des symptômes qu'il reçoit. Nous verrons que cette forme de diagnostic ne concerne qu'une catégorie des noeuds du modèle.
- * Un niveau global au graphe fonctionnel. Nous l'appellerons **diagnostic global**. Cette forme de diagnostic est basée sur la coopération entre noeuds du graphe.

II.2.2 Observabilité totale et partielle.

Dans le paragraphe II.1.2, nous avons introduit le concept d'observabilité qui est lié à la présence de capteurs sur le système. Implicitement, nous avons supposé que les informations données par les capteurs étaient saines. Cette hypothèse simplificatrice avait pour but de trouver une condition suffisante pour réaliser le positionnement des capteurs. Est-elle réaliste ?

Tout dépend de la méthode utilisée pour isoler les défaillances de capteurs. Nous en distinguons deux :

* La première méthode consiste à isoler les défaillances de capteurs avant la phase de diagnostic. Le principe est d'insérer un module de localisation entre la détection et la localisation (cf. première partie). Le but de ce module est l'attribution des symptômes à des défaillances capteurs ou (exclusif) à des défaillances d'autres entités du procédé.

* La deuxième méthode consiste à traiter les défaillances de capteurs au niveau même du diagnostic comme toutes défaillances du procédé. C'est la méthode que nous avons retenue (cf. § I.4 et la notion de STC) . Ce choix s'explique par le fait que globalement nous gagnons au niveau du temps d'identification. Dans ce cadre d'hypothèses, nous respectons donc la contrainte du temps réel.

La prise en compte des défaillances de capteurs au niveau du diagnostic, amène à distinguer deux types d'observabilité :

- l'observabilité totale,
- l'observabilité partielle.

II.2.2.1 L'observabilité totale.

C'est ce concept que nous avons implicitement utilisé pour la génération de l'algorithme de positionnement n°1. Il pose que le diagnostic local réalisé par un noeud directement observable est déterministe. C'est à dire qu'à partir de la connaissance des symptômes reçus, le mécanisme interne du noeud va établir avec certitude, soit la défaillance de la fonction modélisée, soit celle d'au moins un de ses capteurs.

II.2.2.2 L'observabilité partielle.

Ce concept est plus nuancé que le précédent. Le diagnostic local peut se contenter de conclure sur une hypothèse de défaillance de la fonction modélisée. En utilisant un mécanisme que nous appellerons **validation**, le diagnostic global se chargera de confirmer ou rejeter cette hypothèse.

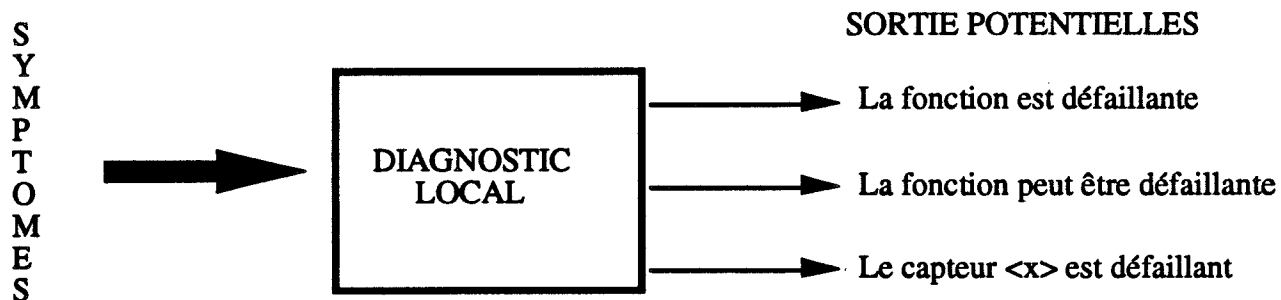


Figure III.30 : Diagnostic local lié au concept observabilité partielle.

Dans notre approche, nous utilisons l'observabilité partielle.

II.2.2.3 Algorithme de positionnement n°2.

Avec l'observabilité partielle, l'algorithme de positionnement n°1 n'est plus valable. Nous allons introduire ici une nouvelle proposition qui permettra de le modifier.

Le nouvel algorithme est basé sur une application verticale du concept de la redondance analytique.

L'idée est de vérifier la cohérence de signaux délivrés par des capteurs liés à des noeuds appartenant à une même classe d'équivalence. Si les signaux sont cohérents, c'est qu'ils sont valides. Prenons l'exemple de du SAS du CU60. La fonction avance/porte est observée grâce aux capteurs po-ar et po-av. De même la fonction vérin-AVANCE est surveillée par les capteurs ve-ar et ve-av (Fig. III.7). Supposons que le SAS soit ouvert (porte et vérin en position arrière) et qu'on lui commande de se fermer. A ce moment po-av devrait être égal à "1" ou bout d'un certain temps (po-av=1 signifie "porte en position fermée"). Si ce n'est pas le cas on ne savoir avec po-av et po-ar si la défaillance est due au capteur (collage à 0) ou si elle est due à la fonction. Par contre la connaissance de ve-av, nous permettra de conclure. En effet si ve-av=1, cela signifie que le capteur po-av est défaillant. Si ve-av = 0, alors c'est la fonction Avance/porte qui est défailante. D'où la proposition :

Proposition III.5 :

Pour que toute défaillance soit identifiable, il faut et il suffit que deux noeuds soient directement observables par classe de chemins complets.

Remarque :

Il doit y avoir au moins un noeud directement observable distinct par classe de chemins complets. Dans la pratique, on cherchera à positionner le second noeud au niveau des noeuds disjonctifs.

II.2.3 Gestion d'hypothèses.

Le concept d'observabilité partielle introduit également, la nécessité de gestion (ou discrimination) des hypothèses. L'idée ici est d'utiliser le graphe fonctionnel comme réseau d'hypothèses.

II.2.3.1 Nature des hypothèses générées.

Nous distinguons deux types d'hypothèses :

*** Les hypothèses d'identification ou H1-def.**

Elles sont gérées par le mécanisme d'identification. Elles correspondent à l'un des contextes suivants :

- le diagnostic local à un noeud qui a établi avec certitude la défaillance de la fonction modélisée. Il faut alors déterminer lesquelles de ses fonctions filles en sont responsables. Le processus d'identification génère alors autant d'hypothèses **H1-def** qu'il y a de fonctions filles soupçonnables.
- le diagnostic local réduit les défaillances possibles à quelques fonctions filles et certains capteurs (cf. § I.4).

Exemple : Contexte n°1.

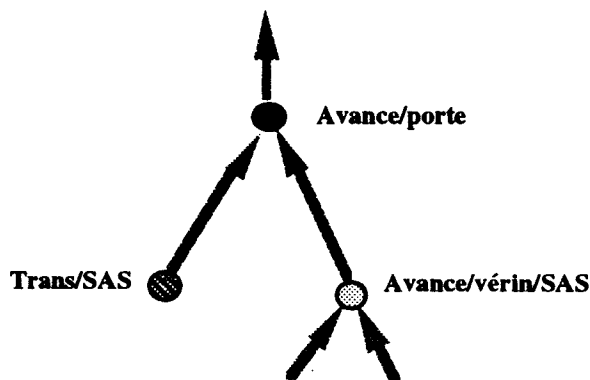


Figure III.31 : Extrait du graphe fonctionnel du SAS du CU60.

La fonction avance de la porte (Avance/porte) a deux fonctions filles : les fonctions Trans/SAS et Avance/vérin/SAS. Si le diagnostic local établi avec certitude que Avance/porte est défaillante, alors dans le cadre de l'identification, on génère :

- l'hypothèse de défaillance de Trans/SAS (**H1-def Trans/SAS**),
- l'hypothèse de défaillance d'Avance/vérin/SAS (**H1-def Avance/vérin/SAS**).

Comme au § I.4, H1-def est un prédicat utilisé pour typer les défaillances. H1-def introduit une modalité du type "**il-y-a-une-forte-probabilité-de-défaillance**". Au moins une des hypothèses générées sera vérifiée. Ainsi, dans l'exemple précédent, on est sûr que l'une des deux fonctions, Tans/SAS ou Avance/vérin/SAS, est défaillante.

*** Les hypothèses de validation ou H2-def.**

Ce type d'hypothèse est généré dans le cadre de la **validation** (cf. § II.2.3.5) de l'hypothèse de défaillance d'une fonction. Elle correspond à une déduction locale incertaine ou à la propagation d'une hypothèse de type H1-def à partir d'un noeud disjonctif. En reprenant l'exemple précédent, l'hypothèse (H1-def Avance/porte) issue du diagnostic local au noeud Avance/porte conduirait à générer les hypothèses (H2-def Avance/vérin) et (H2-def Trans/SAS).

H2-def introduit une modalité du type "**il-est-possible-que**" la fonction soit défaillante. Cette modalité est moins forte que la première dans la mesure où l'ensemble des hypothèses générées peuvent être rejetées..

II.2.3.2 Stratégies d'analyses.

Ces stratégies spécifient comment peuvent être gérées des hypothèses à travers un réseau qui ici est le graphe fonctionnel. Nous en distinguons quatre :

*** Stratégie d'analyses exclusives [FRO90].**

Cette stratégie se situe dans le cadre du principe de défaillance unique. Elle considère l'ensemble des hypothèses issues d'un même noeud père comme des hypothèses exclusives. De ce fait, dès qu'une hypothèse est confirmée toutes les autres sont abandonnées.

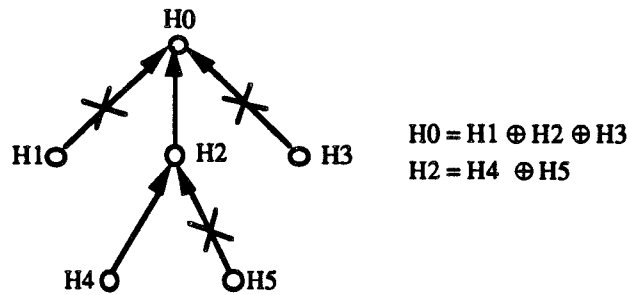


Figure III.32.

Dans l'exemple de la figure III.32, nous supposons que H4 est vraie. Alors H2 est vraie et on arrête la vérification de H5. De la même manière H2 est vraie, entraîne que H0 est vraie et on arrête la vérification de H1 et H3.

*** Stratégie d'analyses complémentaires [FRO90].**

Dans cette stratégie, toutes les voies d'investigation issues d'une même hypothèse, concourent à sa validation. Si l'une des voies n'aboutit pas l'hypothèse est rejetée (Fig. III.33).

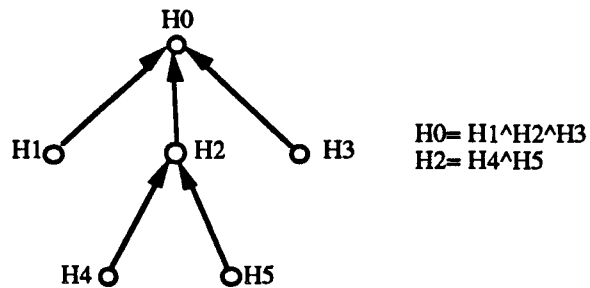


Figure III.33.

*** Stratégie d'analyses concurrentes.**

Dans cette stratégie, toutes les hypothèses ont la même vraisemblance. L'analyse se fait donc en parallèle sur toutes les voies. Elle intègre le principe de défaillance multiple La validation d'une hypothèse n'entraîne donc pas l'arrêt des autres voies d'investigations.

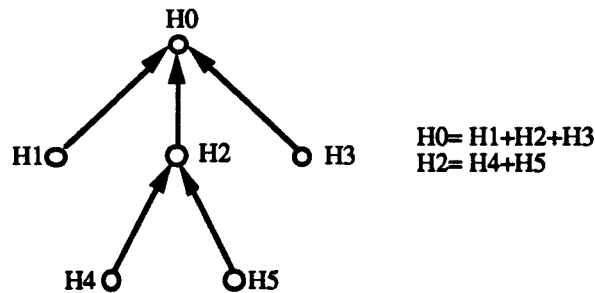


Figure III.34.

* Stratégie d'analyses discriminantes.

L'idée de base de cette stratégie est que si une fonction est défaillante, alors toutes ses fonctions mères, placées dans un contexte particulier, seront défaillantes (Fig. III.35). La notion de contexte s'identifie ici à l'appel de la fonction défaillante...

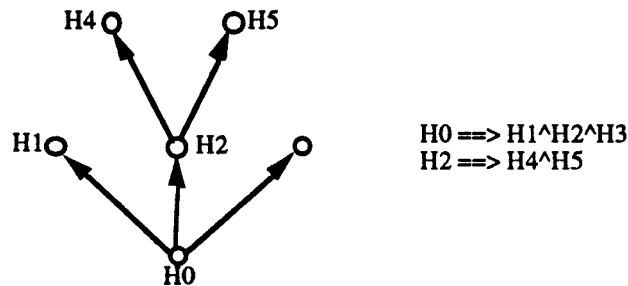


Figure III.35.

Plutôt que d'utiliser la règle précédente, nous utiliserons sa contraposée. Si l'une des fonctions mères, d'une fonction soupçonnée, est non défaillante, alors l'hypothèse de défaillance est rejetée.

II.2.3.3 Le principe

Dans notre approche, nous avons posé comme postulat de base, le principe de défaillances multiples. De ce fait pour mettre en œuvre l'identification et la validation, nous utilisons la stratégie d'analyse concurrentes. Par ailleurs compte-tenu du fait que le modèle fonctionnel est un graphe possédant des noeuds disjonctifs, nous utilisons également la stratégie d'analyses discriminantes. Ce type de stratégie permet de réduire rapidement la combinatoire induite par la présence des noeuds disjonctifs¹.

¹ A partir d'une hypothèse reçue, un noeud disjonctif produit autant d'hypothèses qu'il a de noeuds fils.

II.2.3.4 Mécanisme d'identification.

Le but de l'identification est de déterminer précisément, les fonctions de base (noeuds terminaux), responsables d'une défaillance. Le mécanisme d'identification s'exprime donc de la manière suivante :

Si une fonction est défaillante (une fonction du niveau i), par la stratégie d'analyses concurrentes, on recherche lesquelles de ses fonctions filles (fonctions du niveau $i+1$) sont défaillantes. Pour cela on génère autant d'hypothèse H1-def qu'il y a de fonctions filles. Au niveau de chaque noeud modélisant une fonction fille, l'hypothèse est examinée. Si le noeud est un noeud directement observable, l'hypothèse est directement confirmée ou rejetée ; sinon elle est propagée.

En cas de réfutation d'une hypothèse, il n'y a pas de remontée de l'information vers la fonction défaillante. La confirmation d'une hypothèse est implicitement réalisée, par le mécanisme de pronostic...

Ainsi, le processus d'identification se poursuit de proche en proche à partir des noeuds INITIATEURS.

II.2.3.5 Mécanisme de validation.

Le but de la validation est de confirmer ou rejeter l'hypothèse de défaillance d'une fonction du modèle fonctionnel (Hypothèse H2-def). Cette validation est effectuée en propageant l'hypothèse vers les fonctions filles. La validation est directe pour les fonctions filles directement observables. Pour les autres on propage l'hypothèse de proche en proche jusqu'à atteindre une sous-fonction directement observable.

Lorsque l'exploration d'une voie de recherche aboutit à un échec, la fonction initiatrice en est explicitement informée.

L'hypothèse de défaillance initiale est rejetée lorsque toutes les voies d'exploration aboutissent à un échec. Dans ce cas, le diagnostic local du noeud modélisant la fonction initiatrice conclue alors à une défaillance de son système d'observation.

II.2.4 Mécanisme de pronostic.

II.2.4.1 Condition de déclenchement.

Le pronostic n'est pas réellement un mécanisme de diagnostic. Il fait partie des procédures du recouvrement immédiat (cf. chapitre IV, § I.2.2.1). Il a pour fonction, l'analyse et l'anticipation des conséquences d'une défaillance. Ses conditions de déclenchement sont l'une des suivantes :

1 - La localisation d'une défaillance,

Dès qu'une défaillance est localisée, les noeuds pères du noeud initiateur, doivent en être informés. Ce mécanisme est plus prioritaire que l'identification ou la validation.

2 - L'identification d'une défaillance,

Lorsque la cause d'une défaillance est identifiée, il faut déclencher le pronostic à partir du noeud terminal concerné. Ce déclenchement doit être réalisé même si l'identification fait suite à une phase de localisation. En effet, lors du pronostic initié par une localisation, seuls les noeuds pères du noeud initiateur sont prévenus de la défaillance (Etape 1). En déclenchant de nouveau le pronostic à partir du noeud terminal identifié comme source de la défaillance (Etape 2), on propage l'information à des noeuds qui n'appartiennent pas à un chemin passant par le noeud initiateur. L'étendue du recouvrement est donc plus importante après cette deuxième étape du pronostic (Fig. III.36)

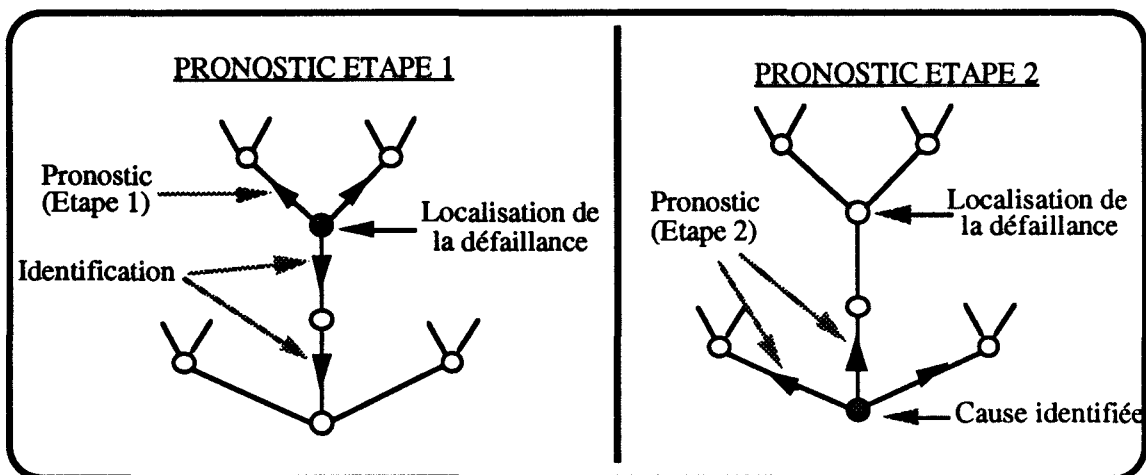


Figure III.36 : Les étapes du pronostic.

3 - La validation d'une hypothèse de défaillance.

En cas de validation, on met en oeuvre l'étape 1 du pronostic. L'étape 2 sera déclenché si la validation est poursuivie par une phase d'identification.

II.2.4.2. Conditions d'arrêts.

Le but du pronostic étant de geler la commande de certaines fonctions, il faut fixer les conditions d'arrêts de ce mécanisme afin d'éviter qu'il y ait une propagation systématique jusqu'au noeud principal. En effet, une telle propagation signifie l'arrêt de l'ensemble du système.

Les conditions d'arrêts sont liées au degré de redondance de chaque fonction :

1 - Cas de la redondance passive.

La fonction défaillante (défaillance directe ou indirecte) peut être remplacée par une fonction du même type mais inactive. L'exemple type dans un atelier est celui de 2 ressources fonctionnellement identiques, dont l'une est au repos.

2 - Cas de la redondance active.

La fonction défaillante peut être remplacée par une fonction du même type déjà active. C'est le cas des 2 fonctions de convoyage liées à chaque échangeur dans l'exemple du CU60.

II.2.4.3 Mise en œuvre.

Le mécanisme du pronostic comprend deux phases :

* **Phase 1** : Elle consiste à évaluer le degré de redondance de la fonction défaillante.

* **Phase 2** : Elle consiste à inactiver (Etat= en-panne) les fonctions dont le degré de redondance est nul. Cette inactivation entraîne la propagation de l'information de défaillance aux noeuds pères (i.e Les fonctions mères) où le mécanisme est répété.

II.2.5 Classification des noeuds du modèle fonctionnel.

II.2.5.1 Introduction

Le concept de base à ce niveau est celui de "**l'intelligence répartie**". L'idée est de retrouver les concepts de capteurs intelligents. Lorsqu'une défaillance est détectée, plutôt que de transmettre l'information brute vers un centre de traitement, on réalise un diagnostic local. Les résultats des différents diagnostics locaux sont échangés par les mécanismes d'identification et de validation. L'information propagée est donc réduite. Cela est conforme à la contrainte temps-réel.

Dans tout ce qui suit, nous allons parler du comportement d'un noeud. Cette notion n'est pas à confondre avec celle introduite dans la modélisation fonctionnelle. Ici le comportement est défini par rapport au processus de diagnostic i.e. que l'on spécifiera l'action dans le cadre du diagnostic, des différents types de noeuds.

II.2.5.2 Classification.

L'ensemble du système de diagnostic s'identifie donc à une **architecture multi-expert**. Dans cette architecture, l'entité de base est le noeud.

Chaque noeud est vu comme un "concept actif" [CHA83/a], [CHA83/b] pouvant réaliser un diagnostic local à partir de son observabilité locale, ou capable de propager des informations déduites ou reçues vers d'autres noeuds.

Nous distinguons quatre types de noeud :

- noeud initiateur,
- noeud de validation directe,
- noeud de validation indirecte,
- noeud simple.

*** Noeud INITIATEUR.**

Ils modélisent les fonctions initiatrices. Ce sont des noeuds directement observables liés à des capteurs d'observation continue (capteur de commande ou capteur de surveillance continue). En cas de défaillance, les noeuds concernés reçoivent des symptômes. Ces derniers initient alors des diagnostics locaux. En fonction du résultat de cette phase locale, chaque noeud peut initier une identification ou une validation.

* Noeud VALIDATION-DIRECTE.

Ce sont également des noeuds directement observables. Mais leur observabilité est due à des capteurs déclenchés à la demande. En mode de fonctionnement normal, ces capteurs sont inactifs. Ils deviennent actifs lorsque le noeud auquel ils sont liés reçoit une demande de validation d'hypothèse. Ce type de noeud permettra l'ouverture du système à d'autres méthodes de diagnostic.

* Noeud VALIDATION-INDIRECTE.

Ce sont des noeuds indirectement observables ayant plusieurs pères ou plusieurs fils. La validation est faite indirectement en utilisant la stratégie d'analyse discriminante (cas d'un noeud ayant plusieurs pères) ou la stratégie d'analyses concurrentes (cas d'un noeud ayant plusieurs fils).

* Noeud SIMPLE.

Ce sont tous les noeuds n'appartenant pas aux types cités précédemment. Ce sont donc des noeuds indirectement observables ayant un père unique et au plus un fils. Les noeuds terminaux simples n'ont pas de fils. Leur présence dans le modèle se justifie par la nécessité de modéliser les fonctions redondantes pour la mise en œuvre des modes de marche dégradés. Leur comportement se limite à propager des informations à travers le réseau.

II.2.5.3 Comportement d'un noeud.

Pour mettre en œuvre le comportement des différents noeuds, nous avons identifié des comportements génériques : **localisation, identification, pronostic, propagation-H2, réponse-H2, validation-directe, validation-indirecte.**

* Localisation ou diagnostic local.

C'est une inférence déclenchée par des symptômes issues de la détection. Elle a pour but d'établir la défaillance de la fonction modélisée ou celle d'un capteur d'observation. Elle est donc spécifique aux noeuds initiateurs.

*** Identification ou propagation-H1.**

Elle correspond au mécanisme d'identification présenté au paragraphe II.2.3.4 . Elle s'apparente à une propagation descendante d'hypothèse H1-def aux différents niveaux du modèle, jusqu'à confirmation ou réfutation par un noeud de validation. Ce comportement est donc général à toutes les classes de noeuds.

*** Pronostic ou réponse-H1.**

Elle correspond au mécanisme de pronostic présenté au paragraphe II.2.4. C'est une propagation ascendante déclenchée dès que l'on est certain de la défaillance d'une fonction. C'est un mécanisme qui survient seulement après une phase d'identification. Il peut être identifié au processus de réponse à la demande de validation d'une hypothèse de type H1-def. C'est également un comportement général à toutes les classes de noeuds.

*** Propagation-H2.**

C'est une propagation avec trace pour la validation d'une hypothèse H2-def. Elle s'effectue des niveaux supérieurs du modèle vers ceux inférieurs. Seuls les noeuds de validation-directe ne possèdent pas ce type de comportement.

*** Réponse-H2.**

C'est le mécanisme de réponse à une requête de validation. On utilise pour cela la trace laissée par propagation-H2. Il est utilisé par tous les types de noeuds saufs les noeuds initiateurs.

*** Validation-directe.**

On utilise l'observabilité directe de certains noeuds pour confirmer ou rejeter des hypothèses reçues. C'est un mécanisme spécifique aux noeuds validation-directe.

*** Validation-indirecte.**

Elle consiste en la vérification de la cohérence d'hypothèses émanant de sources différentes. Nous distinguons deux modes de validation indirecte :

Mode1 : Fils/Pères.

Supposons par exemple qu'un noeud ait plusieurs pères (Fig. III.37). Pour l'identifier, appelons-le **Le-Fils**. Il reçoit une hypothèse de défaillance de l'un de ses pères (Père-1). Supposons qu'un autre des ses pères soit un noeud de validation directe (Père-2). Si on prouve que Père-2 n'est pas défaillant alors l'hypothèse de défaillance du fils est rejetée.

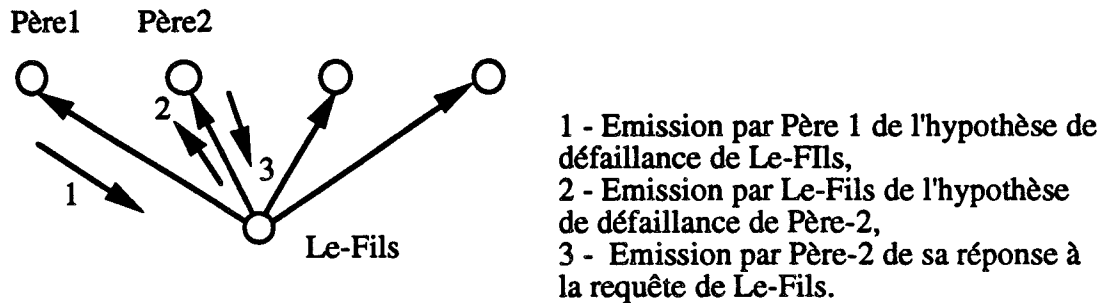


Figure III.37.

Donc dans ce mode, il suffit d'une seule confirmation pour que l'hypothèse de défaillance de la fonction modélisée par Le-Fils soit confirmée ou rejetée.

Mode2 : Fils/Petit-Fils.

Supposons que d'un noeud, soient issus n chemins possédant chacun un noeud de validation directe. D'après la stratégie d'analyses concurrentes, on lance (en parallèle) autant d'hypothèses H2-def qu'il y a de voies de recherche. Si l'on reçoit une réponse positive, alors l'hypothèse est confirmée. Si toutes les réponses sont négatives, alors l'hypothèse de défaillance du noeud est rejetée.

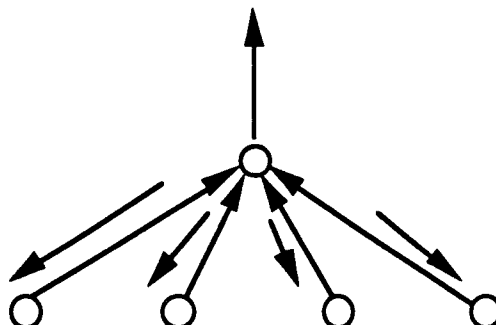


Figure III.38.

La validation indirecte est spécifique aux noeuds validation-indirectes et aux noeuds initiateurs. En effet, un noeud initiateur pouvant propager des hypothèses H2-def, il les valide au retour des réponses en utilisant la validation indirecte.

A chaque type de noeud correspond un ou plusieurs de ces comportements génériques. Nous avons résumé les affectations dans le tableau récapitulatif de la figure III.39.

noeud comportement	initiateur	validation-directe	validation-indirecte	simple
localisation	×			
identification	×	×	×	×
pronostic	×	×	×	×
propagation-H2	×		×	×
réponse-H2			×	×
validation-directe		×		
validation-indirecte	×		×	

Figure III.39 : Tableau récapitulatif des comportements par type de noeuds.

Les croix représentent les liens d'appartenance entre comportements génériques et noeuds.

II.2.6 Conclusion

Les mécanismes de raisonnement présentés dans cette partie sont au nombre de trois : l'identification, la validation et le pronostic. Ensemble ils mettent en œuvre le diagnostic global.

L'identification et la validation sont deux formes de raisonnement hypothétique. La distinction des ces deux mécanismes permet de gagner en efficacité. En effet, notre objectif étant de répartir les centres de diagnostic sur des processeurs répartis (cf. § III), il est

nécessaire de limiter les données échangées. Ainsi, le mécanisme d'identification est moins coûteux (en mobilisation des ressources et donc en temps de réponse) dans la mesure où seul le noeud défaillant doit répondre à l'initiateur du mécanisme. Cette réponse est en fait implicitement réalisée par le pronostic initié par le noeud défaillant. L'identification nécessite toutefois une résolution plus fine par le diagnostic local du noeud initiateur.

La mise en œuvre du pronostic dans la tâche de diagnostic est une originalité de notre approche. En effet, cette fonction met en œuvre l'analyse des conséquences d'une défaillance afin d'assurer l'évitement de faute i.e la sollicitation d'une entité défaillante par une autre partie de la commande. Elle s'intègre donc à l'action du recouvrement à court terme (cf . chapitre IV). Nous justifions son insertion dans le diagnostic par la nécessité d'utilisation du graphe fonctionnel. En effet, nous aurions pu assigner au diagnostic de se contenter de communiquer au module de recouvrement le nom de la fonction défaillante. Une telle approche nécessiterait de dupliquer le graphe fonctionnel au niveau du recouvrement, avec tous les problèmes de cohérence des modèles que cela impose.

III. SPECIFICATIONS INFORMATIQUES .

III.1 Principe

Il est basé sur une mise en œuvre à partir d'une approche hybride : c'est à dire l'utilisation conjointe de la Programmation Orientée Objets (POO) et de la modélisation de la connaissance par des schémas ("frames"). L'idée de base est de constituer des classes d'objets pour chaque type de noeud identifié. Pour une application donnée, le concepteur crée des instances de ces classes, en se basant sur le GF du système. Au niveau de chaque noeud du réseau, les mécanismes locaux permettent d'interpréter les symptômes reçus ou les données propagées par d'autres noeuds. Ils utilisent le mécanisme de démons (procédures réflexes) qui est propre aux langages de schémas. Les échanges inter-noeuds sont réalisés par envoi de messages (Hypothèse Etat-fonction ?, Réponse Etat-fonction, ... ; cf . figure III.40) Les échanges avec les éléments externes sont également mis en œuvre par le mécanisme de message (la réception des actions, symptômes, ... et l'émission des causes de défaillances ; cf . figure III.40).

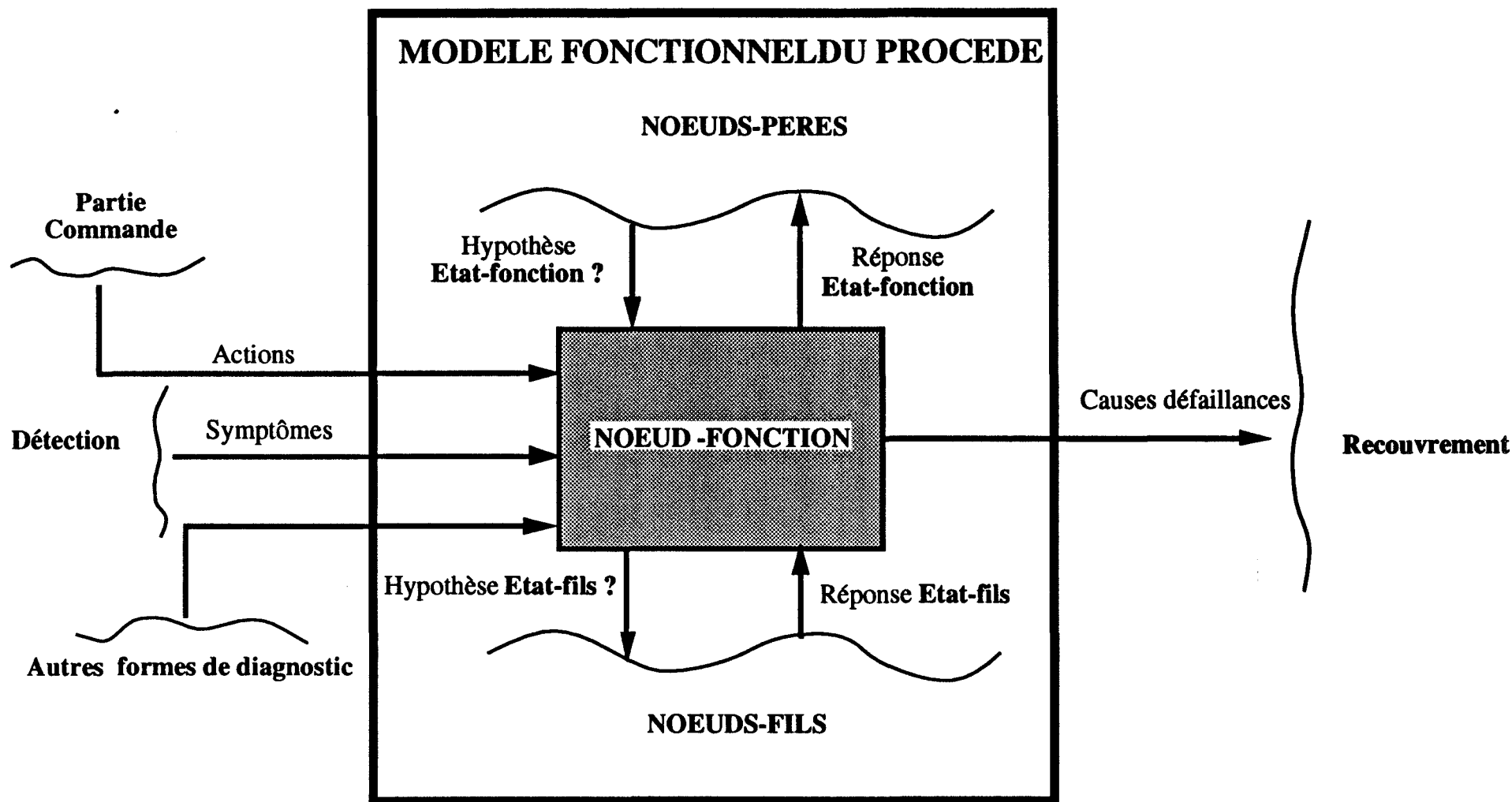


Figure III.40 : Schéma de principe

Les spécifications seront données dans le formalisme de KOOL [BUL90/a], [BUL90/b], [BUL90/c].

III.2 Hiérarchie des noeuds.

Cette hiérarchie repose sur la POO afin de mettre en facteur les comportements communs à plusieurs types de noeuds. Ainsi, nous définissons d'abord un noeud de base dont la structure permet de mettre en œuvre les mécanismes d'identification et de pronostic.

```
{ Class : NoeudBase
Super : Object
<-- Nom-fonction : () ; Donne le nom de la fonction modélisée par le noeud
    Mode : Mono
    Type : String
-- Genre : ()
    Mode : Mono
    Type : [initiateur, simple, validation-directe, validation-indirecte]
-- Fils : [ ] ; liste des noeuds fils
    Mode : Multi
    Type : NoeudBase
-- Pères : [ ] ; liste des noeuds pères
    Mode : Multi
    Type : NoeudBase
-- Etat : () ; C'est l'état d'un noeud
    Mode : Mono
    Type : [normal, dégradé, en-panne]
    Init : normal ; Valeur initiale
    WhenFilled : [pronostic] ; Le pronostic est déclenché lorsque la valeur de Etat=en-panne
-- Nb-alternatives : () ; Spécifie le degré de redondance pour la fonction associée. Si la
    valeur est 0, en cas de défaillance Etat=en-panne ; sinon Etat=dégradé
    Mode : Mono
    Type : Number
    Default : 1 ; Spécifie la valeur par défaut
```

```

-- Composant          : () ; Donne le nom du composant courant associé à la fonction
    Mode              : Mono
    Type              : String
-- H1-def             : [] ; Liste d'hypothèses de type H1
    Mode              : MultiH1
    WhenFilled       : [identification]
-- H2-def             : [] ; Liste d'hypothèses de type H2
    Mode              : MultiH2
    WhenFilled       : [propagation-H2]
    WhenRemoved     : [reponse-H2]
-- Diag-externe      : [] ; liste de données propagées par d'autres noeuds
    Mode              : Multi
    WhenFilled       : [evaluer1]

Methods             :
    AddData             : AjoutDonnee > }

```

La fonction **AjoutDonnee** permet d'ajouter dans l'attribut **Diag-externe**, les données émises par d'autres noeuds.

A partir de **NoeudBase**, nous définissons quatre sous-classes correspondantes aux familles de noeuds répertoriés. Ainsi nous avons :

```

{ Class : NoeudSimple
Super : NoeudBase
<-- Genre          : simple >

```

```

{ Class : NoeudInitiateur
Super : NoeudBase
<-- Genre          : initiateur
-- Action          : () ; Spécifie l'ordre à exécuter ou Start-Event
    Mode              : Mono
    RuleWhenFilled   : [] ; Spécifie les règles qui seront déclenchées en
                        ; chaînage avant lorsque la valeur de l'attribut sera modifiée.
-- Symptomes-reçus : []
    Mode              : MultiSymptom

```

WhenFilled : [deduction]
 -- R/STC : [] ; Liste des STC associées au noeud
Mode : Multi
 -- Diag-interne : [] ; Liste de données déduites par l'utilisation des
 ; STC
Mode : Multi
WhenFilled : [interpreter]
 -- H-collage : [] ; Liste des hypothèses de collage de capteur
Mode : Multi

Methods :
 AddSymp : AjoutSymptome ; Ajoute un symptôme à
 "Symptomes-reçus"
 RemSymp : SuppSymptome; Supprime un symptôme de
 "Symptomes-reçus" > }

Le démon "interpreter" sert à interpréter le résultat du diagnostic interne de chaque noeud initiateur et à le répartir dans des attributs comme "H1-def" et "H2-def". L'attribut "Diag-interne" est une liste constituée elle même de sous-liste. Chaque sous-liste est une des conclusions de la partie postérieure d'une STC. Ainsi, le mécanisme de "interpreter" est le suivant :

Pour toute sous-liste de "Diag-interne", faire :

(1) Si prédicat = def alors

(1.1) si la queue est un atome alors

(1.1.1) si cet atome est dans l'attribut "Fils" alors

(1.1.1.1) émettre la sous-liste vers le noeud fils concerné

supprimer le noeud fils de la liste "Fils"

(1.1.1.2) sinon placer dans l'attribut "H1-def", une hypothèse de type
 H1-def par noeud fils.

(1.1.2) sinon émettre la queue vers le module de recouvrement

(1.2) sinon si prédicat = H1-def alors

(1.2.1) si la queue est un atome alors

(1.2.1.1) si cet atome est dans l'attribut "Fils" alors

(1.2.1.1.1) ajouter la sous-liste dans "H1-def"

(1.2.1.1.2) sinon placer dans l'attribut "H2-def", une
 hypothèse de type H2-def par noeud fils.

(1.2.1.2) **sinon** ajouter la sous-liste dans l'attribut "H-collage"

(2) Supprimer la sous-liste de "Diag-interne"

```
{ Class : NoeudValidationDirecte
Super : NoeudBase
<-- Genre          : validation-directe
-- H2-def          : [ ] ; Liste d'hypothèses de type H2
    Mode           : MultiH2
    WhenFilled     : [validationDirecte]
    WhenRemoved   : [reponse-H2] > }
```

```
{ Class : NoeudValidationIndirecte
Super : NoeudBase
<-- Genre          : validation-indirecte
-- H2-def          : [ ] ; Liste d'hypothèses de type H2
    Mode           : MultiH2
    WhenFilled     : [validationIndirecte]
    WhenRemoved   : [reponse-H2] }
```

Dans la définition des classes NoeudValidationDirecte et NoeudValidationIndirecte, nous avons utilisé la technique du surnomage pour déclencher les comportements réflexifs liés à l'attribut H2-def.

Pour chacune de ses deux dernières classes, nous définissons des sous classes pour mettre en œuvre des mécanismes de validation type.

La hiérarchie complète est décrite par le schéma suivant :

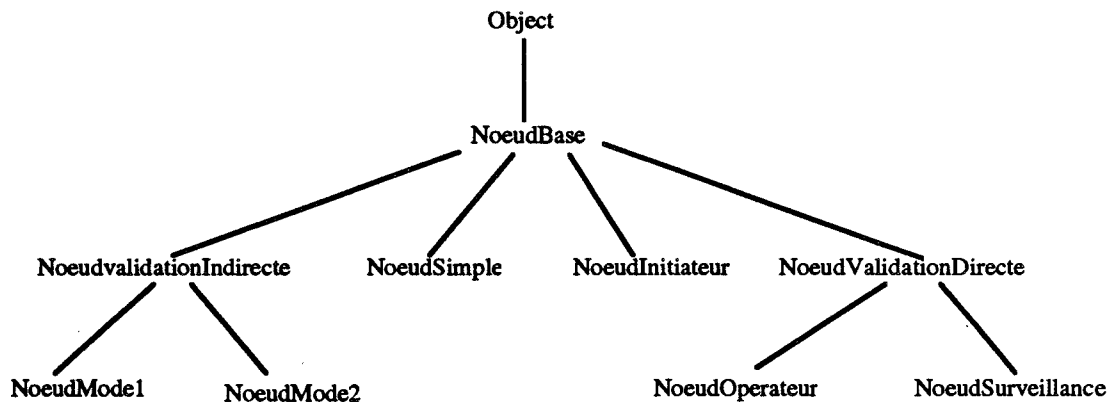


Figure III.41 : Hiérarchie des noeuds.

III.3 Classes de symptômes et de STC.

Dans ce paragraphe, nous allons définir des classes liées aux notions de symptômes et de STC introduites au chapitre I (§ IV.5) et au chapitre III (§ I.4.4).

III.3.1 Classes de symptômes.

III.3.1.1 Méthodes personnelles à la classe "Symptome".

Nous redéfinissons la méthode **New** utilisée pour la création dynamique d'un objet. La différence essentielle est que lorsqu'on crée une instance de la classe **Symptome**, elle est immédiatement datée. La datation se fait par l'utilisation de **Define** et des fonctions "**madate**" et "**horloge**".

III.3.1.2 Attributs du schéma d'instance d'un symptôme.

Chaque symptôme est caractérisé par un "**capteur**" et son "**stype**" (type I ou type II). Par ailleurs, le raisonnement temporel impose la datation (attributs "**date**" et "**heure**") des occurrences d'un symptôme. Lorsqu'une instance d'un symptôme est créée, elle doit être envoyée aux fonctions où il est attendu. Nous définissons donc pour chaque symptôme l'attribut "**fonctions-associees**". Chaque symptôme est également caractérisé par les STC lancées auxquelles il a été attribué. Ces STC sont regroupées sous un attribut nommé

“L/STC”. Cet attribut mémorise en fait l’ensemble des conflits dû à l’attribution du symptôme à plus d’une STC.

III.3.1.3 Méthodes du schéma d’instance d’un symptôme.

Nous définissons deux méthodes qui permettent de modifier dynamiquement l’attribut L/STC. Il s’agit de :

* AjouterSTC (Sélecteur : AddSTC)

Cette méthode permet d’ajouter une STC à la liste de STC lancées par le symptôme. Chaque STC envoie au symptôme un message du type (<-- <symptôme> AddSTC <nom-STC>). Ce message sert de confirmation au lancement de la STC par le symptôme.

* SupprimerSTC (Sélecteur : RemSTC).

Lorsqu’un “timeout” expire dans une STC, elle est détruite par le noeud dont elle dépend. Chaque symptôme qui avait été “attribué” à la STC en est averti de manière à réduire sa liste de STC en conflit (L/STC). Il reçoit un message du type (<-- <symptôme> RemSTC <nom-STC>).

III.3.1.4 Démons associés aux attributs de la classe symptôme.

Nous définissons qu’un seul démon : “AvertirOperateur”. Il est associé à l’attribut L/STC. Si L/STC = [], il faut envoyer une alarme à l’opérateur. En effet, L/STC = [] signifie que l’occurrence du symptôme n’a pu être attribuée à aucune STC validée. Il y a donc incomplétude de la connaissance codée. En principe ce mécanisme est seulement utile lors de la conception du module de diagnostic.

III.3.1.5 Transcription des spécifications en KOOL.

{ Class : Symptome

Super : Object

-- Methods :

 New : CreerSymptome

<-- capteur : () ; nom du capteur associé

Mode : Mono

```

    Type           : String
-- stype          : () ; type du symptôme
    Mode           : Mono
    Type           : [typeI, typeII]
-- date           : () ; date de création de l'instance
    Mode           : Mono
    Type           : String
-- heure          : () ; heure de création du symptôme. Elle est donnée en secondes
    Mode           : Mono
    Type           : Number
-- fonctions-associees : [] ; liste des fonctions auxquelles le symptôme sera envoyé.
    Mode           : Multi
    Type           : String

-- L/STC           : [ ] ; Liste des STC lancées auxquelles le symptôme
    aura été attribuée
    Mode           : Multi
    Type           : STC
    WhenRemoved : [AvertirOperateur]

Methods          :
    AddSTC           : AjouterSTC
    RemSTC           : EnleverSTC>}

```

III.3.2 Classe de STC.

III.3.2.1 Attribut du schéma d'instance.

Une STC est d'abord caractérisée par la fonction à laquelle elle est rattachée. Cette fonction lorsqu'elle est activée constitue l'élément initiateur de la reconnaissance ou Start-Event (cf . chapitre III, § I.4.4.4). Nous définissons donc l'attribut "**fonction-associee**" qui donne le nom de la fonction à laquelle est associée la STC.

Par ailleurs une STC comprend deux parties :

* La partie antérieure constituée d'une liste de symptômes contraints ; nous définissons donc l'attribut "**anterieure**" ; pour définir ces listes, nous introduisons une nouvelle classe d'attributs nommée SymptomeContraint.

* La partie postérieure constituée d'une liste de conséquences ; nous définissons donc l'attribut **"postérieure"**.

A ces attributs, nous ajoutons la paire **"symptome-attendu"** et **"timeout"**. **"Symptome-attendu"** indique le symptôme courant à vérifier dans la STC. **"Timeout"** donne la valeur maximale d'attente du symptôme.

Enfin nous définissons l'attribut **"Symptome-reçu"** qui indique le symptome reçu.

III.3.2.2 Méthodes du schéma d'instance.

Elles sont au nombre de deux. Il s'agit de :

* AjouterSymptome (Sélecteur : AddSymp).

Cette méthode permet l'ajout d'un symptôme à l'attribut **"Symptome-reçu"**. L'ajout est effectué à la réception d'un message du type (**<-- <STC> AddSymp <nom-symptome>**), émis par la fonction associée.

* Decompter (Sélecteur : Decompter).

Permet de décrémenter la présélection du **"timeout"** à chaque top d'horloge.

III.3.2.3 Démons liés aux attributs.

Pour la classe des STC, nous définissons deux démons :

*** "comparer".**

Il est associé à tout ajout de valeur à l'attribut **"Symptome-reçu"**. Ce démon met en œuvre des mécanismes essentiels à l'interprétation des STC. Il peut être décrit de la manière suivante :

(1) **comparer** le symptôme reçu au symptôme attendu.,

(2) s'ils sont égaux, **vérifier** si la contrainte est vérifiée,

(3) si la contrainte est vérifiée, alors

(3.1) **supprimer** le symptôme de la liste des antécédents,

(3.2) si antecedents = [] alors émettre “conclusion” vers “fonction-associee”
sinon **evaluer** “symptome-attendu” et “timeout”.

*** AvertirFonction?**

Ce démon est déclenché à chaque modification de l’attribut “timeout”. Son mécanisme est le suivant :

(1) si timeout = 0 alors

(1.1) si le symptome-attendu est une négation alors

(1.1.3) **supprimer** le symptôme de la liste des antécédents,

(1.1.4) si antecedents = [] alors émettre “posterieure” vers “fonction-associee”
sinon **evaluer** “symptome-attendu” et “timeout”.

(1.2) sinon **avertirfonction**

III.3.2.4 Transcription des spécifications en KOOL.

{ **Class** : STC

Super : Object

<-- **fonction-associee** : () ; nom de la fonction associée

Mode : Mono

Type : String

-- **antecedents** : [] ; liste de symptômes contraints

Mode : Multi

Type : SymptomeContraint

-- **posterieure** : [] ; liste des conclusions de la STC

Mode : Multi

-- **symptome-attendu** : () ; symptôme attendu

Mode : Mono

Type : Symptome

-- **timeout** : () ; valeur du chien de garde servant au contrôle de la STC

```

    Mode           : Mono
    Type           : Number
    WhenFilled      : [AvertirFonction?]
-- Symptome-reçu   : () ; symptôme reçu
    Mode           : Mono
    Type           : Symptome

Methods           :
    AddSymp        : AjouterSymp
    Decompter      : Decompter >}

```

III.3.3 La classe d'attribut SymptomeContraint.

Cette classe est utilisée dans la définition des STC.

III.3.3.1 Attributs de SymptomeContraint.

Il y en a trois :

* **“symp”**.

Il indique le symptôme référencé.

* **“contrainte”**.

Ce descripteur (attribut d'attribut) donne la contrainte liant le symptome à son prédécesseur dans la séquence.

* **“negation”**

Ce descripteur spécifie si le symptôme est affirmé (negation = non), ou nié (negation = oui).

III.3.3.2 Transcription des spécifications en KOOL.

Nous commençons par définir les descripteurs de la classe :

```

{MonoAttributeAttribute : symp}
{MonoAttributeAttribute : contrainte}
{MonoAttributeAttribute : negation}

```

Ensuite la classe s'écrit de la manière suivante :

{ **ClassAttribut** : SymptomeContraint

Super : Mono

```
<-- symp          : () ;
    Mode          : MonoAttributeAttribute
    Type          : Symptome
-- contrainte     : () ;
    Mode          : MonoAttributeAttribute
    Type          : [periode, instant, duree]
-- negation       : () ; Spécifie si le symptôme est affirmé (negation =
    non), ou nié (negation = oui).
    Mode          : MonoAttributeAttribute
    Type          : [oui, non] > }
```

III.4 Exemple : Spécification Informatique du GF du SAS.

Les fonctions Avance/porte et Avance/vérin/SAS sont directement observables. D'autre part, distribution/P a deux pères. On peut donc y mettre en ouvre un mécanisme de validation indirecte du type Model.

Commençons par modéliser le GF du SAS :

```
{ NoeudSimple : Accès/sécurité
-- Non-fonction      : "Accès/sécurité"
-- Fils              : [ Avance/porte ]
-- Pères             : [ Usinage ]
-- Nb-alternatives : 1
-- Composant         : "SAS" }

{NoeudInitiateur : Avance/porte
-- Non-fonction      : "Avance/porte"
-- Fils              : [ Avance/vérin/SAS ]
-- Pères             : [ Accès/Sécurité ]
```

```
-- Nb-alternatives : 1
-- Composant      : "Porte"
-- Action         : ()
    type          : [fermeture, ouverture]
    RuleWhenFilled : [ R1, R2 ]
--R/STC           : [ ]
    type          : [ STC1, STC2, STC3, STC4, STC5, STC6, STC7, STC8 ] }
```

avec :

```
{ Rule+ : R1 ; Spécifie que R1 est une règle déclenchée en chaînage avant
If Avance/porte^Action = fermeture
Then Avance/porte^R/STC <- [STC1, STC2, STC3, STC4] }
```

```
{ Rule+ : R2 ; Spécifie que R2 est une règle déclenchée en chaînage avant
If Avance/porte^Action = ouverture
Then Avance/porte^R/STC <- [STC5, STC6, STC7, STC8] }
```

```
{NoeudInitiateur : Avance/vérin/SAS
-- Non-fonction    : "Avance/vérin/SAS"
-- Fils            : [ Transmission/SAS, Glisse/corps/SAS ]
-- Pères           : [ Avance/porte ]
-- Nb-alternatives : 1
-- Composant       : "vérin/SAS"
-- Action          : ()
    type           : [avance, recul]
    RuleWhenFilled : [ R3, R4 ]
--R/STC            : [ ]
    type           : [ STC9, STC10, STC11, STC12, STC13, STC14, STC15,
    STC16] }
```

```
{ Rule+ : R3 ; Spécifie que R1 est une règle déclenchée en chaînage ; avant
If Avance/vérin/SAS^Action = avance
Then Avance/vérin/SAS^R/STC <- [STC9, STC10, STC11, STC12] }
```

```
{ Rule+ : R4 ; Spécifie que R2 est une règle déclenchée en chaînage ; avant
If Avance/vérin/SAS^Action = recul
```

Then Avance/vérin/SAS^R/STC <- [STC13, STC14, STC15, STC16] }

{NoeudSimple : Transmission/SAS

-- Non-fonction : "Transmission/SAS"
-- Fils : []
-- Pères : [Avance/porte]
-- Nb-alternatives : 1
-- Composant : "SAS"}

{ NoeudSimple : Glisse/corps/SAS

-- Non-fonction : "Glisse/corps/SAS"
-- Fils : []
-- Pères : [Avance/vérin/SAS]
-- Nb-alternatives : 1
-- Composant : "vérin/SAS"}

{NoeudMode2 : Distribution/S1

-- Non-fonction : "Distribution/S1"
-- Fils : [Distribution/P, Conduite/S1]
-- Pères : [Avance/vérin/SAS]
-- Nb-alternatives : 1
-- Composant : "distributeur-secondaire/S1"}

{ NoeudSimple : Conduite/S1

-- Non-fonction : "Conduite/S1"
-- Fils : [Avance/porte]
-- Pères : [Usinage]
-- Nb-alternatives : 1
-- Composant : "distributeur-secondaire/S1"}

{NoeudMode2 : Distribution/P

-- Non-fonction : "Distribution/P"
-- Fils : [Pompage, Conduite/P]
-- Pères : [Distribution/S1]
-- Nb-alternatives : 1
-- Composant : "distributeur-primaire/P"}

{ NoeudSimple : Conduite/P

-- Non-fonction : "Conduite/P"
-- Fils : []
-- Pères : [Distribution/P]
-- Nb-alternatives : 1
-- Composant : "distributeur-primaire/P" }

{ NoeudSurveillance : Pompage

-- Non-fonction : "Pompage"
-- Fils : [Conduite/pompe, Stockage/cuve]
-- Pères : [Distribution/P]
-- Nb-alternatives : 1
-- Composant : "Pompe" }

{ NoeudSurveillance : Stockage/cuve

-- Non-fonction : "Stockage/cuve"
-- Fils : []
-- Pères : [Pompage]
-- Nb-alternatives : 1
-- Composant : "cuve" }

{ NoeudSimple : Conduite/Pompe

-- Non-fonction : "Conduite/Pompe"
-- Fils : []
-- Pères : [Pompage]
-- Nb-alternatives : 1
-- Composant : "cuve" }

Nous allons maintenant décrire les STC des noeuds initiateurs. Pour cela, nous utilisons le modèle temporel correspondant aux actions liées aux capteurs po-av, po-ar, ve-av et ve-ar (Fig. III.42).

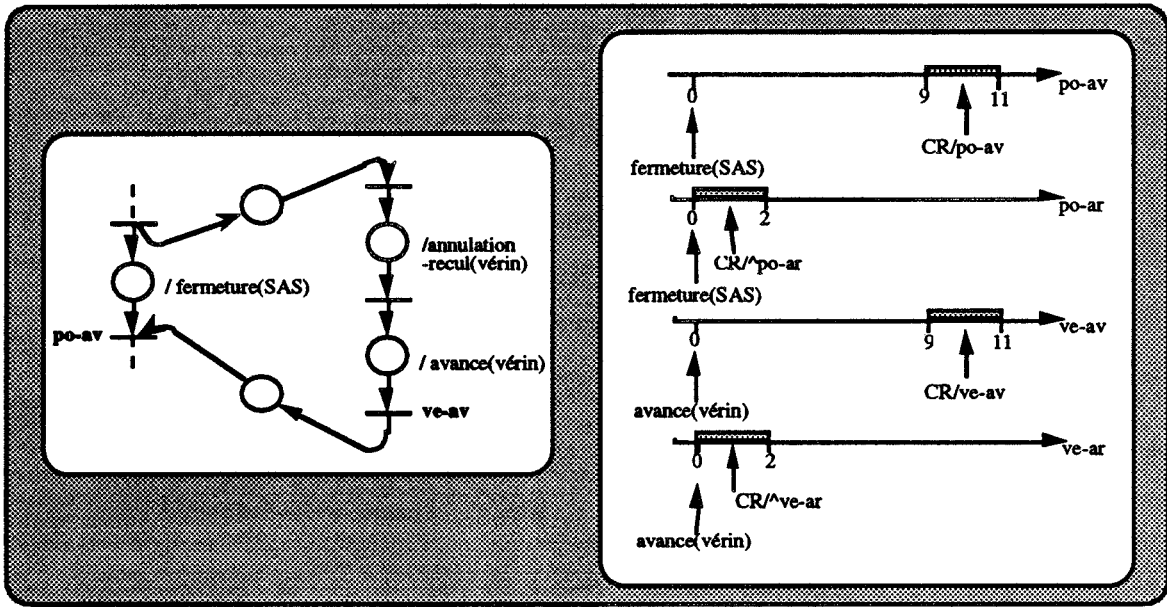


Figure III.42.

L'ensemble des symptômes considérés est :

$$\{ S^1_{po-av}, S^2_{po-av}, S^1_{po-ar}, S^2_{po-ar}, S^1_{ve-av}, S^2_{ve-av}, S^1_{ve-ar}, S^2_{ve-ar} \}$$

Ce qui nous donne les STC suivantes :

* Pour le noeud Avance/porte :

$$STC1 : (occ\ fermeture) * (2 S^1_{po-ar}) * (9 S^1_{po-av}) \Rightarrow (def\ Avance/porte)$$

$$STC2 : (occ\ fermeture) * (2 S^1_{po-ar}) * (9 \neg S^1_{po-av}) \Rightarrow (def\ (collage\ po-ar\ 1))$$

$$STC3 : (occ\ fermeture) * (2 \neg S^1_{po-ar}) * (9 S^1_{po-av}) \Rightarrow (H1-def\ (collage\ po-av\ 0)) \wedge (H1-def\ Avance/porte)$$

$$STC4 : (S^1_{po-ar}) * (-2 \neg(occ\ fermeture)) \Rightarrow (H1-def\ (collage\ po-ar\ 1))$$

$$STC5 : (occ\ ouverture) * (2 S^1_{po-av}) * (9 S^1_{po-ar}) \Rightarrow (def\ Avance/porte)$$

$$STC6 : (occ\ ouverture) * (2 S^1_{po-av}) * (9 \neg S^1_{po-ar}) \Rightarrow (def\ (collage\ po-av\ 1))$$

$$STC7 : (occ\ ouverture) * (2 \neg S^1_{po-av}) * (9 S^1_{po-ar}) \Rightarrow (H1-def\ (collage\ po-ar\ 0)) \wedge (H1-def\ Avance/porte)$$

$$STC8 : (S^1_{po-av}) * (-2 \neg(occ\ ouverture)) \Rightarrow (H1-def\ (collage\ po-av\ 1))$$

* Pour le noeud Avance/vérin/SAS :

$$STC9 : (occ\ fermeture) * (2 S^1_{ve-ar}) * (9 S^1_{ve-av}) \Rightarrow (def\ Avance/vérin/SAS)$$

$$STC10 : (occ\ fermeture) * (2 S^1_{ve-ar}) * (9 \neg S^1_{ve-av}) \Rightarrow (def\ (collage\ ve-ar\ 1))$$

$$STC11 : (occ\ fermeture) * (2 \neg S^1_{ve-ar}) * (9 S^1_{ve-av}) \Rightarrow (H1-def\ (collage\ ve-av\ 0)) \wedge (H1-def\ Avance/vérin/SAS)$$

$STC12 : (S^1_{ve-ar}) * (-2 \neg(occ\ fermeture)) \Rightarrow (H1-def (collage\ ve-ar\ 1))$
 $STC13 : (occ\ ouverture) * (2 S^1_{ve-av}) * (9 S^1_{ve-ar}) \Rightarrow (def\ Avance/vérin/SAS)$
 $STC14 : (occ\ ouverture) * (2 S^1_{ve-av}) * (9 \neg S^1_{ve-ar}) \Rightarrow (def (collage\ ve-av\ 1))$
 $STC15 : (occ\ ouverture) * (2 \neg S^1_{ve-av}) * (9 S^1_{ve-ar}) \Rightarrow (H1-def (collage\ ve-ar\ 0)) \wedge (H1-def\ Avance/vérin/SAS)$
 $STC16 : (S^1_{ve-av}) * (-2 \neg(occ\ ouverture)) \Rightarrow (H1-def (collage\ ve-av\ 1))$

A partir des règles précédentes, nous déduisons la spécification orientée-objet suivante :

{ STC : STC1 ; Définit STC1 comme une instance de la classe STC

-- fonction-associee : Avance/porte
-- antecedents : [SC1, SC2,] ;
-- posterieure : [(def Avance/porte)] ;
-- symptome-attendu : ()
-- timeout : ()
-- Symptome-reçu : () }

avec :

{ SymptomeContraint : SC1 ; Définit SC1 comme une instance de la classe
; SymptomeContraint

-- symp : S^2_{po-ar}
-- contrainte : (0 2)
type : duree ; Définit le type de la contrainte
-- negation : non }

{ SymptomeContraint : SC2

-- symp : S^1_{po-av}
-- contrainte : 9
type : instant
-- negation : non }

Supposons que avance/porte reçoive le symptôme S^1_{ve-av} après une demande de fermeture de porte. En l'absence de S^2_{ve-ar} , son mécanisme de déduction interne va établir l'hypothèse (H2-def avance/porte). Son attribut H2-def sera alors complété ce qui déclenchera la procédure propagation-H2. Le noeud TRANSMISSION/SAS étant un noeud simple terminal, il ne peut rien répondre. Par contre Avance/vérin/SAS confirmera ou rejettera la

réponse en fonction des symptômes qu'il aura reçu. S'il rejette la réponse, avance/porte conclura que TRANSMISSION/SAS est défaillant. Si Avance/vérin/SAS confirme qu'il est défaillant, en parallèle il aura initié l'identification de la défaillance. Quant au noeud avance/porte il évalue alors son ETAT comme étant en panne ce qui a pour conséquence de déclencher un mécanisme de pronostic vers ACCES-SECURITE.

III.5 Mécanisme d'ensemble.

Notre but dans cette partie est de donner une vision synthétique des différents mécanismes déclenchés pour assurer le diagnostic. Bien que de nombreux traitements soient exécutés en parallèle, nous allons le présenter sous forme algorithmique afin d'en faciliter la compréhension.

(1) Mise à jour de l'attribut "R/STC" des noeuds de la classe NoeudInitiateur par déclenchement des règles associées à l'attribut "Action".

(2) A la réception du signal de détection d'un symptôme de type $S^i_{<cap>}$, création d'une instance $S^i_{<cap>-j}$;

i désigne le type du symptôme ;

$<cap>$ donne le nom du capteur auquel est associé le symptôme ;

j désigne le n° de l'instance dans sa classe.

(3) Dès que la valeur de l'attribut "heure" de $S^i_{<cap>-j}$ est ajoutée, envoie du symptôme à toutes les fonctions référencées dans son attribut "fonction-associée" ;

message : (`<--* (getLocal <object> fonction-associee) AddSymp <object>`)

(4) Lors de la réception du message précédent, chaque fonction "associée" ajoute le symptôme dans sa liste "Symptome-reçu".

(5) L'insertion d'un symptôme dans l'attribut "Symptomes-reçus" d'un noeud initiateur déclenche la "localisation" en utilisant les STC définies dans "R/STC".

Pour chaque classe de STC référencée dans "R/STC" et commençant par le symptôme reçu, on crée une nouvelle instance.

Ensuite le symptôme est envoyé en diffusion à toutes les instances de classe de STC référencées dans "R/STC".

(6) Chaque instance de STC recevant le symptôme l'affecte à son attribut "Symptome-reçu". Cela déclenche le démon "comparer".

(7) Lorsque que la comparaison est positive :

(7-1) la STC est mise à jour (cf. définition de "comparer" § III.3) et un accusé de réception est émis vers l'objet symptôme.

message : (<-- (getLocal <object> fonction-associee) AddSTC <object>)

(7-2) si toute la partie antérieure de la STC est vérifiée, les conclusions sont émises vers la fonction définie dans l'attribut "fonction-associee" de la STC.

message : (<-- (getLocal <object> fonction-associee) AddData (getLocal <object> posterieure))

(8) A la réception d'un accusé de réception, un objet symptôme ajoute le nom de la STC émettrice dans la liste de STC en cours de vérification auxquelles il a été attribué (l'attribut "L/STC").

(9) A la réception des conclusions d'une STC, les objets de la classe NoeudInitiateur les ajoutent dans la liste de leur attribut "Diag-interne". Cela provoque le déclenchement du démon "interpréter" (cf. la définition de NoeudInitiateur au § III.2)

(10) Identification

(11) Pronostic

(12) "Timeout" dans une STC

(13) Suppression de la STC

(14) Suppression de la trace de la STC dans l'attribut "L/STC" des symptômes qui lui avaient été attribués

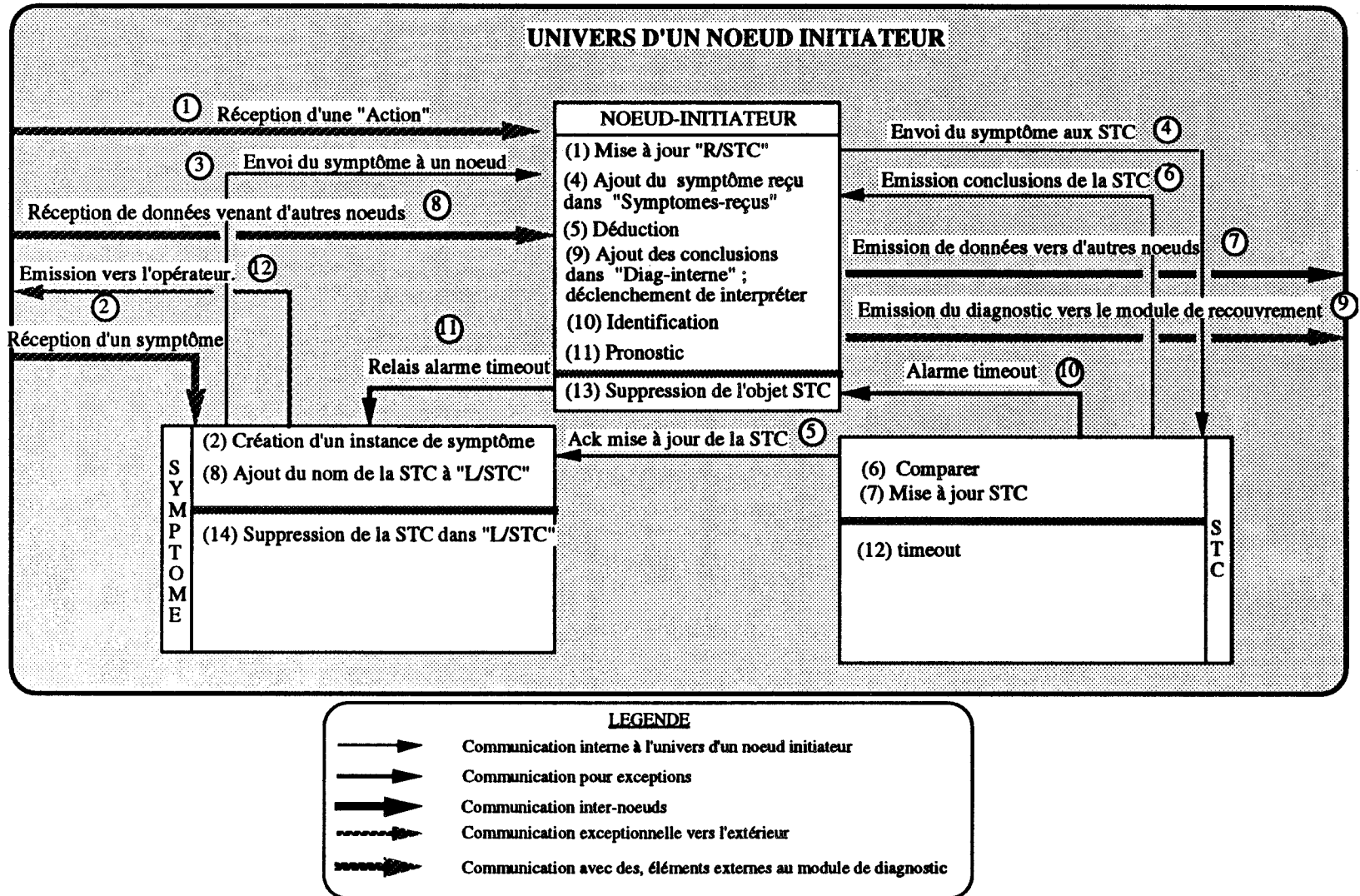


Figure IV.43 : Spécification des mécanismes d'un noeud initiateur

III.6 Conclusion

L'ensemble des mécanismes spécifiés dans cette partie repose sur des communications entre différents types d'objets : les différents modules de la surveillance (détection, diagnostic, recouvrement) , les noeuds du GF, les STC et les symptômes. L'utilisation des communications s'explique par :

- le parallélisme mis en œuvre dans l'exécution des différents traitements,
- l'aspect réparti de l'application.

Du fait de l'aspect réparti de notre application, il eut été plus judicieux de mettre en œuvre ce prototype avec un langage d'acteurs du type de PLASMA ou des ses dérivés (Act1, Act2, ...) [MAS89]. Notamment, de nombreux traitements comme l'identification ou le pronostic aurait pu être mis en œuvre de manière transparente en utilisant le mécanisme de délégation propre aux acteurs.

Néanmoins, le choix de KOOL se justifie par les raisons suivantes :

1 - L'ensemble de la méthodologie CASPAIM est basée sur l'utilisation d'un langage unique, Le-Lisp pour tout ce qui touche à l'intelligence artificielle. Par soucis de cohérence avec les travaux antérieurs du L.A.I.L., nous nous devions d'utiliser pour le prototype un système de développement de systèmes experts construit à partir de Le-Lisp. C'est le cas de KOOLv2.

2 - Toutes les méthodes de conception développées dans le cadre de CASPAIM ([BOU88], [CRA89], [BOI91]) , ont toujours été guidées par le souci de faciliter la mise en œuvre sur site, du modèle de conception. Pour se faire, il faut que le modèle d'implémentation soit très peu différent du modèle de conception.

De ce fait, l'utilisation de KOOL est un avantage. En effet, l'idée est d'utiliser KOOLv2 pour la conception et KOOLv4 pour l'implémentation. KOOLv4 fournit le code de l'application en langage C. Cela assure donc la vitesse de traitement nécessaire à une application temps réel.

La transition de KOOLv2 à KOOLv4 est quasi immédiate. En effet, l'ensemble des objets développés sous KOOLv2 sont réutilisables sous KOOLv4. Le concepteur doit juste transcrire en C, les fonctions qu'il avait écrites en Le-Lisp.

CONCLUSION

Notre approche du diagnostic en ligne est basée sur quatre concepts fondamentaux :

- la modélisation fonctionnelle,
- le raisonnement temporel,
- le raisonnement hypothétique,

- une mise en œuvre répartie.

Nous avons introduit la modélisation fonctionnelle comme une forme de modélisation qualitative causale. Elle permet d'obtenir un modèle de connaissance exploitable en temps réel tout en assurant sa complétude. Nous avons présenté au § I les grandes lignes de la méthode de modélisation. Notre but à terme est d'assister cette démarche de modélisation par le développement d'un outil de CAO.

Afin d'interpréter les symptômes détectés par le module de détection, nous avons mis en œuvre un formalisme original, permettant de faire du raisonnement temporel. Ce formalisme est basé sur l'utilisation d'un type nouveau de règles : les STC. Ces STC peuvent être interprétées aussi bien dans une approche centralisée que répartie. Avec notre approche, il est possible à tout moment de faire des prédictions. En effet, à un instant donné, les conclusions de l'ensemble des STC lancées correspondent aux prédictions de défaillances qui peuvent être faites sur la base des symptômes déjà apparus. L'occurrence de nouveaux symptômes permettra ensuite de réduire ces prédictions jusqu'au diagnostic final. L'intérêt d'une telle approche est qu'elle ne nécessite pas de coupler le système de prédictions à un module de maintien de vérité du type ATMS [KLE87]. Le mécanisme de maintien utilisé ici, est un traitement interne géré par l'intermédiaire des chiens-de-gardes liés à chaque STC lancée. En fonction de la sécurité attendue sur un procédé, on peut utiliser les prédictions pour arrêter des machines du procédé avant l'obtention du diagnostic final.

Le raisonnement hypothétique a été nécessité par l'aspect réparti de notre application. En effet, dans ce cadre, chaque processeur de diagnostic (chaque noeud du GF) a une vision locale donc partielle d'un problème. Pour identifier précisément la source de la défaillance, il doit échanger ses conclusions avec d'autres processeurs. De fait certaines conclusions du diagnostic local seront forcément des hypothèses.

L'approche répartie du diagnostic tient essentiellement à notre volonté de coller avec les innovations technologiques de ces dernières années. Ces innovations concernent :

- L'étude et le développement de capteurs et actionneurs intelligents [STA91/a], [STA91/b]. En effet, de telles entités peuvent être utilisés par exemple pour mettre en œuvre les mécanismes de noeuds initiateurs ou ceux des noeuds de validation directe (en particulier la classe NoeudSurveillance).
- L'apparition de processeurs numériques de plus en plus puissants sur les machines à commande numérique (M.O.C.N). Ces processeurs pourraient traiter les défaillances locales à

la machine avant de transmettre leurs conclusions à un processeur hiérarchiquement supérieur (processeur de poste, processeur de cellule, processeur d'atelier).

Mais nos travaux vont surtout dans le sens du développement de processeurs symboliques spécialisés.

- L'apparition de réseaux de terrains et d'ateliers (FIP, MiniMAP, MAP) de plus en plus rapides permettant le transfert d'informations entre capteurs et processeurs.

Notons toutefois que dans le cadre de l'implantation, l'aspect réparti du système de diagnostic posera un problème d'"asynchronisme". En effet, chaque processeur ayant sa propre horloge, comment intégrer dans un raisonnement temporel global, des données datées par des horloges distinctes ? Dans l'approche proposée, de nombreux raisonnements locaux à chaque processeur sont initiés par les données reçues ou non, en provenances d'autres processeurs. Comment intégrer dans le raisonnement le fait que certains processeurs ou des moyens de communications peuvent être défaillants ? Ces problèmes font parties des difficultés majeures rencontrées par toutes applications réparties. On trouvera une synthèse des solutions actuelles dans [BRE91]

CHAPITRE IV

SPECIFICATION FONCTIONNELLE DE LA COMMANDE D'UN ATELIER FLEXIBLE, DANS LE CADRE DE L'INTEGRATION DES MECANISMES DE RECOUVREMENT DES DEFAILLANCES DU PROCEDE

SOMMAIRE

- I - Le recouvrement : principe et classification,
- II - Gestion des modes de marches et d'arrêts des SFP [BOI91].,
- III - Recouvrement au niveau commande : Exigence d'une approche fonctionnelle,
- IV - Structuration en couches de la commande
- V - Intégration du recouvrement dans l'architecture de commande

INTRODUCTION

Cette partie a pour objet, le recouvrement des défaillances du procédé. Dans ce contexte, le concepteur doit être guidé par deux contraintes :

- 1 - assurer la sécurité du procédé et de son environnement,
- 2 - maintenir le niveau de production.

Pour assurer ces deux contraintes, un certain nombre de mécanismes doivent être intégrés à la commande : il s'agit du **gel**, de la **reprise**, de la **reconfiguration dynamique** [SAH87], [SAH92], [CRA89].

En fait, dans notre approche, le gel est implicite. En effet, la Partie Commande (PC) n'évolue pas directement en fonction des comptes rendus du procédé. Ceux-ci sont filtrés par le module de détection avant d'être émis vers la PC. Lorsqu'un compte rendu n'est pas conforme au comportement modélisé dans le module de détection, un symptôme est détecté et aucun compte rendu n'est envoyé à la PC.

Dans cette partie, nous allons proposer un certain nombre de solutions pour gérer la reprise et surtout la reconfiguration dynamique. Ces deux mécanismes sont à rattacher au problème de maintien du niveau de production malgré l'occurrence de défaillances dans le procédé. Ce maintien passe par une utilisation de la flexibilité de l'ensemble des composantes d'un Système Flexible de Production (SFP).

Dans la première partie de ce chapitre, nous effectuerons une classification des différentes formes du recouvrement. La deuxième partie sera consacrée à la présentation du gestionnaire des modes de marches et d'arrêts des moyens de production d'un SFP [BOI91]. Il a été conçu dans le cadre de la méthodologie CASPAIM-1, en tant qu'un outil de maintien de la production. Il comporte donc une fonction de recouvrement. Dans la troisième partie, nous montrerons l'exigence d'une approche fonctionnelle pour l'intégration de mécanismes de recouvrement au niveau commande. La quatrième partie sera consacrée à définir notre proposition de structuration en couches de la commande des SFP. Dans un premier temps nous présenterons les différents types de flexibilité dans ces systèmes. A partir de cette classification nous proposerons une décomposition de la commande plus apte à gérer les différents modes de marche dégradés. Cela nous amènera à proposer des modifications par rapport à la conception de la commande tel qu'elle a été spécifiée par D. Cruette [CRU91]. Par analogie au travail de D. Cruette, nous illustrerons nos propos par l'exemple de la commande de deux ascenseurs. Dans la cinquième partie, nous proposerons une technique de pilotage adaptée à la structuration en

couches de la commande, et permettant une gestion "transparente" des modes de marches dégradés.

I. LE RECOUVREMENT : PRINCIPE ET CLASSIFICATION.

Au chapitre I, nous avons présenté le recouvrement comme étant une fonction de surveillance faisant suite aux étapes de détection et de diagnostic. Notre but dans cette partie est de présenter le type d'actions menées par le recouvrement, et d'en analyser l'impact sur l'architecture de commande-surveillance d'un SA. Nous allons commencer par définir le recouvrement dans le cadre des SFP.

Définition IV.1 : Le recouvrement.

C'est un ensemble ordonné d'actions correctives, modifiant le comportement du procédé et de la commande, de manière à assurer la sécurité et le maintien du taux de production.

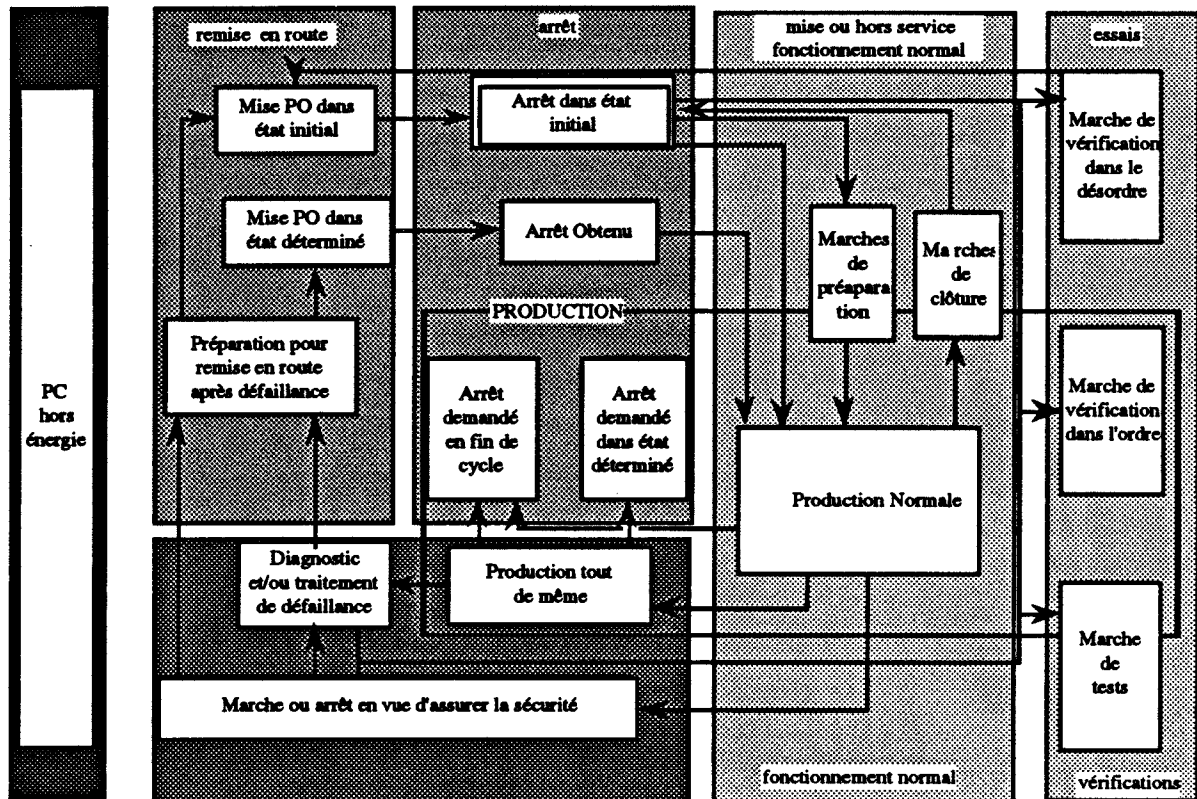
Remarque :

Dans la définition précédente, le terme "maintien" est utilisé pour traduire deux idées. D'une part, le fait que malgré la dégradation du système, on va essayer d'atteindre une production proche du taux planifié, et d'autre part, le fait que dès que possible, il y aura retour du système à un niveau de production normal.

I.1 Présentation du GEMMA [BOU87].

Le GEMMA, Guide d'Etude des Modes de Marches et d'Arrêts est un ensemble de spécifications proposées par l'ADEPA dans le cadre de la gestion des modes de marche d'un SA.

Il distingue trois familles de procédures réparties en cinq groupes (Fig. IV.1) :



LEGENDE

- PROCEDURES EN DEFAILLANCES
- PROCEDURES D'ARRET et DE REMISE EN ROUTE
- PROCEDURES DE FONCTIONNEMENT

Figure IV.1 : Le GEMMA.

I.1.1 Les procédures de fonctionnement.

Elles concernent deux groupes de procédure :

* Les procédures de vérification et d'essais.

C'est l'ensemble des vérifications à mener sur le système avant son intégration ou sa réintégration dans la production. Les marches de test permettent le réglage ou l'étalonnage définitif de la ressource.

*** Les procédures de fonctionnement normal.**

On y distingue trois types de procédure. Par ordre normal de déclenchement, ce sont :

- les **marches de préparation** qui permettent de configurer les équipements dans l'état nécessaire au fonctionnement normal (préchauffage de l'outillage, ...),
- la **production normale**,
- les **marches de clôture** qui consisteront par exemple sur une M.O.C.N., à évacuer les copeaux et à lubrifier certains organes mécaniques arrosés par le liquide de refroidissement.

I.1.2 Les procédures d'arrêt et de remise en route.

*** Les procédures d'arrêt.**

En fonctionnement normal, l'arrêt s'effectue dans l'état initial après la marche de clôture. En cas d'occurrence d'une défaillance, l'arrêt peut se faire dans un état déterminé (AED) ou en fin de cycle (AFC).

*** Les procédures de remise en route.**

Une remise en route peut se faire dans deux contextes :

- Soit après un arrêt normal dans l'état initial ou un arrêt d'urgence.

Elle consiste alors à **remettre la Partie Opérative dans son état initial** après l'exécution de marches de vérification dans le désordre, ou la réparation du système. Dans tous les cas, on considère alors qu'on aborde un nouveau cycle de production.

- Soit après un arrêt momentané suite à une défaillance.

Cette fois, on considère que l'on reste dans le cycle de production dans lequel est apparu la défaillance. Après traitement de la défaillance, la production est reprise là où le cycle avait été suspendu. Aussi, la remise en route consistera à **mettre le système dans l'état dans lequel il était avant l'arrêt.**

I.1.3 Les procédures suite à une défaillance.

En fonction de l'ampleur de la panne, **la production est poursuivie** (défaillance significative) jusqu'à l'arrêt dans un état déterminé. **Le système est alors mis en position de repli afin d'assurer la sécurité.** Dans tous les cas, une phase de diagnostic devra ensuite établir la cause de la défaillance. Elle sera ensuite traitée avant la remise en route.

Le GEMMA se contente de spécifier les fonctionnalités indispensables dans le contexte de la surveillance. La mise en œuvre effective est laissée à la discrétion de chaque concepteur. Le gestionnaire des modes de marche conçu dans le cadre de la méthodologie CASPAIM-1 est un exemple de mise en œuvre du GEMMA.

I.2 Les différents aspects du recouvrement.

Toute procédure de recouvrement peut être caractérisée par cinq paramètres :

*** Le but :**

Au chapitre I, nous avons introduit la surveillance en disant qu'elle a pour objectifs d'augmenter la sécurité et la disponibilité dans un SA. De ce fait, toute action de recouvrement aura pour but d'assurer l'un de ces objectifs.

*** Les actions :**

Il s'agit de l'ensemble des actions constituant une procédure de recouvrement. On distinguera les actions menées sur le procédé de celles concernant la commande.

*** La réactivité :**

Elle a pour but de caractériser le **temps écoulé admissible** entre la détection d'une défaillance et le début du recouvrement. Ce temps est fonction de l'ampleur de la défaillance et des buts privilégiés par la procédure de recouvrement.

*** L'étendue :**

Elle permet de spécifier l'impact de la procédure de recouvrement sur le système. Cet impact peut être local à l'entité défaillante ou globale (sur le système auquel elle appartient).

*** La durée.**

Ici, il s'agit de mesurer le temps pendant lequel le système va fonctionner en mode dégradé.

A partir de ces paramètres, nous établirons des classes de procédures de recouvrement .

I.2.1 Classification en fonction des moyens d'actions.

Dans ce paragraphe, nous allons essayer de préciser les directives du GEMMA par des moyens d'actions pratiques sur le procédé et sur la commande.

I.2.1.1 Moyens d'actions sur le procédé.

Les spécifications du GEMMA montrent que ces procédures sont fonction de la nature de la défaillance (significative ou critique). En effet, un dysfonctionnement peut conduire aux états : production tout de même, ou marche en vue d'assurer la sécurité (cf. figure IV.1). Nous allons préciser le contexte et les conséquences du passage dans l'un ou l'autre de ces états.

La poursuite de la production en mode de fonctionnement dégradé ne peut s'envisager que dans le contexte d'une défaillance significative : c'est à dire soit une défaillance progressive, soit une défaillance catalectique d'un composant dont la fonction peut être remplacée par celle d'un autre composant. Pour illustrer ce dernier cas, prenons l'exemple d'une machine à deux tampons d'entrée-sortie. La panne d'un tampon peut être compensée par l'activité de l'autre tampon. Dans le cas d'une défaillance progressive, une marche en vue d'assurer la sécurité peut consister en la réduction des paramètres de l'opération en cours. Considérons le cas de la surveillance d'outils de coupe, en cours d'usinage. Supposons que le système de surveillance ait détecté que le seuil d'usure de l'outil (seuil d'alarme) est atteint (Fig. IV.2). Pour éviter la rupture de l'outil, ou tout au moins l'arrêt d'urgence, il faut réduire les paramètres du processus de coupe. Ainsi, si la qualité de la pièce à usiner n'est pas trop contraignante, on peut poursuivre l'usinage avec le même outil, en réduisant par exemple, la vitesse d'avance et l'épaisseur des passes (voir la courbe d'évolution en mode dégradé à partir du seuil d'alarme ; Fig IV.2).

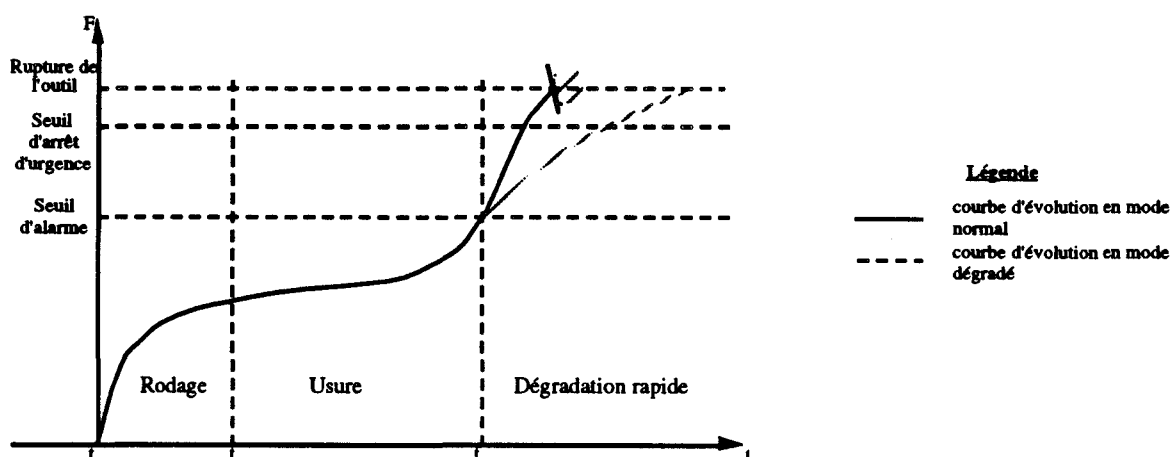


Figure IV.2 : Evolution des efforts de coupe en fonction du degré d'usure d'un outil.

Dans le cas d'une défaillance critique, les marches de sécurité conduisent à court terme à un arrêt d'urgence.

Dans tous les cas de figure, le recouvrement impose un arrêt de la ressource à moyen terme. Les trois principaux arrêts utilisés sont :

- l'**Arrêt en Fin de Cycle (AFC)**,
- l'**Arrêt dans un état Déterminé (AED)** ; l'arrêt des avances (AA) sur une M.O.C.N.en est un exemple type,
- l'**Arrêt d'Urgence (AU)**.

L'arrêt d'urgence est le pire des arrêts du point de vue de la disponibilité. En effet, il met la machine hors énergie ce qui nécessite ensuite de repasser par toutes les étapes de vérification et de préparation. Malheureusement, l'absence d'intégration d'un véritable outil de diagnostic entraîne, pour des raisons de sécurité, une utilisation fréquente de ce type de procédure de recouvrement (par les constructeurs de M.O.C.N).

I.2.1.2 Mécanismes d'actions sur la commande.

Ils sont répartis en trois familles [SAH87], [CRA89], [SAH92] :

- le **gel**,
- la **reprise**,
- la **reconfiguration dynamique**.

Le gel consiste à figer dans l'état certains processus de commande, jusqu'au diagnostic. Lorsque l'indisponibilité de la partie concernée du procédé est de faible durée, on peut envisager une reprise. L'exemple type est celui de la rupture d'un outil. La procédure de recouvrement peut être la suivante :

- 1 - arrêt des avances sur la machine,
- 2 - gel du programme d'usinage,
- 3 - appel d'une procédure de dégagement de l'outil,
- 4 - échange de l'outil avec un autre de la même famille, présent dans le magasin d'outil,
- 5 - reprise de l'usinage au début de bloc programme en cours lors de la rupture.

La reconfiguration dynamique est d'un tout autre genre puisqu'elle modifie la structure des processus de commande, sans pour autant que l'ensemble du procédé soit à l'arrêt. Le problème crucial de ce mécanisme est de faire coïncider le marquage du nouveau graphe de commande avec l'état réel du procédé. Deux approches sont envisageables [TOG91] :

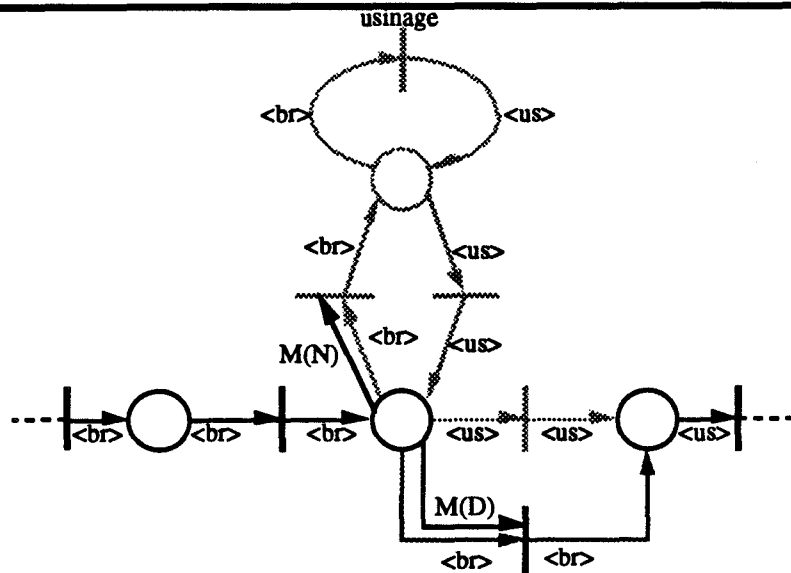
1 - Le téléchargement de programmes adaptés à l'état du procédé.

Cette approche est facilement applicable dans le cas de programmes à déroulement séquentiel, sans indéterminisme, ni parallélisme. La place initiale du programme téléchargé modélise alors l'état courant du procédé. C'est souvent le cas au niveau de la commande locale des machines.

2 - L'adaptation dynamique de la structure de la commande.

Cette modification peut être obtenue par la connexion ou la déconnexion de processus de commande. Elle est donc adaptée aux commandes présentant un parallélisme important. C'est souvent le cas de la commande de coordination des SFP. Ainsi, dans l'approche CASPAIM-1, la reconfiguration dynamique peut être obtenue par la gestion d'arcs adaptatifs ou par l'utilisation de la **notion de domaines de couleurs dynamiques** [TOG91], [BOI91].

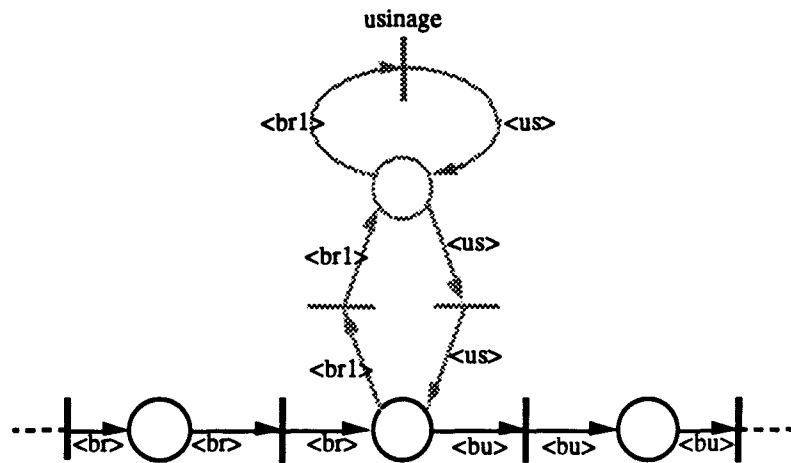
Ces deux solutions ont l'avantage de modéliser au niveau du graphe, toutes les possibilités de commande. Supposons par exemple, que la commande consiste à conduire des pièces "brutes" (
) jusqu'à un poste d'usinage. En sortie de ce poste, les pièces sont usinées (<us>). Selon le mode de fonctionnement, les pièces en sorties du poste de chargement sont des pièces usinées (mode normal) ou un mélange de pièces brutes ou usinées (mode dégradé en cas de panne de la machine). Sur la figure IV.3/a, nous en avons tenu compte en ajoutant une portion de graphe spécifique au mode dégradé. Afin d'éviter une augmentation "artificielle" des indéterminismes du graphe de commande dans un mode de fonctionnement donné, il faut paramétrer le graphe en utilisant des paires places-arcs adaptatifs. Cette solution ne facilite pas la lisibilité du graphe. Nous lui préférons la solution consistant à modifier les domaines de couleurs du graphe initial, en fonction du mode de fonctionnement. Pour la mettre en œuvre, il suffit d'intégrer au Niveau Hiérarchique, des règles du type de celles de la figure IV.3/b.



----- partie du graphe utilisée dans le mode normal
 ——— nouvelle partie ajoutée pour gérer le mode dégradé

Mode normal ○ N ⊙ D
 Mode dégradé ⊙ N ○ D

a) Reconfiguration dynamique par arcs adaptatifs



Règle1 : Si mode=normal alors $\text{dom}(\langle \text{br1} \rangle) := \text{dom}(\langle \text{br} \rangle)$ et $\text{dom}(\langle \text{bu} \rangle) := \text{dom}(\langle \text{us} \rangle)$

Règle2 : Si mode=dégradé alors $\text{dom}(\langle \text{bu} \rangle) := \text{dom}(\langle \text{us} \rangle) \cup \text{dom}(\langle \text{br} \rangle)$ et $\text{dom}(\langle \text{br1} \rangle) := \{\}$

b) Domaines de couleurs dynamiques

Figure IV.3 : Reconfiguration dynamique dans CASPAIM-1

I.2.2 Classification en fonction du but et de la réactivité.

Il s'agit ici de déterminer les conditions temporelles de déclenchement du recouvrement compte tenu du but à atteindre (sécurité ou disponibilité).

I.2.2.1 Recouvrement immédiat.

Pour assurer la sécurité, il faut réagir assez vite après la localisation d'une défaillance, afin d'en empêcher une propagation dangereuse par le procédé. Prenons l'exemple du convoyeur de la figure IV-4. En cas de défaillance de l'aiguillage (cf. chapitre II, § II.1.2), il y a trois possibilités de propagation des défaillances par le procédé par :

- **collision** au niveau de la jonction, entre la palette mal orientée par l'aiguillage et celle arrivant de P4 (cas de la figure IV.4),
- **perturbation du positionnement** d'une palette en attente de dépalettisation sur le poste P3,
- **substitution**, la palette venant de P5 passant la jonction avant celle attendue, en provenance de P4. Le système de commande posséderait alors une image erronée du procédé. L'absence de système de vision sur les manipulateurs pourraient alors entraîner une collision entre le manipulateur et la palette.

Ces défaillances propagées peuvent être évitées par le blocage de la palette égarée par la butée n°2 (butée-2). Cela nécessite un diagnostic rapide (inférieur au temps de transfert de la palette) de la défaillance de l'aiguillage.

De manière générale, toutes les fonctions de la surveillance peuvent contribuer à assurer la sécurité. Au niveau détection, la classification des symptômes en fonction des risques de propagation (notion de symptôme urgent) permet déjà d'agir avant même la phase de diagnostic. Au niveau du diagnostic, dans certains cas, la localisation est suffisante pour engendrer une série d'actions qui permettront de confiner la panne. Au niveau des moyens d'actions sur le procédé, la méthode la plus sûre est l'arrêt momentané des entités soupçonnées, quitte à ce que certaines soient remises en marche après identification de la défaillance.

Les procédures de sécurité font parties du recouvrement immédiat. Aussi, elles doivent être prises en compte par les niveaux les plus bas d'une architecture de commande-surveillance.

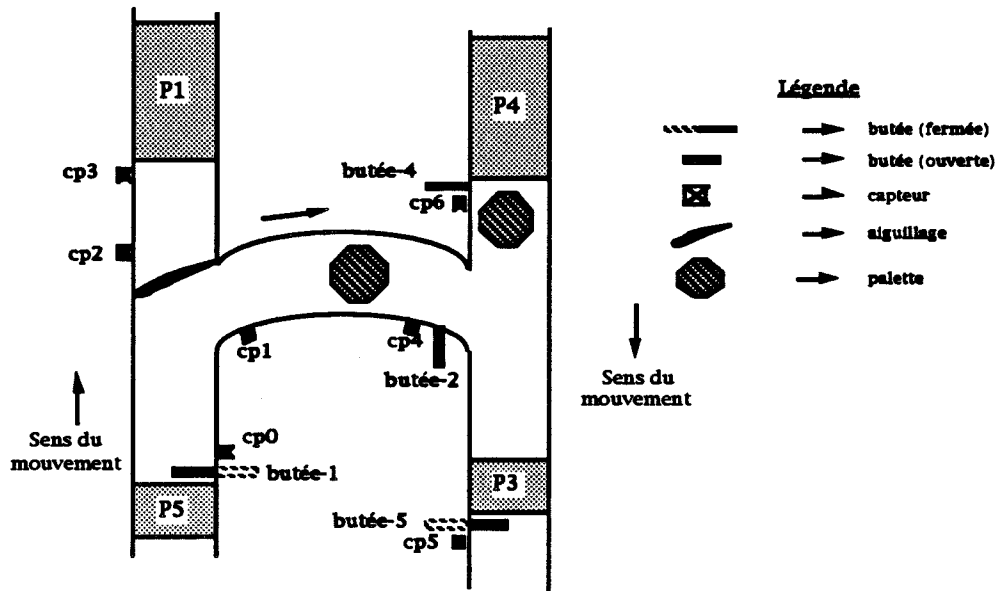


Figure IV.4 : Système de convoyage des pièces.

I.2.2.2 Recouvrement différé.

Ce type de recouvrement est à mener pour le maintien du niveau de production planifié. Posons le problème par l'exemple suivant. Supposons que des pièces soient en attente devant un poste de travail défaillant. Que faut-il faire de ces pièces ? Plusieurs solutions sont envisageables :

1 - Laisser les pièces en attente jusqu'à la réparation de l'entité défaillante.

Cette solution n'est possible que si la durée d'indisponibilité est faible par rapport au temps de transfert des pièces sur un autre poste de production. D'un point de vue pratique, sa mise en œuvre exige une **remise en état du poste défaillant**, puis une **reprise** de l'opération en cours lors de l'arrêt du poste. L'ensemble de ces actions correctives correspond à un recouvrement à moyen terme.

2 - Transférer les pièces vers un autre poste de production.

La réparation de l'unité défaillante est menée en parallèle. Les autres unités dont le fonctionnement est lié à celui de l'unité défaillante sont arrêtées. L'ensemble est remis en marche après la réparation de l'unité défaillante. Ce recouvrement à moyen, voir long terme, nécessite une **reconfiguration de la commande** et parfois une **révision du plan de fabrication** en cours.

La deuxième solution est liée à une caractéristique essentielle des ateliers flexibles : la flexibilité. En effet, sa mise en œuvre nécessite :

- la disponibilité de machines polyvalentes travaillant à un taux de charge inférieur au taux maximal [TOG91] ;
- la disponibilité d'un système de transport flexible, permettant le transfert des pièces vers les postes sains,
- une flexibilité de la commande permettant de l'adapter à tout instant à l'état réel du procédé.

Le recouvrement différé est basé sur l'évaluation de la durée d'indisponibilité, de l'entité défaillante. Aussi, il ne peut se faire qu'après l'identification de la cause de la défaillance (Fig. IV.5),, information nécessaire au calcul de la date de remise en route. Cette date peut être approximée par la formule suivante¹ :

$$\text{date-de-remise-en-route} = \text{date-détection} + \text{MDT} \quad (\text{IV.1})$$

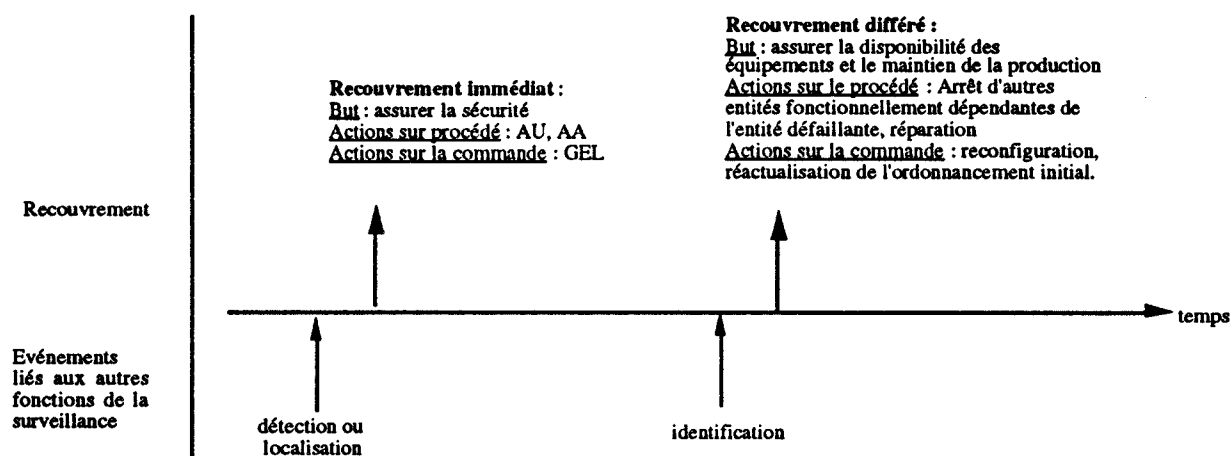


Figure IV.5 : Réactivité du recouvrement en fonction du but.

I.2.3 Classification en fonction de l'étendue et de la durée.

L'étendue et la durée sont des paramètres qui mesurent les conséquences d'une procédure de recouvrement, d'un point de vue spatial et d'un point de vue temporel sur le système.

L'étendue peut être locale à l'entité défaillante ou globale au système. Au niveau de la durée, nous en distinguerons 3 termes : le court terme, le moyen terme ou le long terme.

¹ Le MDT correspond à la durée moyenne d'indisponibilité de l'entité considérée (cf. annexe A)

I.2.3.1 Recouvrement local à court terme.

Ce type de recouvrement correspond à un traitement local de faible durée. Le système retrouve rapidement son mode de fonctionnement normal. C'est le type de recouvrement que l'on mènera dans le contexte d'une défaillance progressive.

I.2.3.2 Recouvrement global à moyen terme.

Ce type de recouvrement à un impact de moyenne durée sur l'ensemble du système. Il correspond à une indisponibilité de faible durée, d'une ressource dont la charge peut être momentanément assurée par d'autres. C'est par exemple le cas lors d'un arrêt d'urgence n'occasionnant pas de réparation. Au niveau commande, le recouvrement se traduira par une reconfiguration.

I.2.3.3 Recouvrement global à long terme.

Ce type de recouvrement fait suite à la défaillance d'une ressource critique ou à la défaillance d'une ressource dont l'indisponibilité sera importante. Il peut nécessiter une modification de l'ordonnancement en cours, voir même la révision du plan de production. Au niveau coordination, ce type de recouvrement a pour conséquence la reconfiguration du système de commande.

I.2.4 Conclusion

A partir des paramètres précédents nous établissons trois classes de procédures de recouvrement :

*** Classe A.**

- But : assurer la sécurité,
- Réactivité : immédiatement après la détection,
- Actions phase 1 : AU ou Arrêt des mouvements, gel de la commande
- Actions phase 2 : arrêts d'autres entités fonctionnellement dépendantes, réparation, reprise ou reconfiguration
- Etendue : locale (reprise) ou globale (reconfiguration)
- Durée : moyenne (reprise) ou longue (reconfiguration).

Dans cette classe, on privilégie l'aspect sécurité. Aussi le recouvrement (phase 1) doit être immédiat et conduit rapidement à l'indisponibilité d'un certain nombre d'entités. En fonction de la durée d'indisponibilité, l'étendue peut être locale et la durée moyenne (Classe A1), ou l'étendue peut être globale et la durée importante (Classe A2).

*** Classe B.**

- But : assurer la sécurité et le maintien de la production.
- Réactivité : après la localisation,
- Actions : réduction des paramètres d'exploitation, AED ou AFC, reprise
- Etendue : locale
- Durée : courte ou moyenne

Ici on recherche un compromis entre sécurité et maintien de la production. Dans cette classe, on distinguera deux familles de procédures. Certaines consisteront à poursuivre la production à un niveau de performances dégradé. D'autres consisteront à minimiser le temps d'arrêt effectif dû au recouvrement. Dans tous les cas, cette forme de recouvrement ne peut être que locale et de faible durée.

*** Classe C.**

- But : maintien de la production.
- Réactivité : après identification de la cause de la défaillance,
- Actions : AFC, reconfiguration dynamique
- Etendue : globale
- Durée : longue

Cette fois-ci, on cherche à privilégier le maintien de la production. Aussi, la décision d'arrêter l'entité ne sera prise qu'une fois la cause de la défaillance identifiée. Pendant ce temps, on peut prévoir les conséquences d'un arrêt et planifier la reconfiguration totale du système. Dans ce contexte, l'étendue est globale et la durée importante.

II. GESTION DES MODES DE MARCHES ET D'ARRETS DES SFP [BOI91].

L'idée directrice des travaux de S. Bois est qu'il existe des liens de coopérations (**Notion de contrainte opérationnelle**) entre moyens de production connexes. C'est liens entraînent des dépendances entre les modes de marches et d'arrêts de ces moyens.

II.1 Modélisation du comportement des moyens de production.

On distingue deux types de machines :

- les **machines effectives**,
- les **machines virtuelles**.

On appelle machine effective toute entité matérielle commandable : butées, aiguillages, convoyeur, machines outils, robots ... Le comportement de chaque machine est représenté par un graphe d'état. Dans le cas le plus complexe (M.O.C.N.), le graphe comporte six états :

* **Etat 0 ou Tension-Off.**

L'entité est hors énergie. C'est l'état initial. Graphiquement il est représenté par un double cercle.

* **Etat 1 ou Tension-On.**

Il correspond à une mise sous tension de l'entité en dehors de toute autre commande.

* **Etat 2 ou Initialisé.**

L'entité a été configurée (vérification, marche de préparation, etc ...) et est prête à passer en marche normale.

* **Etat 4 ou Marche Normale.**

C'est le mode de fonctionnement normal de l'entité.

* **Etat 5 ou Marche Dégradé.**

Une défaillance a été détectée. On poursuit la production en mode dégradé.

* **Etat 3 ou Arrêt sous-défaut.**

contraintes de fonctionnement. Celles-ci présentent des caractères impératifs qui rendent les analyses de comportement de chaque sous-machine complètement indissociable les unes des autres.

Une machine virtuelle étant composée de sous-machines, elle en hérite les comportements. L'idée directrice du regroupement de machines effectives en machines virtuelles est de faire coïncider les états des sous-machines composant une machine virtuelle. De ce fait, la structure de base du graphe de comportement d'une machine virtuelle est la même que celle des machines effectives. Pour tenir compte des modifications pouvant survenir individuellement sur les sous-machines, on est obligé d'introduire dans le comportement de base, des états transitoires appelés **états instables**.

Définition IV.3 : Etat instable [BOI91].

Une machine virtuelle se trouve dans un état dit "instable", si une, au moins, de ses sous-machines se trouve dans un état incompatible avec celui des autres sous-machines ou lorsque la conformité des états de ses sous-machines n'a pas été vérifiée. Il s'agit alors d'effectuer un test de cohérence des états dépendants.

Un état instable est en fait un état transitoire entre deux états stables, i.e du comportement de base (Fig. IV.7).

Le principe de la modélisation pour la gestion des modes de marches et d'arrêts des moyens de production est d'intégrer toutes les machines dans un regroupement. La modélisation est alors menée selon une approche ascendante basée sur l'exploitation des contraintes opérationnelles. On distingue deux types de contraintes :

* Les relations sur états :

- contrainte de coopération (cf. figure IV.10),
- contrainte de coopération partagée,
- contrainte d'exclusion,
- contrainte structurelle série ou parallèle.

* Les relations sur changements d'états.

- contrainte procédurale,
- contrainte d'observabilité.

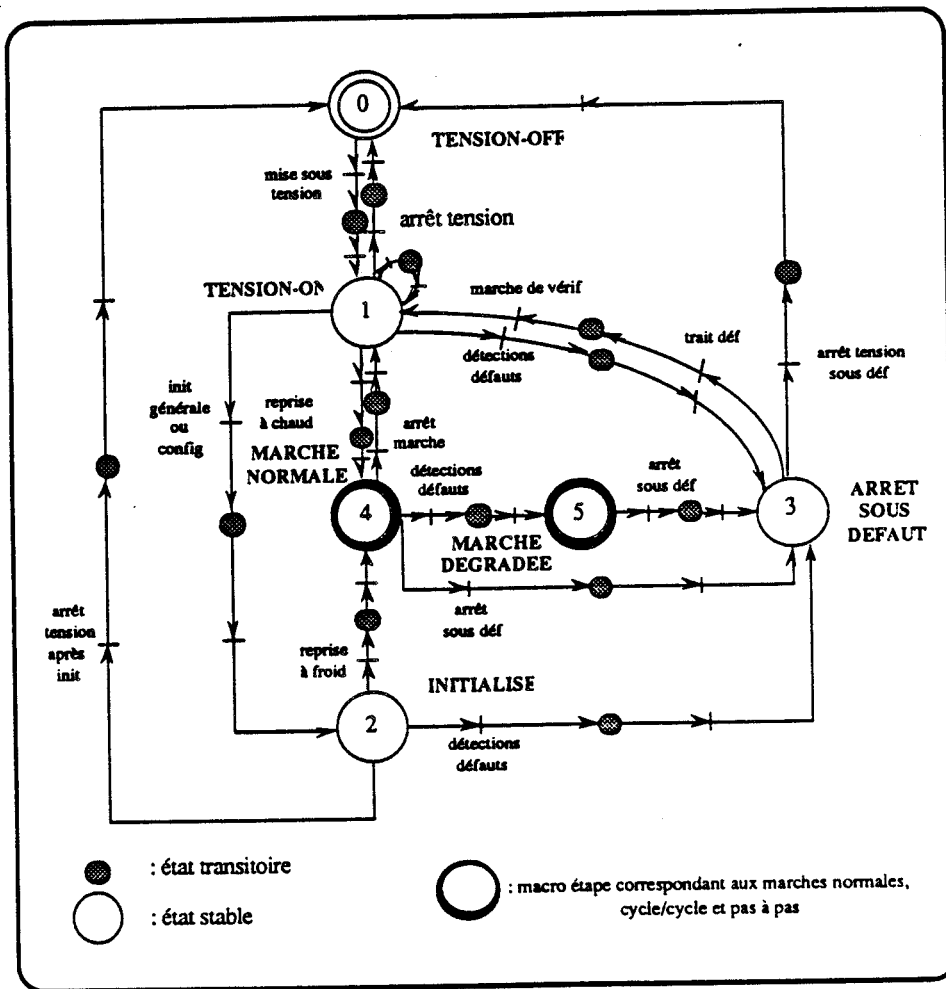


Figure IV.7 : Graphe de comportement d'une machine virtuelle.

Au terme de l'analyse, on obtient une représentation arborescente (Fig. IV.8) de l'atelier permettant d'évaluer dynamiquement le comportement d'une machine, à partir de la connaissance de l'état de toutes ses sous-machines. Cette représentation est particulièrement adaptée à l'analyse de la cohérence des états, à la détection des états incohérents et donc à la propagation des états courants selon une stratégie "bottom-up".

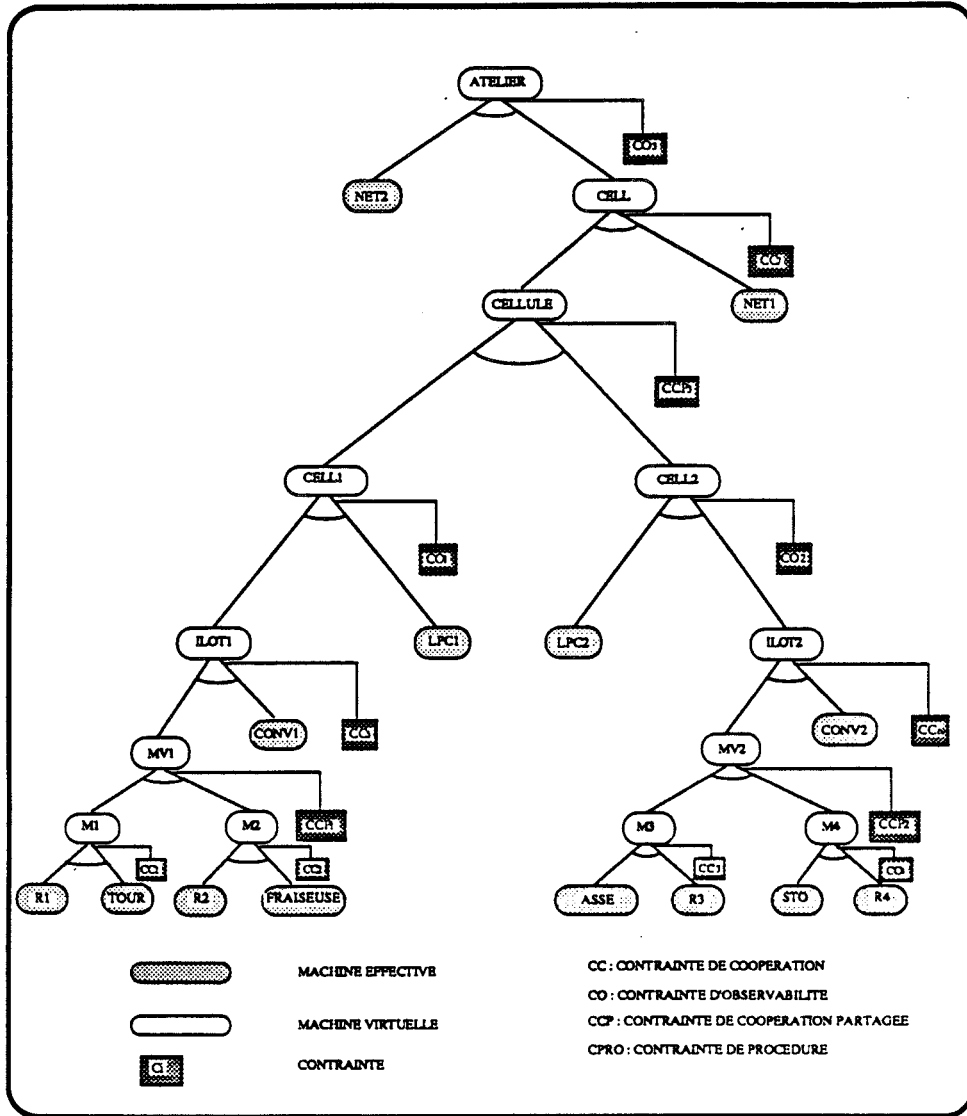


Figure IV.8 : Exemple de représentation finale d'un atelier d'après [BOI91].

II.2 Structuration et fonctionnement du gestionnaire.

Dans son approche, S. Bois confère au gestionnaire une fonction de pilotage. De ce fait, il sert non seulement à la gestion des modes de marches, mais également à la coordination des différentes fonctions du Niveau Hiérarchique. Ces fonctions sont (Fig.IV.9) :

- la surveillance,
- la coordination des automates qui a pour but essentiel la résolution des indéterminismes du graphe de coordination [CRA89],
- l'interface utilisateur qui permet à l'opérateur de paramétrer le fonctionnement de l'atelier ou de le gérer manuellement.

Le fonctionnement du gestionnaire se déroule selon trois phases (Fig. IV.11) :

- une phase de génération d'ordres également appelés buts,
- une phase de résolution des buts,
- une phase de correction.

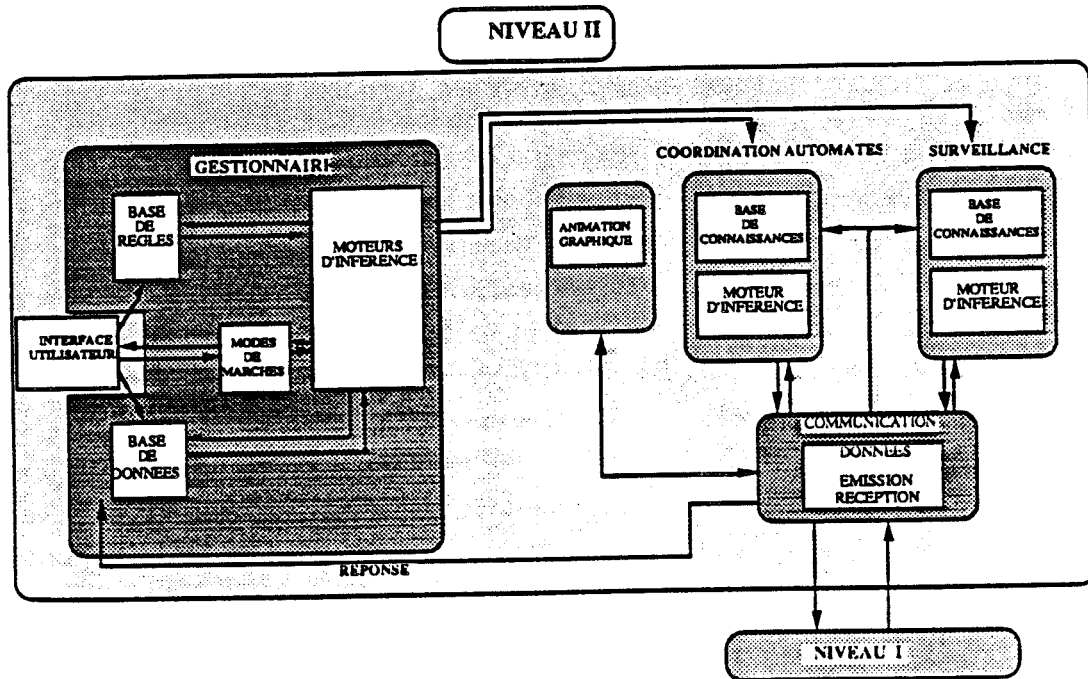


Figure IV.9 : Structuration du Niveau Hiérarchique et du gestionnaire des modes de marche d'après [BOI91].

Définition IV.4 : Un ordre [BOI91].

C'est un changement direct de mode de marche d'une machine ;

La phase de génération d'ordres a deux modes d'initialisation. Le premier correspond à un mode manuel à l'initiative de l'opérateur. Il peut décider l'arrêt d'une ou plusieurs machines dont le fonctionnement ne lui semble pas normal, ou à des fins de maintenance préventive.

Le deuxième mode est initié automatiquement par toute modification de la base de données dynamiques du procédé. Elle est automatiquement mise à jour par scrutation du procédé. Pour chaque machine, les données dynamiques modélisent :

- l'état antérieur de la machine,
- l'état calculé ou état effectif de la machine.

Lorsque une différence est détectée entre l'état effectif et l'état antérieur d'une machine, cela conduit à recalculer l'état de la machine virtuelle dont elle est une sous-machine. Ce calcul

se fait en tenant compte des contraintes de coopération fonctionnelle. Il s'en suit une propagation ascendante dans l'ensemble du modèle pour connaître la répercussion aux différents niveaux de la hiérarchie. A chaque niveau autre que celui de la machine effective initiatrice de la résolution, toute différence détectée conduit à la génération d'ordres visant à rendre cohérent l'état des sous-machines avec celui de la machine virtuelle concernée. Supposons par exemple qu'un TOUR et un ROBOT soient regroupés au sein d'une machine virtuelle MV1 par une contrainte de coopération (Fig. IV.10). L'arrêt sous défaut de l'une ou l'autre de ses deux machines effectives va conduire à la génération du but ? **arrêt-direct-sous-déf** de MV1.

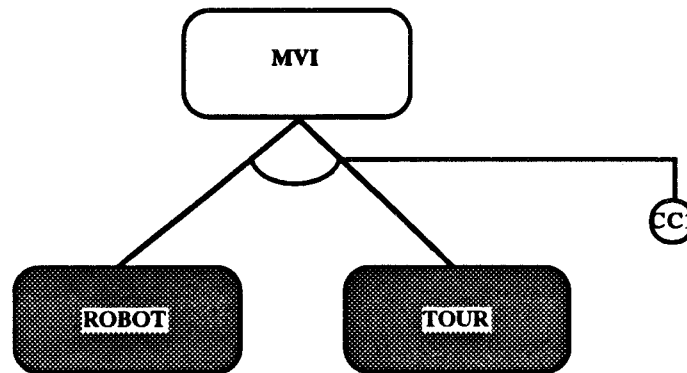


Figure IV.10 : Contrainte de coopération.

La phase de résolution des buts a pour objet de spécifier les différentes actions à engendrer pour atteindre un but donné. De fait, plusieurs buts concurrents ou complémentaires peuvent avoir été générés précédemment. Cette phase va alors établir une classification des buts par ordre de priorité. La résolution ici est du type génération de plan. Dans l'exemple de la figure IV.10, la résolution aboutira à émettre les sous-buts ? **arrêt-direct-sous-déf** de la machine défaillante et ? **arrêt-tension-on** de l'autre machine. Ensuite, à chaque but mettant en jeu une machine effective, la résolution associera une action.

Une action met en jeu en parallèle, un recouvrement au niveau du procédé (arrêt effectif d'une machine) et un recouvrement au niveau de la commande. Le recouvrement au niveau de la commande consiste à établir un nouveau routage des pièces en tenant compte du nouvel état du système. Le nouveau routage est obtenu en utilisant la technique des domaines de couleurs dynamiques.

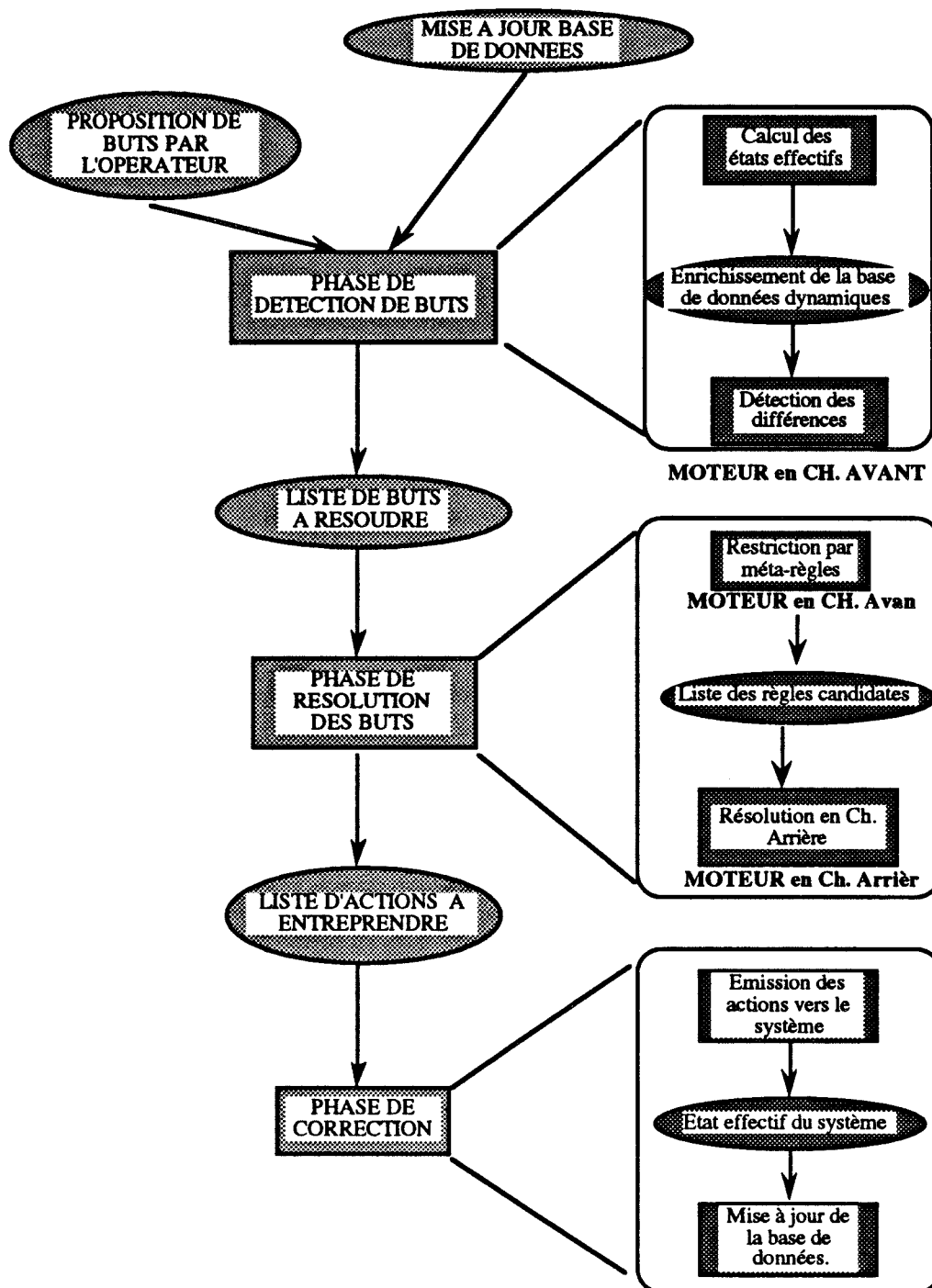


Figure IV.11 : Organigramme de fonctionnement du gestionnaire adapté d'après [BOI91].

II.3 Conclusion.

Le travail de S. Bois est une approche originale qui propose une opportunité intéressante de réalisation des spécifications décrites par le GEMMA. En outre, elle est adaptée au contexte

des ateliers flexibles. Notre but est d'utiliser le gestionnaire comme un outil de maintenance. Son rôle sera de gérer l'arrêt et la remise en marche des ressources à l'initiative de la maintenance ou de la surveillance (cf. figure IV.29). L'intégration de ces deux fonctions au fonctionnement du gestionnaire se fera par l'intermédiaire de la base des données dynamiques. Elle consistera à modifier directement des états des machines effectives.

Dans l'utilisation que nous souhaitons, le gestionnaire ne modifiera plus directement la structure du séquentiel de commande. Les actions sur la commande consisteront plutôt en la modification des paramètres de pilotage. Nous reviendrons sur cette question au paragraphe V.2.

III. RECOUVREMENT AU NIVEAU COMMANDE : EXIGENCE D'UNE APPROCHE FONCTIONNELLE

III.1 Flexibilité et gestion des modes de marche.

Nous allons mettre en relation ces concepts à partir de l'exemple suivant. Supposons que les étages d'un bâtiment soient desservis par deux ascenseurs : A1 (Ascenseur positionné à gauche) et A2 (Ascenseur positionné à droite) (Fig. IV.12).

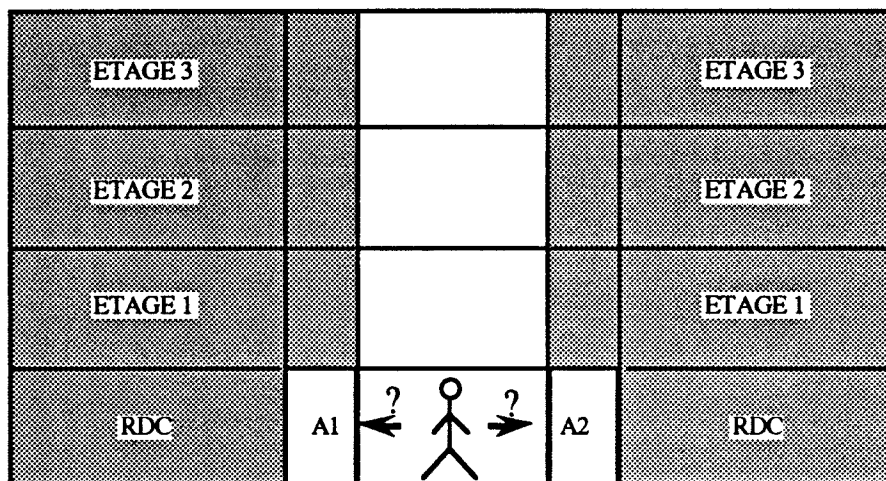


Figure IV.12

La commande en est très simple. Elle consiste :

- 1 - à déplacer un des ascenseurs vers l'étage d'appel,
- 2 - à transférer l'ascenseur appelé vers l'étage de destination.

Analysons l'influence de la flexibilité du procédé sur la gestion des modes de marches dégradés. Pour cela envisageons les deux configurations suivantes :

*** Procédé-1**

- A1 dessert les étages impairs et le rez-de-chaussée (RC),
- A2 dessert les étages pairs et le RC,
- Il n'y a pas d'escalier.

Si l'un des ascenseurs est en panne, il n'y a aucune solution de rechange pour les usagers des étages qu'il dessert. En conséquence avec ce procédé, on ne peut gérer aucun mode dégradé.

*** Procédé-2**

Les deux ascenseurs desservent tous les étages (RC compris). Par ailleurs on dispose d'un escalier.

Si l'un des ascenseurs est en panne, l'autre assure toutes les requêtes des usagers. En cas de pannes des deux ascenseurs, les usagers peuvent encore emprunter l'escalier.

Proposition IV.1 :

Dans un système de production il y a d'autant plus de modes de marches dégradés que la flexibilité matérielle est importante.

III.2 Gestion des modes de marches et pilotage.

Analysons l'influence de la politique de pilotage sur la gestion des modes de marches. Envisageons les deux politiques suivantes :

*** Politique P1**

La requête d'un usager (appui sur le bouton d'appel) est exécutée par l'ascenseur appelé.

L'ascenseur appelé correspond toujours à l'ascenseur positionné du même côté (droit ou gauche) que le bouton d'appel.

*** Politique P2**

La requête d'un usager est exécutée par l'ascenseur disponible, le plus proche de l'étage d'appel, sachant que :

- 1 - Lors de l'exécution d'une requête, s'il est libre, le deuxième ascenseur se positionne au RC. Cela permet de tenir compte du fait qu'en moyenne, il y a plus de personnes désirant aller du RC vers un des étages supérieurs.
- 2 - Après exécution d'une requête, chaque ascenseur reste dans sa position courante.

Le choix de l'une ou l'autre de ces deux politiques va influencer sur la facilité et la qualité de la gestion des modes de marches dégradés.

Si on applique la politique P1 au procédé n°2, le passage en mode dégradé est à l'initiative de chaque usager. En effet, se rendant compte que l'ascenseur appelé n'arrive pas, il devra décider d'appeler le second. Cette démarche entraîne une perte de temps importante due à la dissociation des fonctions pilotage et recouvrement.

Si on applique la politique P2 au procédé n°2, le passage en mode dégradé est transparent pour les usagers. En effet, en cas de panne de l'un des ascenseurs, l'autre assurera le service à chaque appel. Le système de commande force le basculement en mode de marche dégradé dès la détection de la panne. La seule conséquence pour les usagers est la perception d'un temps d'attente plus long en moyenne.

Cet exemple montre qu'il y a une inter-dépendance entre la gestion des modes de marches dégradés et le pilotage. Il montre par ailleurs que l'intégration de la gestion des modes de marches à la fonction pilotage permet de réduire le temps de commutation entre modes. Nous dirons qu'on a une gestion transparente des modes de marches si leur gestion est assurée par la fonction pilotage.

Quels critères permettent de reconnaître formellement qu'une politique de pilotage sera adaptée à une gestion automatique et transparente des modes de marches ?

Formalisons les modes de marches normaux et dégradés par rapport à chacune des politiques de pilotage précédente.

Dans ce qui suit, A_i signifie que l'ascenseur n°i est en état de marche.

Politique P1

Dans cette politique, le mode normal consiste à voir arriver l'ascenseur appelé. Dans ce contexte, les deux ascenseurs doivent être simultanément disponibles.

Mode de marche normal : $A_1 \wedge A_2$

Politique P2

Dans cette politique, l'usager prend l'ascenseur qui arrive. Le mode normal correspond donc à la disponibilité d'au moins un des deux ascenseurs.

Mode de marche normal : $A1 \vee A2 = (A1 \oplus A2) \vee A1 \wedge A2$

Mode dégradé : $A1 \oplus A2$

On note que le mode de marche dégradé énuméré dans la politique n°2 est un sous-mode du mode de marche normal.

Règle IV.1 : Condition suffisante pour une gestion transparente des modes de marches.

Les modes de marche dégradés doivent être des sous-modes du mode de marche normal (Fig. IV.13).

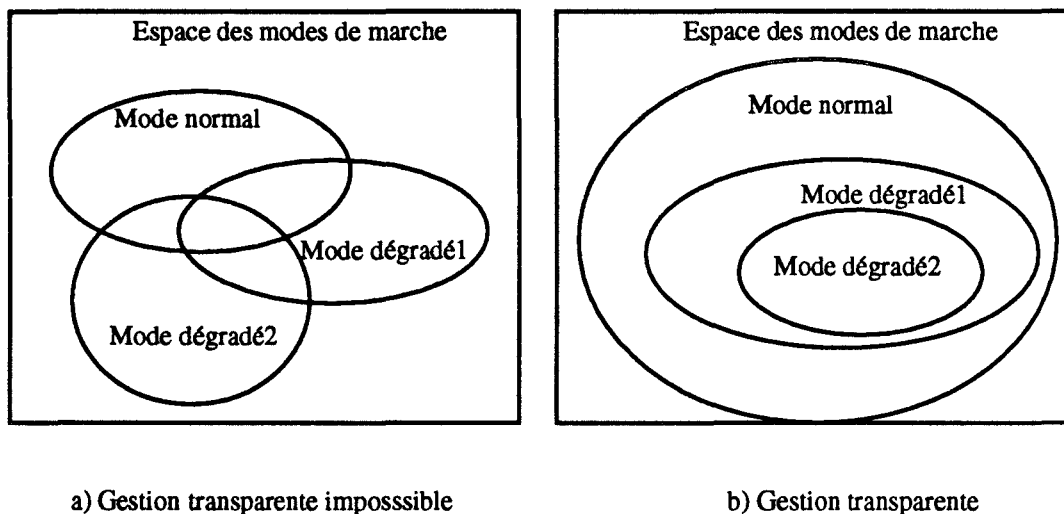


Figure IV.13

Cette proposition à une conséquence directe sur la conception de la commande. Dans CASPAIM-1 [BOU88], la commande de coordination est un ensemble de ressources logicielles coopérantes. Le passage en mode dégradé correspond à une réduction des ressources pouvant être activées. Ce résultat signifie que le graphe de coordination d'un mode dégradé est un sous-ensemble du graphe du mode de marche normal. Donc, du point de vue du routage des pièces, le graphe du mode normal doit intégrer toutes les possibilités de routage offertes par le système de transport.

Dans ce qui suit, nous verrons les conséquences de la règle IV.1 sur la conception de la commande selon la méthodologie CASPAIM-2.

La règle IV.1 exprime qu'il est souhaitable de contenir les modes dégradés dans les modes normaux d'un système de commande. C'est un vœu pieu qui ne sera pas toujours réalisable. Restons toujours dans le cadre de la commande des ascenseurs d'un bâtiment. Cette fois, nous supposons que les deux ascenseurs sont situés dans des halls d'entrée différents disposés aux deux extrémités du bâtiment, et distants de plusieurs dizaines de mètres. A chaque étage du bâtiment, un couloir relie les halls des deux cages d'ascenseurs. Dans ce contexte, le pilotage doit tenir compte du fait que pour se rendre à l'ascenseur opposé, l'utilisateur mettra plus de temps qu'en mettrait l'ascenseur appelé pour arriver. Avec cette configuration de procédé, la politique P1 se révèle la meilleure. En cas de défaillance de l'ascenseur appelé, le mode dégradé consisterait alors à afficher un message d'alarme invitant l'utilisateur à utiliser l'ascenseur de l'entrée opposée. Le positionnement de ce dernier à l'étage d'appel, serait géré automatiquement par le système de commande pendant que l'utilisateur se rendrait au "hall" opposé.

Cet exemple montre que pour envisager une gestion transparente des modes de marches d'un système automatisé, il faut également que le procédé ait été conçu de manière à le permettre.

La gestion des modes de marche dégradés n'est pas une question que l'on se pose à posteriori après la conception d'un système, selon son mode de marche normal. Au contraire, elle doit être intégrée comme contrainte dès le début de la conception.

III.3 Conclusion

Dans cette partie, nous avons montré l'influence de la flexibilité du procédé sur la gestion des modes de marches et sur le pilotage. Toutefois, les exemples de politiques de pilotage montrent que cette flexibilité peut être utilisée par la commande selon des points de vue différents. Notre analyse nous amène à distinguer deux points de vue : un point de vue structurel et un point de vue fonctionnel. Ainsi, la politique P1 correspond à une approche structurelle : c'est à dire que la structuration de la commande est une transposition de celle du procédé. La politique P2 par contre, correspond à une approche fonctionnelle. On part du principe que ce qui intéresse un usager c'est de disposer d'un ascenseur pour se rendre à l'étage désiré. C'est la fonction ascenseur qui est prise en compte et non pas le fait que ce soit l'ascenseur A1 ou A2. La conclusion que nous inspire cette analyse, est que l'organisation de la commande selon une approche fonctionnelle, est la forme la mieux adaptée à la prise en compte

du recouvrement et donc à la gestion automatique des modes de marches par la fonction de pilotage.

IV. STRUCTURATION EN COUCHES DE LA COMMANDE

L'idée de base de cette structuration se situe dans une séparation au niveau de la commande, des différentes formes de flexibilité. Cette séparation doit permettre le recouvrement des défaillances à chaque niveau de commande, par le biais de la flexibilité. La structuration sera menée selon une approche fonctionnelle visant à intégrer la gestion des modes de marches à la fonction pilotage.

IV.1 Classification de la flexibilité du procédé.

La flexibilité du procédé concerne les produits et les ressources de production.

Au niveau produit, la flexibilité s'exprime par :

*** La flexibilité du produit fini.**

Elle consiste en la capacité à fabriquer des produits finis différents à partir d'un même produit brut.

*** La flexibilité de gamme.**

Certains produits peuvent être obtenus par différents séquençements des opérations de leur gamme de fabrication. Par exemple, supposons que les opérations d'un produit soit le perçage d'un trou et le dressage des faces perpendiculaires à ce trou. Dans ce contexte, les opérations de perçage et de dressage peuvent être menés dans un ordre indifférent.

Au niveau des ressources de production, on distinguera la flexibilité des machines de celle des moyens de transport. Dans les deux cas, la flexibilité s'exprimera également de deux manières :

*** La flexibilité d'affectation.**

Elle est définie par rapport à une opération. On définit une opération comme étant la fonction de transformation appliquée à un produit. C'est un processus opératoire indivisible [HAM91]. La flexibilité d'affectation consiste en la capacité du système de production à pouvoir affecter une opération à plus d'une ressource (existence de plusieurs machines de perçage dans l'atelier)

*** La flexibilité intra-ressource.**

Elle est définie par la capacité d'une ressource à réaliser plusieurs fonctions différentes (perçage, dressage, ...).

IV.2 Limites des spécifications de CASPAIM-2.

Comme nous l'avons vu au §I.1 du chapitre I, la méthodologie **CASPAIM-2** est **caractérisée par une mise à plat au niveau de la gamme opératoire, de l'ensemble des formes de flexibilité**. La flexibilité de gamme est introduite très tôt dans la spécification de la commande (dès les gammes logiques ; cf. figure I.3).

La flexibilité d'affectation est traduite au niveau de la gamme opératoire par la duplication des lieux opératoires (flexibilité des moyens de production) et la duplication des chemins entre lieux opératoires consécutifs (flexibilité du système de transport).

La mise à plat de ces différentes formes de flexibilité se traduit par une combinatoire importante. En fait, cela n'est pas étonnant ! Dans CASPAIM-2, la gamme opératoire traduit l'ensemble des ordonnancements admissibles définis à partir d'une gamme logique, en tenant compte de la flexibilité d'affectation relative à chaque opération, et de la flexibilité du système de transport. Une gamme opératoire correspond donc à un pré-ordonnement des opérations définies dans la gamme de fabrication d'un produit. Cela explique la complexité de la gamme opératoire que nous avons donné au chapitre I, figure I.5.

Analysons les raisons d'une telle démarche et essayons d'en tirer les avantages et les inconvénients.

Le but de cette démarche, est une gestion explicite des emplacements occupés par le produit, tout au long de son cheminement dans l'atelier. L'intérêt est qu'elle permet d'avoir un **pilotage de l'atelier déterministe**, basé sur le plan de fabrication défini en temps masqué par la fonction ordonnancement.

Le premier inconvénient que nous évoquerons se rapporte à l'utilisation de l'outil RdP. Ce dernier ne présente pas seulement l'avantage d'exprimer la concurrence et le parallélisme de la commande, mais encore permet une spécification graphique, facilement compréhensible. Lorsqu'on a une représentation telle que celle de la figure I.5 (chapitre I), on est en droit de se demander si les RdP restent appropriés. Il faut noter que le graphe de la figure I.5, correspond à une combinatoire de 16 cheminements, sachant que nous nous sommes limités à une flexibilité intra-ressource d'ordre 2 (opérations t+ et f+). Lorsqu'on passe à une flexibilité

d'ordre 3 (opérations t+, f+, ta+), le nombre de cheminements possibles passe à 64. Imaginons seulement la complexité d'une telle gamme opératoire !

Le deuxième inconvénient que nous citerons se rapporte à la gestion des modes de marche dégradés. Pour illustrer nos propos, nous allons utiliser de nouveau la gamme opératoire de la figure I.5. Supposons que pour une pièce donnée, on ait décidé de réaliser le tournage (t+) sur la machine M1 et le fraisage (f+) sur M2. Supposons par ailleurs que le tournage se soit bien déroulé et que la pièce soit située sur le poste P1 (i.e. La place correspondant à l'état d'avancement T+ et la localisation P1.palette-i est marquée) au moment où M2 tombe en panne. Le recouvrement est simple en apparence. Il suffit de reprendre la pièce en question avec le Robot R1 est de la mettre dans le tampon d'entrée E1 de la machine M1. La gamme opératoire ne spécifiant que le fonctionnement normal du système, ce cas n'est pas géré pour le moment par la commande. Il y a deux manières d'envisager cette gestion :

- * Une méthode consistant, par une procédure d'exception, à forcer le marquage du graphe ; on enlève le jeton de la place (Etat=T+, Lieu=P1.palette-i) et on le met dans la place (Etat=T+, Lieu=E1). Il faut par ailleurs prévoir la commande de R1 de manière à ce qu'il accomplisse effectivement cette opération de transport. Ce type de procédure viole les règles d'utilisation classique des RdP.
- * Une méthode plus cohérente, consistant à prévoir ce cas au niveau de la gamme opératoire (cf. figure IV.14) ; en mode normal la transition notée REC/M2 n'est jamais tirable ; elle le devient en cas de panne de M2 grâce au marquage de la place notée PANNE/M2. Dans ce cas, nous interprétons un mode dégradé au sein même du mode normal.

Ces deux méthodes présentent des inconvénients. La première parce qu'elle implique une manipulation de pointeurs et de contexte difficile à gérer. L'adopter serait signifier que l'on pourrait directement coordonner et commander des machines, sans utiliser les graphes de coordination.

La deuxième solution paraît plus intéressante en dépit de la combinatoire qu'elle induit. En effet, pour la mettre en œuvre, il faut intégrer au graphe toutes les opérations supplémentaires, nécessaires au recouvrement, quelque soit le contexte de production du système. Par exemple, dans le cas d'une panne de M2, nous distinguerons les contextes suivants :

- la pièce est en transit vers M1,
- la pièce est entre les tampons E1 et S1 de M1,
- la pièce est sur R1,
- la pièce est en transit vers M2,

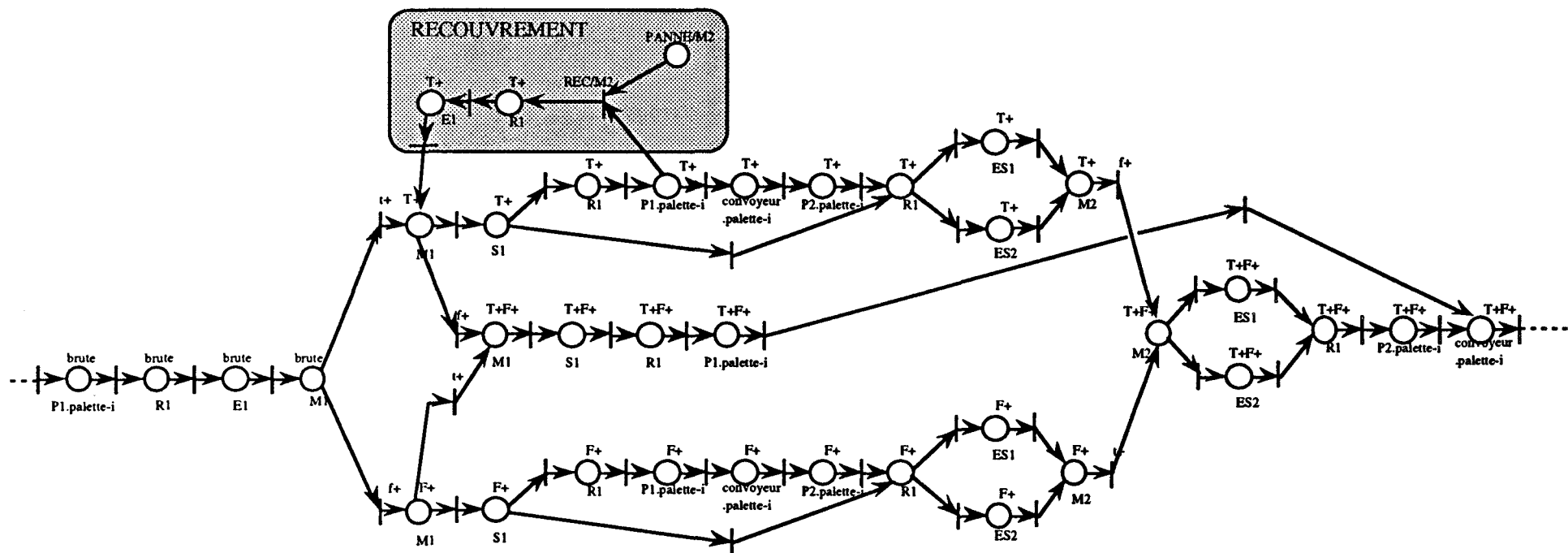


Figure IV.14 : Modélisation d'un recouvrement au sein d'une gamme opératoire

- la pièce est sur l'un des tampons ESi ou dans la ressource d'usinage M2.

Si on rapporte tous les cas de localisation de la pièce (ceux cités précédemment) aux différentes possibilités de défaillances (M1, M2, convoyeur, dériviations), on obtient un graphe dont la complexité est beaucoup plus importante que celle du graphe du fonctionnement normal. Cette solution reste toutefois applicable dans le cadre d'application peu complexe (peu flexible).

Pour limiter la combinatoire des états de la commande, nous pensons que :

- 1 - chaque forme de flexibilité doit être prise en compte à un niveau différent de la commande,
- 2 - la commande du mode de marche optimal doit être facilement paramétrable pour permettre la gestion des modes dégradés sans modification du graphe initial.

Pour mettre en œuvre une telle solution, nous proposons une décomposition en couche de la commande. Nous suggérons également que la gestion des modes de marche dégradés soit assurée par le pilotage.

IV.3 Présentation du modèle OSI [PUJ87].

La méthode de décomposition en couches est une méthode classique utilisée dans le cadre de la conception d'applications complexes. C'est une méthode ascendante consistant à **définir les fonctions de chaque couche à partir de la couche qui la précède**. En fait cette méthode est bien adaptée à une décomposition des systèmes par niveaux d'abstraction. Au niveau d'abstraction (i+1), on ne voit que les fonctions du niveau i.

L'exemple le plus connu, d'application de cette méthode, est le modèle OSI (Open System Interconnection) proposé par l'ISO (IS 7498 ou norme X200 du CCITT), pour la conception de réseaux informatiques hétérogènes pour la transmission de données (Fig. IV.15).

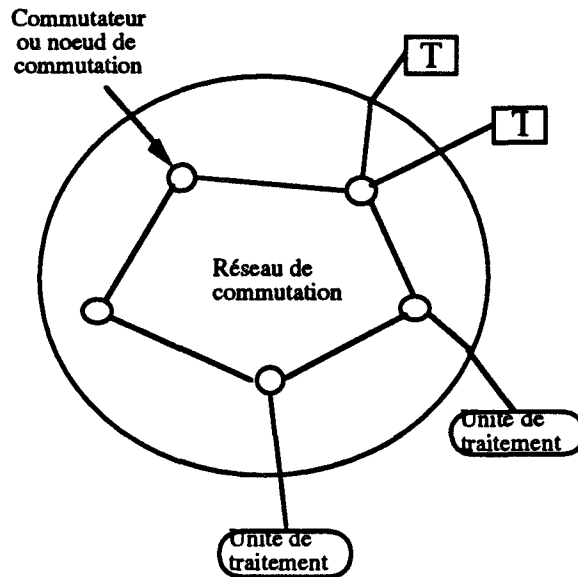


Figure IV.15 : Réseau maillé.

Le modèle OSI est constitué des sept couches suivantes :

* La couche physique.

Elle assure le transport de l'information. Il existe plusieurs techniques de transmissions normalisées ou non. La plus connue est l'avis V24 ou liaison série (RS232). Dans tous les cas l'unité d'information à ce niveau est le bit.

* La couche liaison.

A ce niveau, l'unité d'information est la trame. Elle est formée de bits de données et de bits de contrôles. Ces derniers permettent de contrôler les erreurs de transmission entre deux noeuds adjacents.

* La couche réseau.

A ce niveau l'unité d'information est le paquet. La couche réseau est responsable de l'acheminement des paquets à travers le réseau. Elle gère deux fonctions de base pour la transmission : le **routage des paquets** et le **contrôle de flux**.

Le contrôle de flux a pour but d'empêcher la congestion du réseau. Différentes techniques sont utilisées dans ce cadre :

- les politiques à seuil consistant à fixer un seuil maximal de paquets dans le réseau, ou en entrée de certains noeuds (en particulier des unités de traitement),
- les politiques de préallocation consistant à réserver les ressources (liaisons, tampons des commutateurs) nécessaires à l'acheminement de l'ensemble des paquets d'un même message,

- les contrôles de congestion consistant par exemple en la fermeture autoritaire de certains des circuits virtuels passant par un noeud congestionné. La couche transport est alors chargée de leur réouverture.

La fonction routage est une fonction essentielle à la reconfiguration du réseau en cas d'anomalies (panne de commutateur ou congestion, rupture de liaison ...). Aujourd'hui, les techniques centralisées tendent à disparaître, au profit des techniques distribuées. Ces dernières sont basées sur l'échange de tables de routage entre les différents noeuds du réseau. Ces tables permettent de connaître l'état des liaisons sur une certaine profondeur à partir d'un noeud. Sur cette base ont été élaborés de nombreuses techniques de routage adaptatif. Un exemple de cet type de routage est la méthode baptisée "Hot-potatoe". Le principe consiste à se débarrasser du paquet, le plus rapidement possible dans la première voie d'émission libre.

La couche réseau intègre également des fonctions de correction d'erreurs permettant le maintien de la qualité du service. Les erreurs irrécupérables sont notifiées à la couche transport.

* La couche transport.

Elle assure essentiellement deux fonctions :

- le contrôle du transport des données de bout en bout ; c'est elle qui s'occupe de la réouverture des circuits virtuels fermés par la couche réseau en cas de congestion d'un commutateur.
- le découpage du message en paquets sur le poste émetteur et l'assemblage des paquets sur le poste destinataire.

* La couche session.

Elle permet la synchronisation d'événements liés à des tâches distantes.

* La couche présentation.

Elle permet d'adapter les données à l'hétérogénéité des matériels raccordés au réseau.

* La couche application.

Elle gère l'application traitée qui peut être du type transfert de fichiers, messagerie, etc...

Notons ici, que si le modèle OSI constitue le modèle de référence, il est simplifié dans certains cas par la disparition de certaines couches. Ainsi, dans les réseaux locaux, les couches réseau, transport, session et présentation, disparaissent (Fig. IV.16).

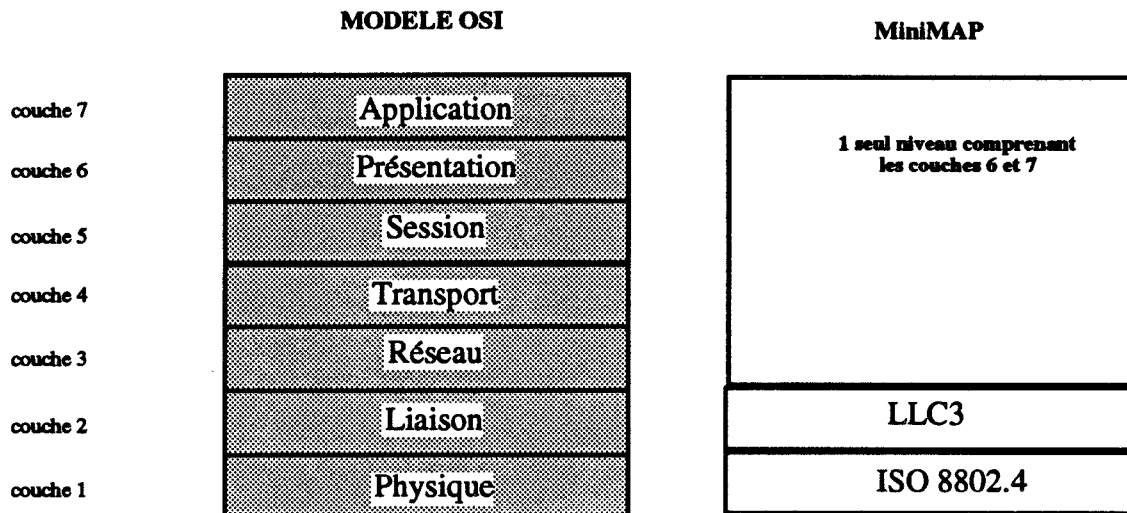


Figure IV.16 : Modèle OSI et MiniMap.

IV.4 Présentation des couches du modèle de commande

Dans la partie III (§ III) nous avons abouti à l'idée d'exigence d'une approche fonctionnelle pour l'intégration de la gestion automatique de modes de marches, dans la commande d'un SFP. Dans ce cadre, nous proposons une structuration de la commande en couches (Fig. IV.17).

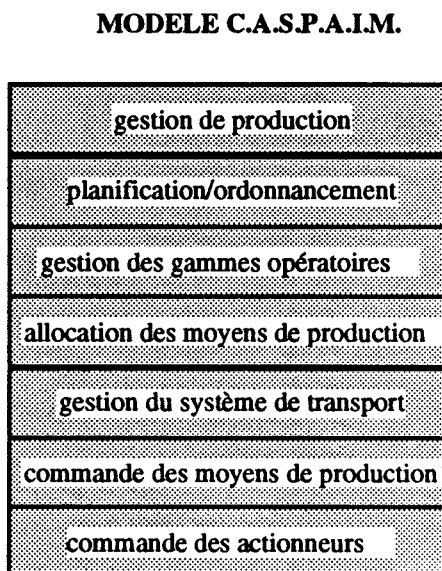


Figure IV.17 : Décomposition en couches de la commande d'un atelier flexible

Par la suite, nous nous intéresserons en particulier aux 5 dernières couches. Ces couches intègrent deux aspects de la commande : la commande locale (couche 1 et 2) et la commande de coordination (couche 3 à 5).

IV.4.1 Commandes actionneurs

C'est la commande la plus élémentaire que nous puissions avoir dans un atelier flexible. Il s'agit de gérer la marche et l'arrêt d'actionneurs tels qu'un vérin ou un moteur. A ce niveau, les actions sont indivisibles car adressées chacune à un et un seul actionneur.

IV.4.2 Commande des moyens de production

Au niveau précédent, nous avons une vision ensembliste de l'atelier. C'est à dire que l'atelier était vu comme un ensemble d'actionneurs. En fait, un atelier est composé de sous-ensembles réalisant une partition structuro-fonctionnelle de l'ensemble des actionneurs : ce sont les moyens de production (machines d'usinage, machines d'assemblage, ...) et les moyens de transport (robots, chariots filoguidés, convoyeur etc...).

En général, dans les méthodologies de conception des S.E.D., on ne s'intéresse pas à ce niveau de commande. En effet, souvent, il est directement intégré par les concepteurs, à la commande des M.O.C.N. Toutefois dans le contexte de la surveillance, afin de pouvoir réduire le temps de détection des défaillances, on est obligé de s'intéresser à ce niveau de commande. Au niveau maintenance également, la connaissance des comportements locaux permet de substituer en cas de pannes, des composants aux fonctions analogues. Prenons le cas de la commande d'ouverture de la porte du SAS (Système d'Accès et de Sécurité) présenté au chapitre III (cf. figure III.31). Formalisé avec les Réseaux de Pétri (RdP), cette commande se limite à envoyer la requête à une boîte noire implémentant la commande de l'actionneur de la porte (Fig. IV.18).

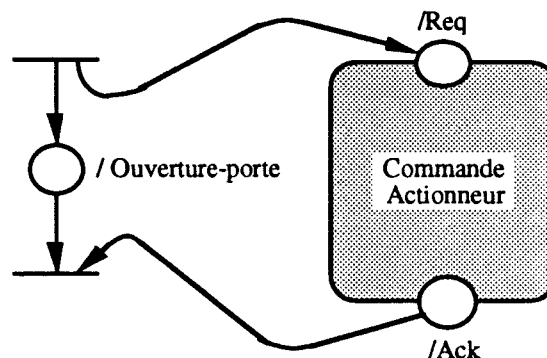


Figure IV.18 : Commande des moyens de production

En pratique, l'action d'ouverture de la porte du SAS peut être réalisée par un moteur (Fig. IV.19/a) ou par un vérin (Fig. IV.19/b).

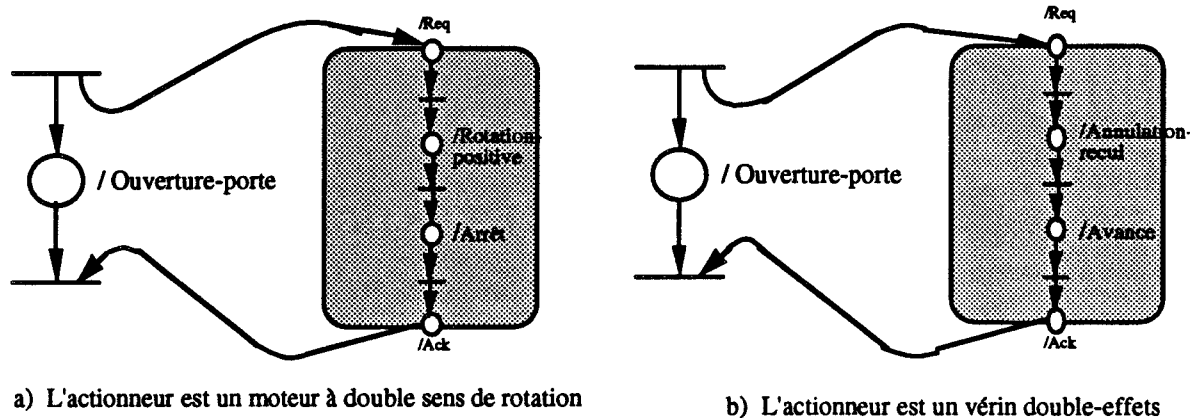


Figure IV.19

L'exemple précédent montre que la commande des moyens de production sert à coordonner le fonctionnement des actionneurs.

IV.4.3 Gestion du système de transport.

Le système de transport est caractérisé par :

- les moyens de transport qui le composent,
- et les lieux de stockage des produits (lieux de transit et postes opératoires).

L'approche fonctionnelle du transport consiste en la gestion du **transfert** des produits entre les lieux de stockage, indépendamment du moyen utilisé. Vu sous cet angle, le système de transport a une structure similaire à celle d'un réseau informatique. Les lieux de stockage sont assimilables à des commutateurs (cas des lieux de transit) ou des centres de traitement (cas des postes opératoires). Les ressources de transport (manipulateurs, convoyeurs, chariots filoguidés, ...) permettent de mettre en œuvre les liaisons entre les noeuds du réseau.

Considérons l'exemple du système flexible constitué des machines M1 et M2, du robot R1 et du convoyeur (cf . chapitre I, figure I.4). Les lieux de stockage gérés par la fonction transport sont :

- sur le convoyeur : FIFO-IN, FIFO-OUT, P1 et P2 ;
- sur M1 : les tampons E1 et S1,
- sur M2 : les tampons ES1 et ES2.

Un transfert élémentaire modélise l'accessibilité directe entre 2 lieux de stockage. La modélisation de l'ensemble des transferts élémentaires possibles donne le graphe de la figure IV.20. Ce graphe traduit la flexibilité du système de transport.

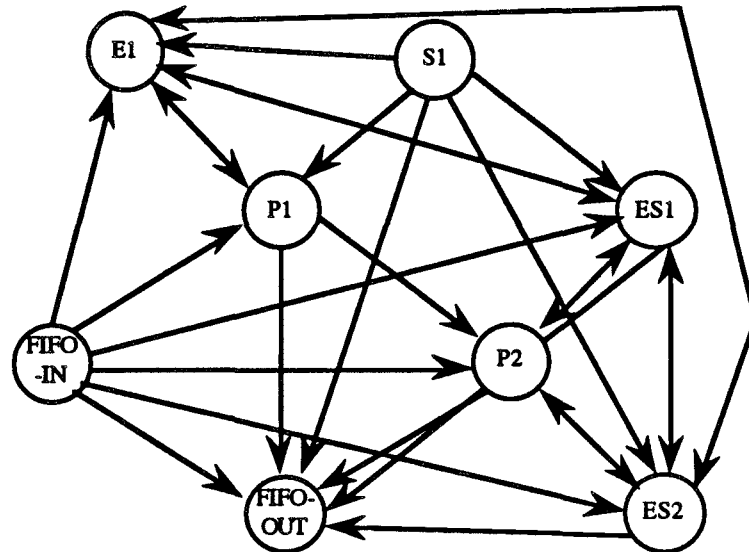


Figure IV.20 : Réseau de transport d'un SFP.

On peut remarquer que le graphe précédent s'identifie à la notion de prégraphe introduit dans la méthodologie CASPAIM-1. Cela n'est pas surprenant, dans la mesure où avec CASPAIM-1, la commande est conçue pour mettre en œuvre le rôle coordonnateur de la fonction de transport.

L'analogie entre les architectures des réseaux informatiques et celles des SFP se précise à ce niveau. En effet, on peut **identifier la fonction transport d'un SFP à la couche réseau** du modèle OSI. Ainsi, on distinguera principalement trois sous-fonctions à la gestion du transport :

- le **roulage** des produits et des outils dans l'atelier,
- le **contrôle de l'accessibilité** mutuel entre ressources physiques d'une même ressource logique (cf. § IV.4.4),
- la **synchronisation** des moyens de production.

Liaison	Mode de fonctionnement	Moyens
FIFO-IN -> P1	normal	convoyeur
FIFO-IN -> P2	normal	convoyeur
FIFO-IN -> E1	panne convoyeur	R1
FIFO-IN -> ESi	panne convoyeur	R1
FIFO-IN -> FIFO-OUT	dégradation fonctionnalités M1 et M2	convoyeur
P1 -> E1	normal	R1
P1 -> P2	normal	convoyeur
P1 -> FIFO-OUT	normal ou dégradé	convoyeur
P2 -> ESi	normal	R1
P2 -> P1	normal	convoyeur
P2 -> FIFO-OUT	normal ou dégradé	convoyeur
E1 -> P1	panne M1	R1
E1->ESi	panne M1	R1
S1-> P1	normal	R1
S1-> FIFO-OUT	panne convoyeur	R1
S1-> E1	panne M2	R1
S1-> ESi	panne convoyeur	R1
ESi -> P2	normal	R1
ESi -> FIFO-OUT	panne convoyeur	R1
ESi -> ESj	panne ESi	R1
ESi -> E1	panne convoyeur	R1

Figure IV.21 : Table des liaisons.

IV.4.3.1 Le routage.

Cette fonction est dépendante du degré de flexibilité du système de transport. Nous définissons la **flexibilité du système de transport** de la manière suivante :

Définition IV.5 : La flexibilité du système de transport [HAM91].

Un réseau de transport est **totalelement flexible** si son graphe est **fortement connexe** (tous ces sommets sont directement accessibles deux à deux).

Lorsque le réseau n'est que connexe, on dit que le système de transport est **flexible**.

Etant donné un lieu source et un lieu destination, le routage a pour fonction d'acheminer le produit par un des chemins potentiels reliant les deux lieux de stockage.

Définition IV.6 : Notion de chemin (du point de vue routage).

Un chemin est composé par l'ensemble des transferts élémentaires permettant de relier un lieu à un autre.

Comme dans le cas des réseaux informatiques, plusieurs politiques de routage sont envisageables. On les classera en deux familles : le routage statique et le routage adaptatif.

Le routage statique consiste à déterminer avant le début du transfert, le chemin qu'empruntera un produit. Cette détermination se fait par la recherche du chemin de longueur minimale [TAW90].

Au cours du cheminement, le routage peut être révisé pour trois raisons :

- la **rupture du chemin** : c'est à dire qu'un de ses transferts élémentaires ne peut plus être assuré faute de moyens de transport disponibles,
- la **congestion** d'un noeud appartenant à la partie aval (par rapport à la position du produit) du chemin,
- la **panne de la machine destinataire** : il n'y a plus lieu d'acheminer le produit sur une machine dont le temps d'indisponibilité est supérieur à la date de fin au plus tard de l'opération à réaliser sur le produit.

Dans ce contexte, le recouvrement considère trois cas :

* **Cas1** : Il y a panne d'un moyen de transport sans rupture de chemin.

Le recouvrement est local à la couche gestion du transport et s'appuie sur la flexibilité du système de transport. Un des moyens de transport restant permettra d'assurer le transfert sur la portion de chemin touchée par la panne.

* **Cas2** : En cas de rupture de chemin ou de congestion, le recouvrement est également local à la couche gestion du transport. Il consiste à trouver un nouveau chemin entre la position courante du produit et son lieu de destination.

* **Cas3** : En cas de panne de la machine destinataire, le routage initial est suspendu. Le recouvrement se déroule en deux étapes. La première étape consiste à déterminer une nouvelle machine de destination. Elle est réalisée par la couche supérieure (préallocation de moyens production). La deuxième étape consiste à déterminer un chemin allant de la position courante, au nouveau lieu de destination.

Le routage adaptatif consiste à décider en temps réel, à chaque noeud, du noeud suivant à atteindre. Il n'y a donc pas de chemin pré-déterminé. Comme dans le cas du routage statique, l'orientation des produits est basée sur l'évaluation d'un critère d'optimalité. On suppose qu'on dispose d'une fonction de valuation, permettant de pondérer les liaisons en temps réel. Une telle fonction doit évaluer le temps moyen d'un transfert élémentaire, à partir de paramètres tels que :

- le **flux planifié de produits** supposés transiter par la liaison ; il est obtenu en temps masqué, à partir du plan de fabrication pour la période,
- le **flux réel à la date considérée**,
- la **vitesse moyenne des moyens de transport opérationnels** assurant la liaison,
- le **taux d'occupation du poste desservi** par la liaison.

Des critères précédents, on se rend compte que seul le premier est fixe sur une période donnée. Le flux réel peut être calculé périodiquement. La comparaison du flux réel et du flux planifié permet d'éviter la congestion d'un noeud receveur.

La vitesse moyenne dépend des moyens de transport opérationnels. Ainsi en cas d'indisponibilité d'un de ses moyens, elle est recalculée. Une vitesse moyenne nulle est équivalente à une rupture de liaison.

Le plus difficile dans cette approche est de trouver une bonne fonction de valuation, réalisant un compromis entre la contrainte temps-réel et la finesse de l'évaluation.

Dans le contexte du routage adaptatif, le recouvrement au niveau local est implicite. Par contre, si on ne peut trouver une direction pour orienter la pièce, ou s'il y a panne de la machine destinataire, la couche supérieure doit être avertie. Elle aura la charge de déterminer une nouvelle machine de destination, compte tenu de la position courante du produit concerné.

IV.4.3.2 Le contrôle de l'accessibilité directe entre lieux de stockage.

Ce contrôle est fondamental pour le choix des machines destinataires. A chaque panne d'un moyen de transport, on doit évaluer si cela entraîne la rupture d'une liaison. En cas de rupture, la couche d'allocation des moyens de production doit être avertie. C'est elle qui a pour fonction le calcul de l'accessibilité relative entre machines.

IV.4.3.3 La synchronisation des moyens de production.

Lorsqu'un objet arrive sur son lieu de destination, la synchronisation avec la commande de la machine correspondante est réalisée par la mise à jour des données concernant les pièces en attente... En cas de panne de la machine, l'information est remontée jusqu'à la couche d'allocation des moyens de production.

IV.4.4 Allocation des moyens de production.

IV.4.4.1 Machines logiques.

La fonction d'un allocateur est d'affecter un moyen à l'exécution d'une tâche donnée. Dans le cas des SFP, la tâche met en relation une opération et un produit donné. Pour chaque produit, l'allocation est fonction (d'un point de vue statique) des critères suivants :

- l'opération à réaliser ,
- l'accessibilité de la machine allouée à partir du lieu de stockage courant,
- la disponibilité de la machine.

Plusieurs machines peuvent éventuellement satisfaire les critères précédents. D'où l'idée d'organiser les machines selon un point de vue fonctionnel visant à simplifier le pilotage et le recouvrement des défaillances : c'est le concept de **machine logique**.

Les machines logiques sont créées lors de la conception du système. Mais leur "structure" évolue dynamiquement en fonction de l'état de leurs composants respectifs (les machines physiques), et de l'accessibilité relative entre ces composants.

Définition IV.7 : Machine logique.

C'est un regroupement fonctionnel de machines physiques vérifiant les propriétés suivantes :

- 1 - Toutes les machines physiques du regroupement sont capables de réaliser la même opération (équivalence fonctionnelle),
- 2 - Toutes les machines du regroupement sont disponibles (équivalence de disponibilité),
- 3 - Toutes les machines sont mutuellement accessibles à l'intérieur du regroupement (équivalence d'accessibilité).

Proposition IV.3 :

Les machines logiques établissent des classes d'équivalences sur l'ensemble des machines physiques².

Démonstration :

Nous allons démontrer que les propriétés 1 et 3 définissent une relation d'équivalence sur l'ensemble des machines physiques.

Appelons O la relation "réalise la même opération que" (propriété 1) et A la relation "est mutuellement accessible avec" (propriété 3). Montrons que O et A sont des relations d'équivalence.

$\forall M_1, M_2, M_3 \in \mathcal{M}$

Relation O .

Réflexivité : $M_1 O M_1$

Symétrie : $M_1 O M_2 \Rightarrow M_2 O M_1$

Transitivité : $M_1 O M_2$ et $M_2 O M_3 \Rightarrow M_1 O M_3$

Démontrer que A est une relation d'équivalence est tout aussi évident.

Les propriétés précédentes sont utilisées pour introduire l'idée d'associer des machines physiques du point de vue de la maintenance. Les machines logiques seront utilisées dans la définition des gammes opératoires. Les machines logiques successives d'une gamme doivent vérifier des critères d'accessibilité externe. Nous définissons ces critères à l'aide des propositions IV.4 et IV.5.

² Par contre il n'y a pas partition de l'ensemble des machines physiques. En effet, une machine physique multifonctionnelle peut appartenir à plusieurs machines logiques.

Définition IV.8

Il y a accessibilité externe entre une machine logique et un lieu de stockage externe, si l'accessibilité est vérifiée par rapport à chaque machine physique du regroupement.

Proposition IV.4 :

Une machine logique est accessible à partir d'un lieu si l'une de ses machines est accessible à partir de ce lieu.

Démonstration :

Elle est triviale en raison de la propriété 3 de la définition de la machine logique.

Proposition IV.5 :

Une machine logique permet d'accéder à un lieu externe si l'une de ses machines à accès à ce lieu.

En phase de conception, la répartition des machines en machines logiques peut être obtenue facilement une fois l'architecture du procédé établi. Cette répartition peut être assistée par un outil de programmation par contrainte implémentant les propriétés 1 et 3.

IV.4.4.2 Pilotage

En phase d'exploitation, il est possible d'allouer à chaque machine logique la fonction d'affecter l'une des ses machines physiques à la réalisation de l'opération considérée (celle qui a servi de critère à la création de la machine), pour un produit donné. Cette allocation utilise les paramètres suivants, définis pour chaque machine physique :

- le taux d'allocation planifié,
- le taux d'allocation réel,
- la date de disponibilité au plus tôt de la ressource physique,
- la durée moyenne d'acheminement.

L'acheminement est réalisé par la couche transport à laquelle est transmise un jeton (requête) et l'indication du lieu de départ et du lieu de destination³. La donnée de ces deux lieux définit un

³ Cette forme d'acheminement est liée à l'affectation préliminaire de la machine cible. On pourrait envisager un pilotage plus adaptatif consistant à réaliser l'affectation à posteriori, en fonction du résultat d'un routage adaptatif de type adressage hiérarchisé (méthode utilisée dans les réseaux). Cela est concevable uniquement dans le contexte d'un regroupement géographique des constituants d'une machine physique.

ensemble de chemins possibles. Tous les chemins potentiellement admissibles sont modélisés au niveau de la couche de gestion du transport.

Lorsqu'un couple (opération, objet) a été affecté à une machine physique, ses paramètres taux-d'allocation-réel et disponibilité-au-plus-tôt doivent être recalculés.

L'allocation est ainsi déclenchée dans deux cas :

* Lorsqu'un produit doit subir une nouvelle opération. L'allocation est alors initiée par la couche supérieure (Gestion des gammes opératoires).

* Lorsque la disponibilité de la machine physique destinataire est remise en cause (défaillance) ou lorsque cette machine n'est plus accessible. Dans ce cas l'allocation est souvent limitée à une révision de l'allocation initiale. Elle est initiée par la propagation de l'information de défaillance par les couches inférieures (couche 2 ou couche 3).

IV.4.4.3 Recouvrement.

Nous distinguons trois cas :

* La panne d'un constituant de la machine logique.

Dans ce cas il y a violation de la propriété n°2. Cela entraîne le retrait de la machine physique en panne du regroupement. La machine logique continue à fonctionner de manière dégradée : c'est à dire que son taux d'engagement maximal devient inférieur au taux d'engagement optimal prévu à la création. Si des produits avaient été affectés à la machine physique défaillante, il faut effectuer une révision de l'allocation pour chacun d'eux.

Si tous les composants d'une machine logique sont en panne, l'information est transmise à la couche supérieure (gestion des gammes opératoires).

* La modification de l'accessibilité interne (propriété 3).

Le critère d'accessibilité interne a été défini pour permettre en cas de défaillance de la machine allouée, de trouver rapidement une autre machine accessible à partir de la position courante du produit. Il faut donc qu'au sein de la machine logique coexistent uniquement les machines mutuellement accessibles. Ainsi, toute modification de l'accessibilité interne a une répercussion sur la structure de la machine. En utilisant un test d'accessibilité [TAW90], on peut décomposer le regroupement en sous-regroupements dans lesquels toutes les machines physiques restent mutuellement accessibles.

*** La modification de l'accessibilité externe (proposition IV.4 et IV.5).**

Toute modification de l'accessibilité externe signifie qu'il n'y a plus de chemins disponibles entre deux machines logiques. Le seul recouvrement possible consiste dans ce cas à propager vers la couche supérieure, l'impossibilité de réaliser l'opération en cours.

IV.4.4.4 Conclusion.

Le concept de machine logique, est une notion importante dans notre approche du recouvrement. Cette notion est purement fonctionnelle. Les composants d'une machine logique peuvent appartenir à des cellules différentes d'un atelier. L'essentiel est qu'il y ait une accessibilité mutuelle entre ces cellules.

A contrario, la décomposition d'un atelier en cellules, postes, etc ... est liée à une approche structurelle. En effet, par définition une cellule regroupe un ensemble connexe de moyens de transport et de production. La connexité est une propriété structurelle (non fonctionnelle). Dans notre approche, une machine physique peut appartenir à des machines logiques différentes (Fig. IV.22). Il suffit qu'elle soit multi-fonctionnelle (flexibilité intra-ressource).

Sur la figure IV.22, on peut noter l'absence de machines logiques dans le modèle structurel. Par contre, elles sont représentées dans le modèle fonctionnel par les fonctions TOURNAGE, FRAISAGE et TARAUDAGE. Dans la représentation proposée du système M1-M2 (figure I.5), ces fonctions ne sont en relation avec aucune entité du modèle structurel.

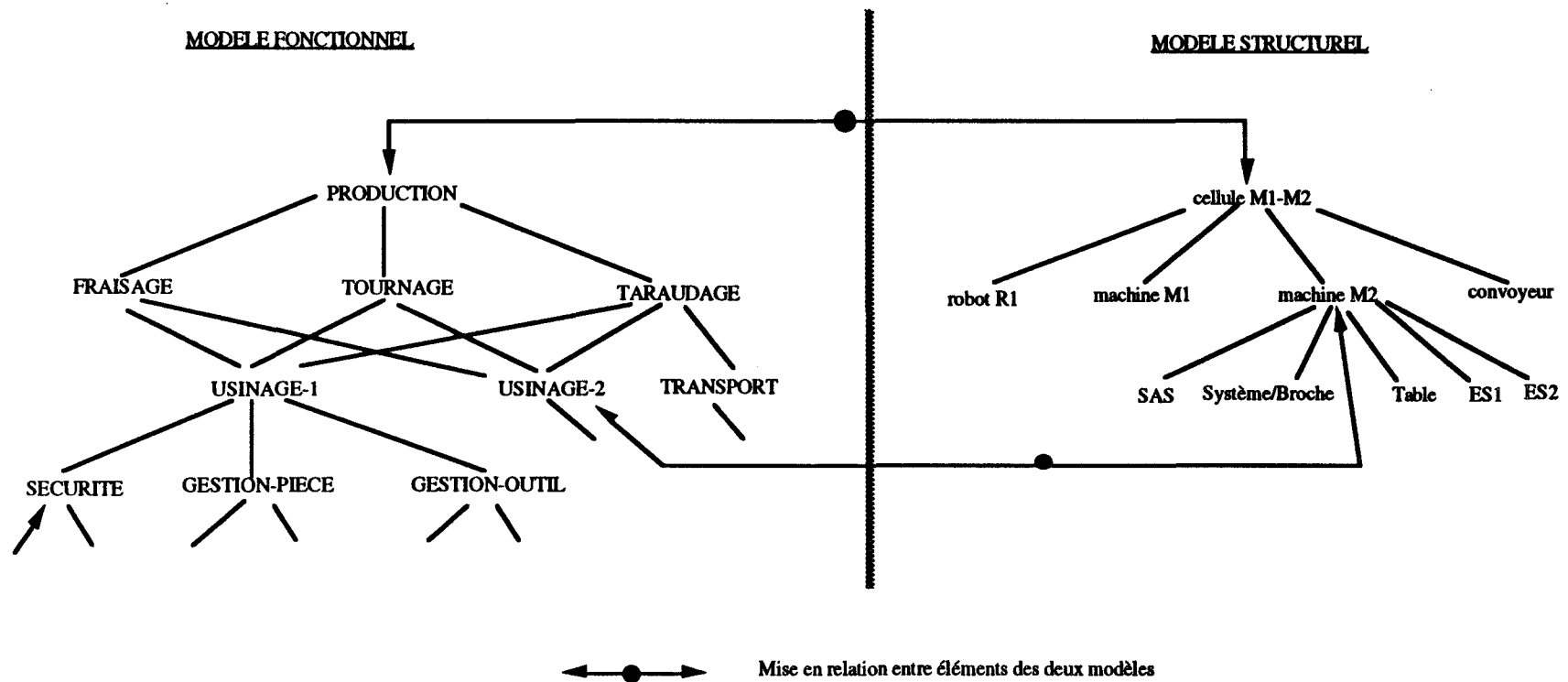


Figure IV.22 : Modèles fonctionnels et structurels des machines M1-M2.

On peut essayer cependant de concilier autant que possible les notions de machine logique et de cellule. L'idéal au niveau conception est de regrouper physiquement l'ensemble des machines physiques communes à plusieurs machines logiques (Regroupement spatial des constituants de machines logiques à intersection non vide). Ainsi, on limitera la durée des acheminements pour cause de recouvrement.

IV.4.5 Les gammes opératoires.

La notion de gamme opératoire que nous utilisons ici n'est pas celle introduite par D. Cruette (cf. chapitre I, § I.2.2.1.3). En fait, la notion de gamme opératoire que nous proposons ici, se situe à un niveau intermédiaire entre les définitions de gamme logique et de gamme opératoire présentées au chapitre I.

Définition IV.9 : Gamme opératoire.

Une gamme opératoire modélise les **opérations de transformation** que doit subir une famille de produit. Elle est représentée par un Réseau de Petri. Les places modélisent les zones de stockage d'entrée et de sortie des produits, et les lieux opératoires logiques sur lesquels doivent être exécutées les opérations.

La définition précédente correspond à une approche fonctionnelle du concept d'opération. En effet, un lieu opératoire logique modélise l'ensemble des lieux opératoires des machines physiques, membres d'une même machine logique. Cette définition permet de modéliser au niveau de la gamme uniquement la flexibilité de gamme. La flexibilité d'affectation est prise en compte au niveau de la couche allocation des moyens de production. De même, les lieux de stockages temporaires et les moyens de transport ne sont pas modélisés dans une gamme opératoire. La flexibilité du système de transport n'apparaît donc pas à ce niveau. Elle est modélisée dans la couche de gestion du transport.

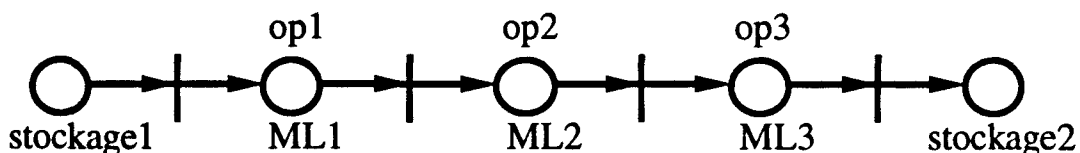


Figure IV.23 : Gamme opératoire d'usinage linéaire.

Ainsi, la figure IV.23 représente une gamme opératoire d'usinage linéaire, dans laquelle ML1, ML2 et ML3 représentent des machines logiques. Il n'y a pas de flexibilité de gamme

dans cet exemple. Toutefois, les opérations OP1, OP2 et OP3 pourraient être réalisées par une même machine physique disposant de ces trois opérations.

Il est important de noter ici, que l'opération notée au dessus de chaque place ne représente pas l'état d'avancement de la pièce. Elle représente seulement l'opération en cours de réalisation et associée à la machine logique notée sous la place. La notion d'opération introduite ici n'est pas limitée à sa réalisation effective par une machine. Elle intègre également les étapes d'allocation et d'acheminement. Comme nous l'avons illustré au chapitre I, la réalisation de chaque opération peut être contrôlée en associant des durées de sensibilisation minimale et maximale aux transitions de la gamme.

Sur la figure IV.24, nous avons représenté un exemple de gamme opératoire flexible. La place "test-out" est introduite ici pour représenter la flexibilité de gamme. Elle n'est associée à aucun lieu en particulier. Elle permet d'effectuer un choix.

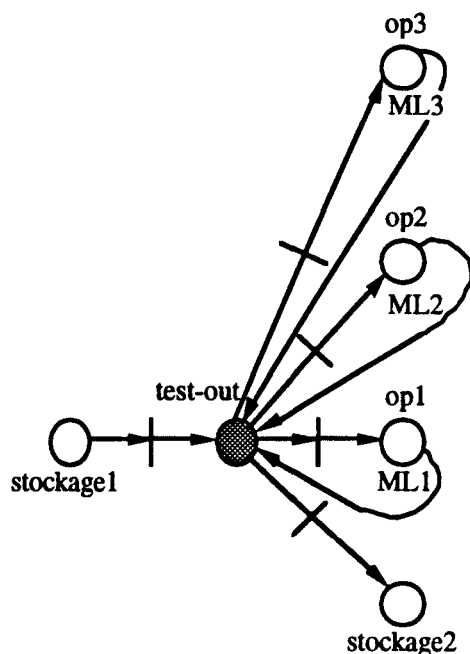


Figure IV.24 : Gamme opératoire flexible.

Avec $\text{stockage1} = \text{FIFO-IN}$, $\text{stockage2} = \text{FIFO-OUT}$, $\text{ML1} = \text{ML2} = \text{ML3} = \text{M1} + \text{M2}$, $\text{OP1} = \text{Tournage}$, $\text{OP2} = \text{Fraisage}$ et $\text{OP3} = \text{Taraudage}$, la gamme représentée par la figure IV.24 permettrait de représenter toutes les séquences d'opérations possibles pouvant être réalisées par le système des machines M1-M2 dont le cahier de charges est donné au chapitre I, § I.2.2.1.3.

Dans cette approche, plutôt que de représenter les caractéristiques des produits sur le modèle graphique, nous avons choisi de les associer au jeton représentant chaque pièce. Cela nécessite d'utiliser les Réseaux de Petri Objet pour modéliser les gammes. Ainsi, chaque jeton pourra être défini à partir des attributs suivants :

- **nature** : spécifie la famille du produit,
- **gamme** : liste donnant l'ordre des opérations que doit subir le produit ; lorsque plusieurs opérations correspondent à une flexibilité de gamme, elles sont représentées par une sous-liste dont l'ordre correspond à la priorité fixée par l'ordonnancement.
- **date-due** : date à laquelle le produit doit être livrée au plus tard,
- **l'état** : état d'avancement du produit dans sa gamme,
- **opération** : nom de l'opération en cours,
- **affecté-à** : machine physique auquel le produit est affecté pour l'opération en cours,
- **localisation** : lieu sur lequel se trouve le produit ; initialement il correspond au tampon d'entrée.

En fonction de la nature du pilotage, ces champs de base peuvent être initialement complétés ou non. Ainsi, pour gérer une pré-allocation de moyens de production, il suffit d'ajouter un attribut "pré-alloué-à" qui servira à coder le nom de la machine physique pré-allouée (par le niveau ordonnancement) à la fabrication d'un produit.

Pour l'exploitation de cette connaissance, il faut trouver un mécanisme de transfert des données du jeton nécessaires aux décisions prises dans les différentes couches de commande. Avec les RdPO, on est obligé de dupliquer les connaissances du jeton dans chaque couche. Le mécanisme de duplication est mis en œuvre des couches supérieures vers les couches inférieures (requête). En cas de modification de la valeur d'un attribut par une couche des niveaux inférieurs, la mise à jour à un niveau supérieur donné s'effectue par l'intermédiaire du compte-rendu d'activité ou "Ack". Nous illustrons ce point de vue sur la figure IV.25.

La gamme opératoire telle que nous l'avons définie ne permet pas de visualiser l'état d'un produit (état d'avancement, localisation sur le procédé, ...) . Toutefois, cet état peut être obtenu par la lecture des attributs du jeton relatif à un produit donné. La duplication des connaissances ne permet pas une mise en œuvre simple de ce mécanisme d'interrogation. Supposons qu'à partir du marquage d'une gamme opératoire nous voulions connaître l'état d'un produit. L'interrogation de l'attribut localisation donnerait par exemple le lieu de la dernière opération réalisée et non le lieu courant où est stocké la pièce. Il serait possible d'avoir à tout moment une image uniforme du produit, dans toutes les couches, en étendant la définition initiale des RdPO de C. Sibertin-Blanc aux **objets composites** [MAS89].

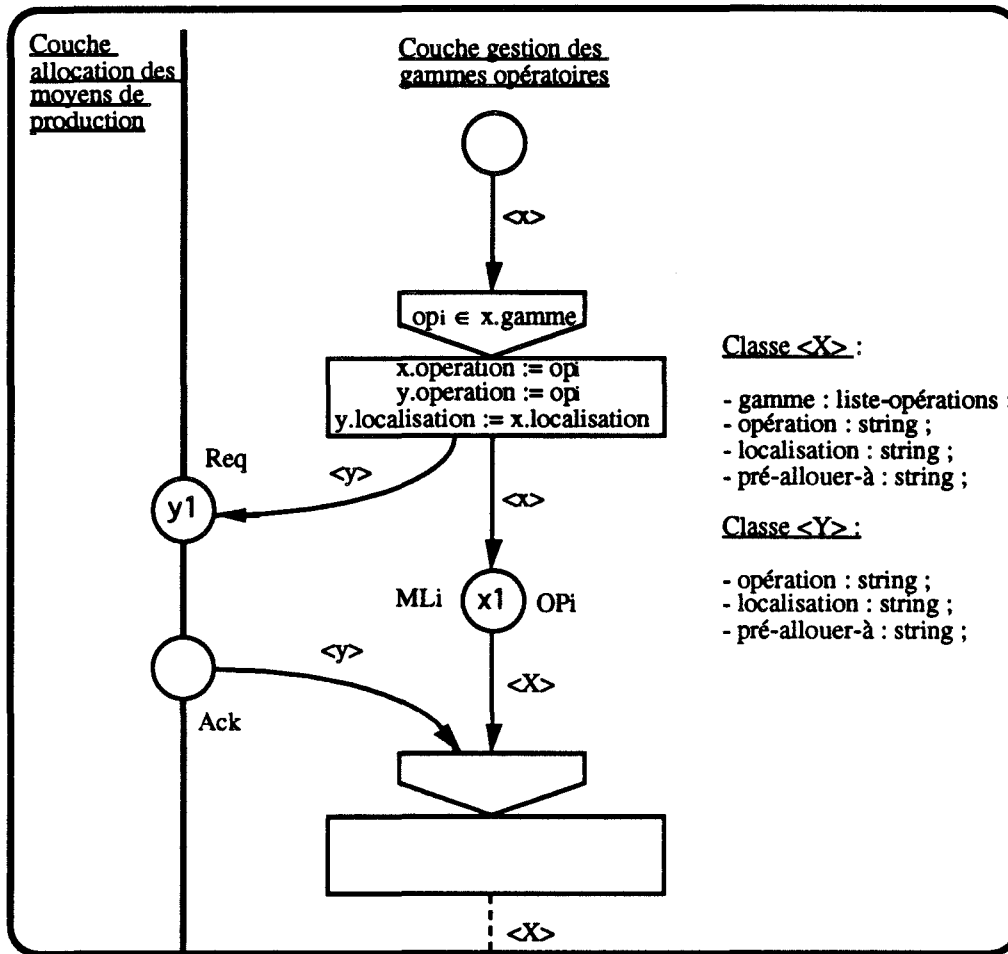


Figure IV.25 : Passage de données entre la couche gestion des gammes opératoires et la couche gestion de l'allocation.

Dans le cadre du recouvrement des défaillances, la notion de gamme flexible est très intéressante. En effet, elle permet de tenir compte du cas où une machine logique est indisponible (i.e. Cas où toutes les machines physiques qui la composent sont indisponibles ou non accessibles). Plutôt que de rester bloqué en attendant la disponibilité de cette machine logique, on peut réaliser une autre opération.

IV.5 Conclusion

La proposition de structuration en couches que nous proposons n'est pas une remise en question de la méthodologie CASPAIM-2. Les principes de base de CASPAIM-2 sont conservés. Elle s'avère mieux adaptée à l'implantation des modèles de commande [HUV92].

V INTÉGRATION DU RECOUVREMENT DANS L'ARCHITECTURE DE COMMANDE.

Nous avons proposé une alternative originale de structuration de la commande de coordination. Notre but ici, est de proposer une méthode de pilotage permettant une gestion du recouvrement adaptée à une structure de la commande de cette nature.

V.1 Ordonnancement et pilotage dans le contexte de CASPAIM-2 [HAM91].

Le but de la gestion de production est de planifier la production d'un système sur un horizon donné. Dans ce cadre, on distingue plusieurs fonctions qui sont :

- la planification,
- la programmation,
- l'ordonnancement,
- et le pilotage.

Le but de la planification est de définir la production de chaque période pour un horizon donné. Pour chaque période, elle fixera la nature et la quantité des produits à fabriquer, et leurs dates dues respectives. Elle aboutit à la détermination d'un plan directeur de production.

La programmation a pour but de vérifier l'adéquation entre la production planifiée et la capacité de production du système. Pour connaître les ressources disponibles, elle fait appel à la fonction maintenance.

En ordonnancement dynamique, on distingue deux grands types de problème : **l'affectation et l'ordonnancement.**

L'affectation est une décomposition spatiale qui consiste à déterminer à quelle ressource affecter une opération, sachant que potentiellement plusieurs ressources sont capables de la réaliser (Flexibilité d'affectation).

Au contraire, l'ordonnancement est une classification dans le temps des différentes opérations de la gamme d'un produit donné (Fig. IV.26) ; l'ensemble des ressources disponibles et leurs capacités sont figées.

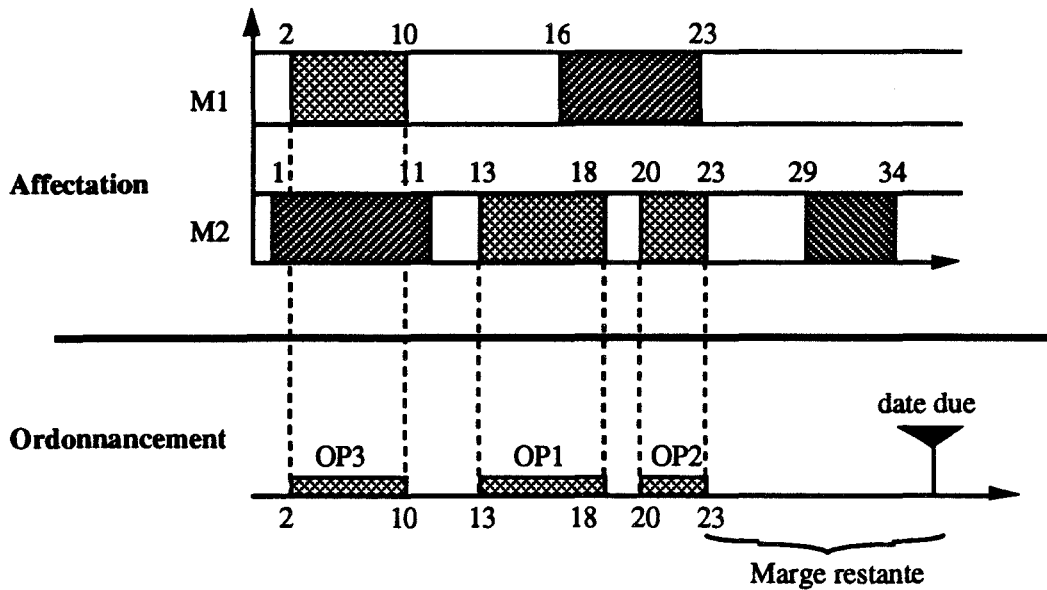


Figure IV. 26 : Affectation - Ordonnancement.

Dans la méthodologie CASPAIM, l'approche proposée par S. Hammadi [HAM91] propose une solution qui permet de résoudre conjointement les problèmes d'affectation et d'ordonnancement dynamique. Ainsi, dans un cycle de fabrication, chaque opération est caractérisée par :

- sa durée,
- sa date de début au plus tôt,
- sa date de fin au plus tard,
- la ressource à laquelle elle est affectée.

La différence entre la date de fin au plus tard et la date de fin au plus tôt constitue la marge (Fig. IV.26).

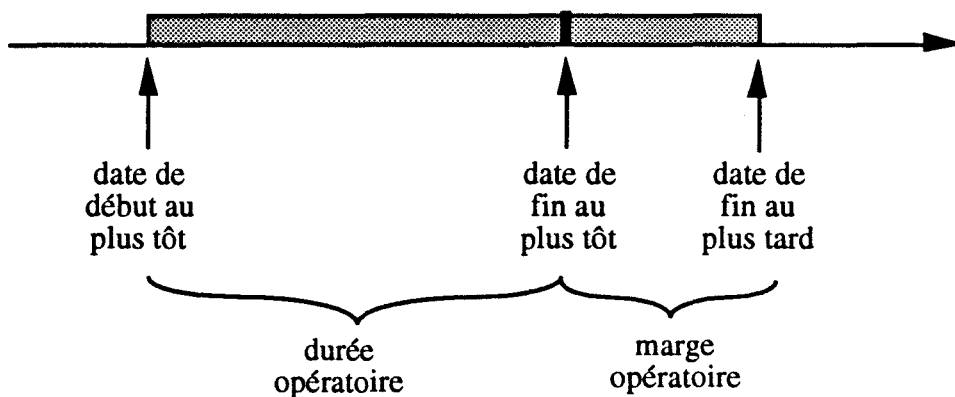


Figure IV.27 : Notion de marge d'opération.

La méthode d'ordonnancement retenue, permet de maximiser la marge de chaque opération grâce au principe d'insertion [HAM91]. Celui-ci consiste sur un pré-ordonnancement, à insérer des opérations de faible durée entre des opérations de durée plus importante. Le but est d'obtenir par les marges, une flexibilité temporelle permettant au pilotage d'adapter l'ordonnancement initial aux aléas de la production (rupture de stocks, pannes de ressources, ...). L'idée sous-jacente est que toute perturbation va décaler la date de début, d'au moins une opération dans une gamme. Dans ce contexte, trois modes de recouvrement sont envisageables :

* La marge restante de l'opération permet une reprise.

On met en œuvre un recouvrement de classe A1 (étendue locale et durée moyenne), géré par la fonction pilotage.

* La marge restante de la gamme permet d'envisager une reprise.

Cette fois-ci, l'ordonnancement reste admissible mais il faut décaler les dates de début de toutes les opérations de la gamme suivant l'opération défaillante. Le recouvrement est encore à l'initiative du pilotage qui va exploiter la flexibilité de marge de la gamme.

* L'ordonnancement n'est plus admissible.

La date due imposée par l'ordonnancement ne peut plus être assurée. Le pilotage en avertit le module d'ordonnancement qui va procéder à une **réactualisation** [HAM91] de l'ordonnancement initial.

Dans cette approche, lorsqu'un produit brut entre dans l'atelier, toutes les opérations qu'il doit subir sont connues par avance. De ce fait, sa gamme est linéarisée, même si à l'origine elle était flexible. Le seul degré de liberté du pilotage réside alors en l'exploitation des marges restantes (marges des opérations, marges des gammes). En aucun cas, le pilotage n'exploitera directement la flexibilité d'affectation ou la flexibilité intra-ressource. Pour cela il doit faire appel à l'ordonnancement.

L'approche que nous proposons a pour but d'exploiter toutes les formes de flexibilité au niveau même du pilotage.

V.2 Nouvelle proposition adaptée à la structure en couches.

Dans cette approche, le plan de fabrication produit par l'ordonnancement n'est plus vu comme une consigne stricte, mais plutôt comme une indication. De ce fait, le principe du pilotage consiste à tout instant à prendre les décisions permettant de respecter les ratios de production affectés par l'ordonnancement à chaque ressource. La fonction pilotage est décentralisée dans chaque couche. Lorsqu'une pièce rentre dans le système, sa gamme opératoire n'est pas figée. En fonction de sa nature et du différentiel entre les taux d'affectation réels et les taux planifiés corrigés (i.e corrigés en fonction des aléas), le pilotage de la couche gestion des gammes opératoires va choisir une opération de la gamme flexible. Une fois l'opération choisie, la machine logique correspondante en est informée. La fonction pilotage de la couche "gestion des machines logiques" choisit localement une machine physique. Cette machine devient le poste destinataire de la pièce. La "couche gestion des machines logiques" transmet à la "couche gestion du transport", un certain nombre de caractéristiques du produit parmi lesquelles le nom du poste source, du poste destinataire, et la durée admissible du transport. A partir de ces données, la couche de gestion du transport va acheminer le produit au poste destinataire, selon une technique de routage adaptatif. Dès que le produit atteint le poste destinataire, celui-ci en est avisé : c'est la synchronisation. Cette synchronisation a pour but de lancer effectivement l'opération de transformation sur la pièce. Lorsque le processus se termine normalement, il y a acquittement par les différentes couches jusqu'au niveau de la couche gestion des gammes opératoires. Et l'ensemble du processus recommence pour l'opération suivante.

En cas de défaillance, le recouvrement au niveau de la commande est assuré par le système de pilotage. Il est basé sur l'exploitation des différentes formes de flexibilité codées dans les couches successives rencontrées au cours de la propagation ascendante de l'information de défaillance. Supposons dans l'exemple des machines M1-M2, que la défaillance concerne l'une des fonctions d'une des machines. Pour une pièce se trouvant déjà sur la machine en question, on essaiera en priorité d'utiliser sa flexibilité intra-ressource : c'est à dire que le recouvrement consistera à exécuter une autre des opérations restantes de la gamme. Dans ce cas, c'est la couche "gestion des ressources physiques" qui fixe l'opération et la machine sur laquelle elle est exécutée. Ces données sont transmises aux couches supérieures pour une mise à jour dans chacune des connaissances relatives au produit.

Si la pièce est sur le système de transport au moment de la détection de la défaillance de la machine, ou si on est dans l'impossibilité d'exploiter la flexibilité intra-ressource, on utilise

alors la flexibilité d'affectation traduite dans les machines logiques. Dans ce contexte, la couche de gestion des gammes opératoires n'est pas informée de la modification de l'allocation initiale.

A travers cet exemple, on s'aperçoit que le recouvrement immédiat, à court terme, est assuré en utilisant la flexibilité selon une priorité inversement proportionnelle au niveau des couches, et fonction du type de la ressource (Fig. IV.28).

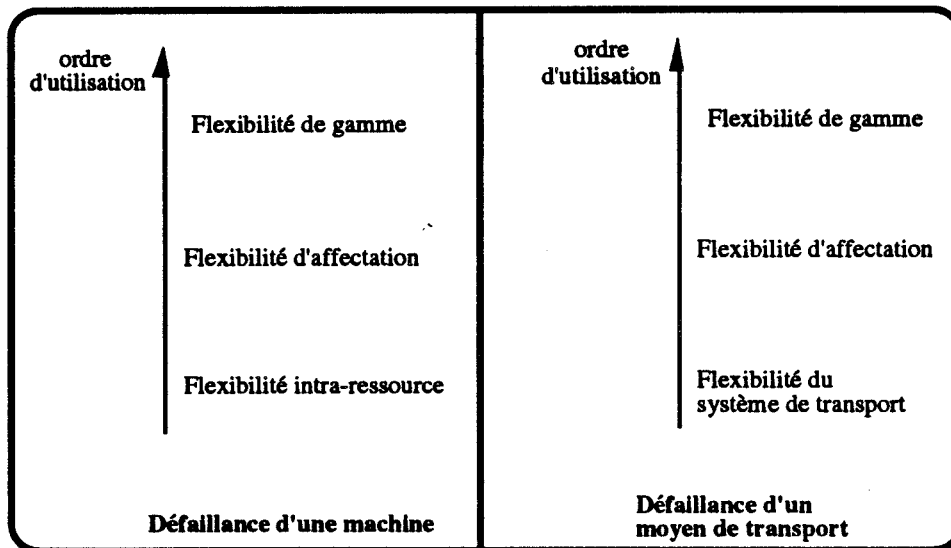


Figure IV.28 : Utilisation de la flexibilité.

Concernant le recouvrement différé à long terme, il est basé sur l'évaluation de l'indisponibilité des ressources défaillantes. Cette évaluation est assurée par la fonction maintenance, à partir de l'identification des composants défaillants. Le recouvrement différé peut conduire à deux solutions extrêmes :

- * Une solution peu contraignante consistant à corriger les ratios de production initiaux calculés pour chaque ressource par l'ordonnancement. Le principe de cette correction consiste à répartir la charge d'une ressource défaillante ou inaccessible sur les autres ressources physiques d'une même ressource logique. Ce paramétrage est directement effectué par le gestionnaire des modes de marche qui est positionné entre les modules maintenance et pilotage (Fig IV.29).

- * Une solution plus contraignante dans la mesure où elle fait appel aux fonctions supérieures de la gestion de production (ordonnancement, programmation et planification). Dans ce contexte, il y a une interaction directe entre la maintenance et ces fonctions.

L'intégration du recouvrement dans l'architecture de commande-surveillance est ainsi résumée par la figure IV.29.

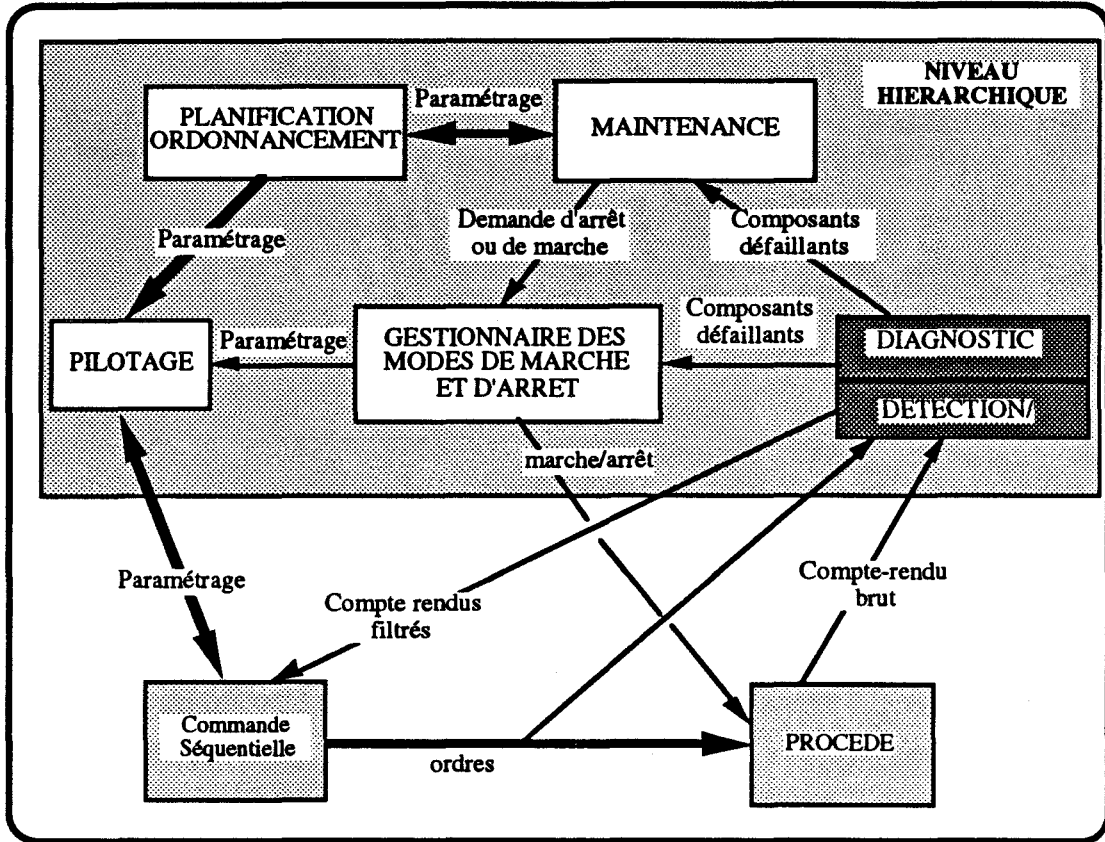


Figure IV.29 : Intégration du recouvrement dans l'architecture de commande-surveillance d'un SFP.

Dans l'architecture proposée, le **GESTIONNAIRE** a uniquement pour rôle de gérer les marches et arrêts des composants de l'atelier. Il n'assure aucune fonction de pilotage. Par contre, il est intégré à la gestion globale de la maintenance. Ainsi, le module de maintenance peut demander l'arrêt ou la remise en route d'un composant de l'atelier, dans le cadre de la maintenance sur état (recouvrement) ou dans le cadre de la maintenance préventive. Le fonctionnement du **GESTIONNAIRE** est identique à celui décrit dans la partie § II. La seule différence se situe au niveau des ordres générés. Le **GESTIONNAIRE** n'agit plus directement sur la structure du graphe de commande. Par rapport à la commande, les actions générées vont permettre la modification du paramétrage initial du système de pilotage.

CONCLUSION

Au cours de ce chapitre nous avons pu aboutir à l'idée que le recouvrement n'est pas une fonction de surveillance de même nature que la détection et le diagnostic. Cette fonction est intimement liée à la structure de la commande. En fait, chaque fonction du Niveau Hiérarchique intervient dans sa mise en œuvre.

Notre objectif principal était de montrer ici que la structure d'un système de production doit être conçue en prenant en compte les besoins du recouvrement, donc de la surveillance. Schématiquement, on peut dire que les étapes d'une démarche complète de conception d'un SFP sont les suivantes :

- 1 - dimensionner les capacités de chaque machine logique à partir des gammes de fabrication et des ratios de production de chaque produit ;
- 2 - faire un choix de ressources physiques (machines, moyens de transport) et d'architecture du procédé en fonction des critères de sûreté de fonctionnement du cahier des charges (disponibilité, maintenabilité, sécurité) ;
- 3 - concevoir de manière cohérente : la commande de coordination (gamme opératoire, graphe de transport) et la surveillance (détection, diagnostic, Gestionnaire des modes de marches).

En conclusion, si nous recommandons de séparer les fonctions commande et surveillance, nous sommes convaincus de la nécessité d'une conception conjointe de ces deux grandes fonctions.

CONCLUSION GENERALE

Notre approche de la surveillance est caractérisée par deux contraintes à priori opposées : d'une part assurer la sécurité des biens et équipements, et d'autre part permettre la disponibilité permanente de l'outil de production. Cette condition est nécessaire pour assurer le maintien des performances de production. Les contraintes citées, nous ont conduits à nous placer dans le contexte de la surveillance en ligne permettant **d'identifier et confiner les défaillances sans arrêter l'ensemble du système.**

Cette démarche induit une nécessité de réactivité au niveau de chaque fonction de la surveillance. Pour y faire face, nous avons proposé une structuration de la surveillance, dont l'originalité repose sur la détection par filtrage des comptes-rendus émis par le PROCÉDE vers la PARTIE COMMANDE.

La détection elle-même est basée sur la modélisation temporelle de chaque opération demandée au procédé. L'interprétation de ces modèles correspond à un mécanisme de perception-filtrage générant des symptômes. Nous avons proposé une spécification de ce mécanisme à l'aide des RdPO temporels. La détection est complétée par la classification des symptômes, technique permettant d'accroître la sécurité, en accélérant le processus de recouvrement.

Nous avons également indiqué que la contrainte de temps réel pose un problème délicat au niveau de la conception de la fonction diagnostic. Pour le résoudre, nous avons proposé une méthode de conception de cette fonction dont les caractéristiques principales sont les suivantes :

- * La **modélisation fonctionnelle du procédé** doit aboutir à l'élaboration de deux modèles distincts (le **modèle fonctionnel** et le **modèle structurel**) coopérant à la mise en œuvre du diagnostic et du recouvrement.
- * La nécessité d'un **algorithme de choix et de positionnement optimal de capteurs** supplémentaires, nécessaires au diagnostic. Ces capteurs permettent d'assurer l'observabilité du système, condition nécessaire pour permettre l'identification de toute défaillance en l'absence de l'apport d'information supplémentaire par l'opérateur.
- * L'introduction d'un **mécanisme de raisonnement temporel** basé sur un nouveau type de règles appelées STC ou **Signature Temporelle Causale**.
- * La mise en œuvre d'un **mécanisme de raisonnement hypothétique** basé sur l'utilisation du GF (**Graphe Fonctionnel**) comme réseau d'hypothèses, permettant d'implémenter l'**identification**, la **validation** et le **pronostic**.

* L'identification de chaque noeud du GF à un processeur de diagnostic, démarche permettant une **implantation répartie** du système de diagnostic.

Au niveau de la fonction recouvrement, notre démarche a consisté à intégrer un certain nombre de travaux connexes, élaborés au LAIL. Il s'agit notamment des travaux portant sur la gestion des modes de marches et ceux portant sur l'ordonnancement et le pilotage des ateliers flexibles. Cette intégration nous a conduit à proposer une structuration en couches de la commande permettant la gestion automatique des modes de marches par la fonction pilotage. Par contre la gestion effective des marches et arrêts des ressources est assurée par un gestionnaire conçu comme un outil de maintenance. Cette approche permet d'assurer la cohérence de l'ensemble des décisions prises par le Niveau Hiérarchique.

Les travaux présentés dans ce mémoire, permettent de dégager plusieurs axes prospectifs :

* Au niveau du recouvrement, il faut **poursuivre les travaux sur la structuration de la commande en couches**, en précisant notamment les interfaces entre couches successives, les procédures de reprise à l'intérieur de chaque couche, et les conditions de propagation des défaillances aux couches supérieures.

* Il faut **intégrer à notre approche, des méthodes de surveillance issues de la maintenance prédictive**. L'idée est de pouvoir détecter et suivre plus tôt les défaillances progressives avec des techniques plus fines que la discrétisation d'un signal continu à partir d'un seuil d'alarme.

* Il faut **intégrer l'ensemble des méthodes présentées ici, dans une démarche de Conception Assistée**. Une telle démarche doit permettre de générer en parallèle la commande, et les fonctions surveillance et supervision, selon une démarche cohérente, structurée et modulaire, en garantissant la réactivité du système de commande et les performances de production.

ANNEXES

SOMMAIRE

A - Sûreté de fonctionnement : Définitions des principaux concepts et des paramètres associés [VIL88],

B - Les Réseau de Petri A Objets ou RdPO [SIB88],

C - Compléments de la théorie logique.

ANNEXE A : SURETE DE FONCTIONNEMENTS - DEFINITIONS DES PRINCIPAUX CONCEPTS ET DES PARAMETRES ASSOCIES [VIL88]

Dans cette annexe, nous allons rappeler les définitions formelles et mathématiques d'un certain nombre de concepts fondamentaux de la sûreté de fonctionnement. Toutes les mesures s'expriment en terme de probabilité. Dans tout ce qui suit, E désignera une entité matérielle (système ou composant).

Définition A.1 : La fiabilité [CEI74].

C'est l'aptitude de E à accomplir une fonction requise, dans des conditions données, pendant une durée donnée.

En admettant que E est non défaillante à $t=0$, la fiabilité de E sur $[0,t]$ est :

$$R(t) = P [E \text{ non défaillante sur } [0,t]] \quad (A.1)$$

Remarques :

- On mesure parfois l'aptitude contraire appelée "défiabilité" et donnée par $1 - R(t)$
- $\lim_{t \rightarrow +\infty} R(t) = 0$

Définition A.2 : La disponibilité.

C'est l'aptitude d'une entité à être en état d'accomplir une fonction requise dans des conditions données à un instant donné.

Pour une entité E à un instant t donné, la disponibilité est :

$$A(t) = P [E \text{ non défaillante à l'instant } t] \quad (A.2)$$

Définition A.3 : La maintenabilité.

C'est l'aptitude d'une entité à être maintenue ou rétablie dans un état dans lequel elle peut accomplir une fonction requise, lorsque la maintenance est accomplie dans des conditions données avec des procédures et des moyens prescrits.

Sachant qu'une entité E est défaillante à l'instant $t=0$, l'expression mathématique de la maintenabilité est :

$$M(t) = P [E \text{ est réparée sur } [0,t]] \quad (A.3)$$

Définition A.4 : La sécurité

C'est l'aptitude d'une entité à éviter de faire apparaître, dans des conditions données, des événements critiques ou catastrophiques.

Nous pouvons établir des relations mathématiques entre certains de ces concepts. Ces relations sont fonctions de la nature du système :

- pour les systèmes irréparables nous avons $A(t) = R(t)$
- pour les systèmes réparables, nous avons $A(t) \geq R(t)$

Dans l'ensemble des travaux présentés dans ce mémoire, nous nous situons toujours dans le contexte des **systèmes réparables cohérents**.

Définition A.5 : Système Cohérent [VIL88].

Un système est cohérent si :

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

Dans certains domaines comme l'électronique, on peut avoir des systèmes non cohérents.

Les concepts précédents permettent de définir un certain nombre de paramètres qui permettent de mesurer le degré de confiance à accorder à un système. Ces paramètres sont :

Définition A.6 : Le MTTF ou Mean Time To Failure.

C'est la durée moyenne de fonctionnement d'une entité avant la première défaillance.

$$MTTF = \int_0^{+\infty} R(t) dt \quad (A.4)$$

Ce paramètre est surtout utile dans le cas des systèmes neufs ou considérés comme tels. Sinon, il vaut mieux utiliser le MTBF.

Définition A.7 : Le MTBF ou Mean Time Between Failure.

C'est la durée moyenne entre deux défaillances consécutives d'une entité réparée.

Définition A.8 : Le MDT ou Mean Down Time.

C'est la durée moyenne d'indisponibilité qui correspond à la durée de toutes les phases depuis la détection d'une panne, jusqu'à la remise en service.

Définition A.9 : Le MUT ou Mean Up Time.

C'est la durée moyenne de fonctionnement après réparation.

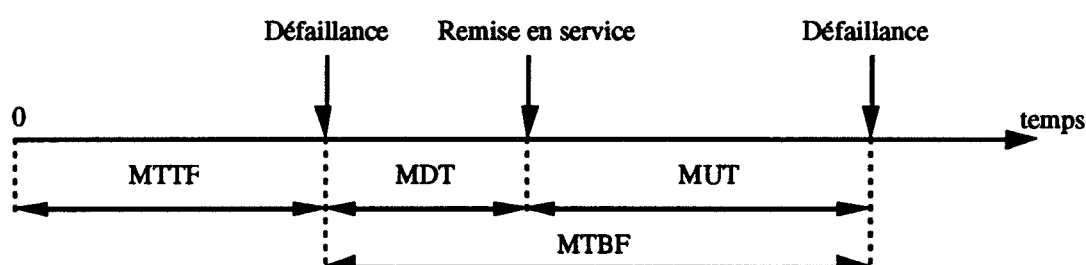


Figure A.1 : Représentation temporelle des principaux paramètres de fiabilité d'après A.VILLEMEUR [VIL88]

Sur la figure A.1, nous avons représenté ces paramètres par rapport au cycle de vie d'une entité. Il est important de noter que le MTBF est l'addition du MDT et du MUT.

D'un point de vue opérationnel, le MDT est le paramètre le plus important. En effet, **à la suite d'une défaillance, l'indisponibilité du système doit être la plus courte possible.** Il faut donc faire en sorte de réduire ce paramètre. C'est l'un des objectifs principaux d'un système de surveillance en ligne.

ANNEXE B : LES RESEAUX DE PETRI A OBJETS OU RDPO [SIB88].

Définition formelle :

Si O est l'ensemble des noms des Objets, on notera O^* l'ensemble des n-uplets d'Objets $O^* = \{ \langle o_1, \dots, o_n \rangle ; n \in \mathbb{N}, o_i \in O \}$ où 1 est l'élément de O^* tel que $n=0$, et $\mathcal{P}(O^*)$ l'ensemble des parties de O^* . On utilise donc les notations suivantes :

- * $o^* = \langle o_1, \dots, o_n \rangle$ est un jeton, c'est à dire, un n-uplet des instances de classes d'objets
et
- * o_i est une instance d'une classe d'objet, élément du n-uplet.
- * si $n=0$, $o^* = \langle 1 \rangle$ et le jeton est dit banalisé.

Un réseau de Petri à Objets est défini par la donnée des éléments suivants :

1 - Un ensemble C de Classes (ou type) d'objets éventuellement organisé sous la forme d'une arborescence de sous-types par héritage ; un n-uplet d'objets est typé par $C_o : O^* \longrightarrow C^*$.

2 - Un ensemble V de variables typées par $C_v : V \longrightarrow C$; une variable ne pourra être instanciée que par un objet de même type qu'elle, ou d'un sous-type de ce type.

3 - Un ensemble P de places ; chaque place p a une certaine arité a_p , et un type qui précise la Classe d'Objet de chacune de ses a_p composantes. La place p est typée par $C_p : P \longrightarrow C^*$. Une place d'arité 0 est dite de **jeton banalisé**.

4 - Un ensemble T de transitions.

5 - Une fonction d'incidence avant :

$$\text{Pre} : P \otimes T \longrightarrow \mathcal{P}(V^*)$$

et une fonction d'incidence arrière :

$$\text{Post} : P \otimes T \longrightarrow \mathcal{P}(V^*)$$

qui respectent l'arité et le type des places ; si une variable v figure dans $\text{Pré}(t, p)$, on dit que v (respectivement p) est une **variable** (resp. place) **d'entrée** de t ; si une variable v figure dans $\text{Post}(t, p)$, on dit que v (resp. p) est une **variable** (resp. place) **de sortie** de t .

6 - A chaque transition peut être associée une expression booléenne appelée **condition**.

7 - A chaque transition peut être associée un ensemble d'affectations, appelée **action**, de la forme :

$v. \text{attr} := \text{EXPR}$

où :

- v est une variable de sortie de la transition,
- attr est l'un des attributs de la classe de v ,
- EXPR est une expression de même type que $v. \text{attr}$ et dont les variables sont des variables d'entrée de la transition.

8 - Un marquage initial m_0 défini par la donnée d'un ensemble de n -uplets d'objets et d'une fonction de répartition qui donne pour chaque place son marquage, c'est-à-dire l'ensemble des n -uplets d'objets qu'elle contient.

Marquage.

Soit O l'ensemble des noms que peuvent prendre les objets, c'est-à-dire le domaine de l'attribut identificateur des objets.

Un état d'un réseau de Petri à Objets , que l'on appelle un marquage, est défini par la donnée :

1 - D'une fonction de répartition

$$m : P \longrightarrow \mathcal{P}(O^*)$$

qui indique pour chaque place les n -uplets d'objets qu'elle contient ; ces n -uplets doivent respecter l'arité et le type des places.

2 - De la valeur des objets figurant dans les places, c'est-à-dire, l'extension des relations définissant les Classes d'Objets du réseau.

Transition franchissable

Soit un marquage m et t une transition d'un Réseau de Pétri à Objets ; t est franchissable depuis m , ce qui est noté

$$m \xrightarrow{(t,S)}$$

s'il y a des Objets qui sont disponibles dans ses places d'entrée et qui satisfont sa condition. Plus formellement, cela veut dire qu'il existe une substitution S associant un objet à chaque variable :

$$S : V \longrightarrow O$$

telle que :

1 - Pour toute place p , $S(\text{Pré}(t,p)) \subseteq m(p)$, c'est-à-dire que chaque n -uplet de variables d'entrée de t peut capturer un n -uplet d'objets dans la place dont il étiquette l'arc allant vers t .

2 - Les objets capturés par les variables d'entrée satisfont la condition de t : si l'on note $C_t(v_1, \dots, v_n)$ cette condition pour la transition t et si la variable v_i capture l'objet o_i ($i=1 \dots n$), alors la proposition $C_t(o_1, \dots, o_n)$ est vraie.

Franchissement d'une transition

Le franchissement d'une transition franchissable depuis un marquage m produit un nouveau marquage m' , que l'on note $m \xrightarrow{(t,S)} m'$. Le calcul de m' se déroule de la façon suivante :

- 1 - Retirer des places d'entrée de t les n -uplets d'objets capturés par les variables ;
- 2 - Créer un nouvel objet pour chaque variable de sortie qui n'est pas en même temps une variable d'entrée, ce qui permet à toutes les variables de t de capturer un objet ;
- 3 - Exécuter en parallèle chacune des affectations de l'action de t ;
- 4 - Déposer dans les places de sortie de t les n -uplets d'objets capturés par les variables de sortie.

ANNEXE C : COMPLÉMENT SUR LA THÉORIE LOGIQUE.

Approche de R. Davis [DAV84].

a- Modèle de connaissance.

Dans le cadre de la description structurelle d'un système électronique, Davis a introduit trois concepts de base : les "**modules**", les "**ports**" et les "**terminals**" (Fig. C.1).

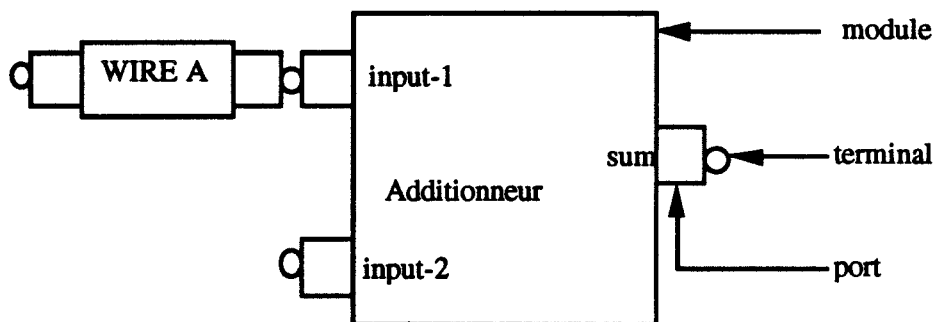


Figure C.1 : Principaux termes utilisés dans la description de la structure d'un système

Les modules représentent les composants du système. Selon le niveau d'abstraction considéré, ils modélisent des cartes électroniques ou des chips.

Les ports et les terminals permettent de modéliser la connectivité des entités du système. Les ports modélisent les liaisons entre l'extérieur et l'intérieur d'un module. De ce fait tout module comporte au moins deux ports. Les terminals modélisent les liaisons inter-modules d'un niveau donné.

Récursivement, tout module peut être modélisé par la même approche (Fig. C.2). On crée donc une hiérarchie de modules.

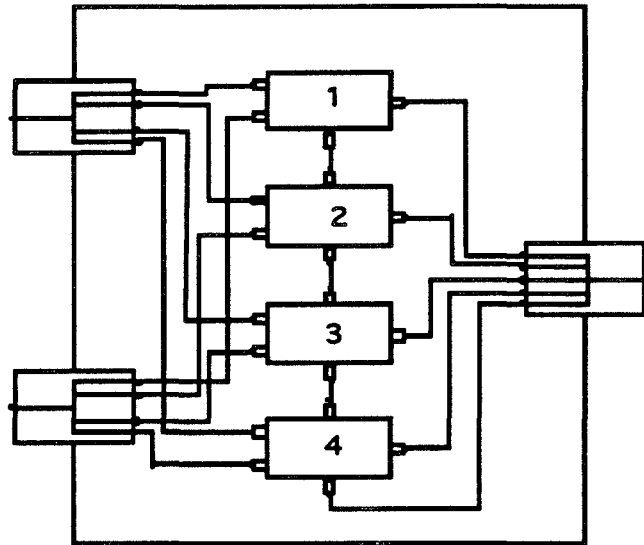


Figure C.2 : Modélisation structurelle d'un additionneur 4 bits.

Davis a développé en LISP, un langage permettant de générer une structure de données sur la base des composants et de la connectique spécifiée par un modèle graphique. Cette structure de données est utilisée par le diagnostic.

La seconde étape des travaux de Davis concerne la modélisation du comportement des constituants du modèle structurel. Le formalisme utilisé permet d'exprimer les relations entre les entrées et les sorties d'un module sous forme de règles ou contraintes. L'auteur distingue deux types de règles :

* Les règles exprimant le flux électrique au travers d'un module.

Si nous considérons l'exemple de la figure C.1, nous aurons la contrainte suivante :

to get sum from (input-1 input-2) do (+ input-1 input-2).

* Les règles exprimant le flux inférentiel.

to get input-1 from (sum input-2) do (- sum input-2).

to get input-2 from (sum input-1) do (- sum input-1).

b- Diagnostic des défaillances.

Le raisonnement est initié en comparant les valeurs des sorties observées d'un module par rapport aux valeurs simulées, à partir de celles des entrées du système. Il a pour fondement **le principe de violation de contraintes (Violated Expectations)** posé par J. de Kleer dans l'article intitulé " Local Methods for Localizing faults in electronic circuit". Ce principe est le suivant : si le comportement observé d'un module ne correspond pas au celui attendu, alors soit le module est défaillant, soit il plongé dans un environnement anormal (i.e ses entrées sont erronées).

Afin d'illustrer la démarche nous reprenons la carte de la figure C.1.

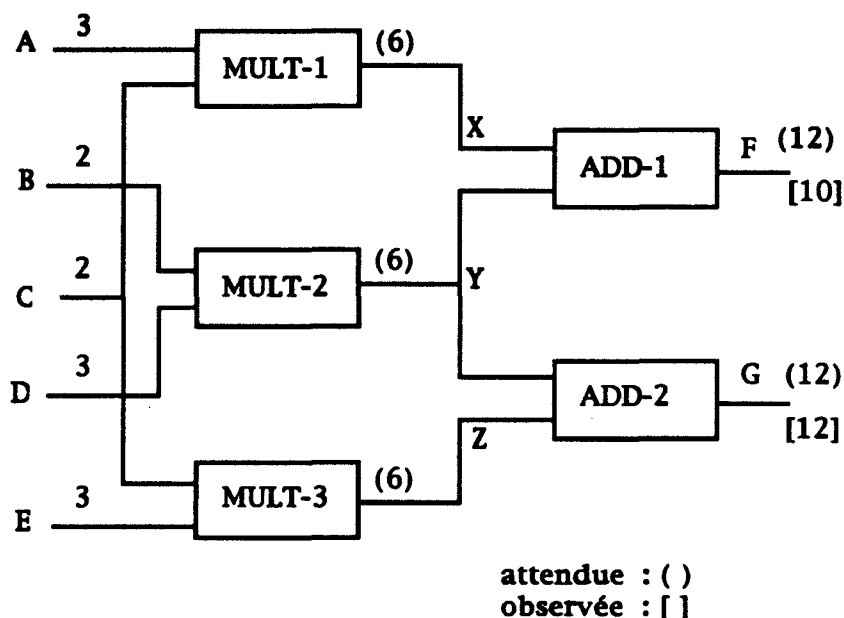


Figure C.3 : Exemple de diagnostic utilisant la violation des contraintes.

La carte électronique de cet exemple est constituée de trois multiplicateurs (MULT-1 à MULT-3) et de deux additionneurs (ADD-1 et ADD-2).

L'observation des variables de sorties (F et G) nous montre, une différence entre la valeur réelle de F qui est de 10 et sa valeur attendue 12. D'après le principe de violation des contraintes, trois hypothèses sont envisageables :

- ADD-1 est défaillant,
- la variable d'entrée de ADD-1, X est fausse,
- la variable d'entrée de ADD-1, Y est fausse.

Le principe de la résolution est d'écarter aucune hypothèse. L'utilisation des règles du flux inférentiel permet de calculer les valeurs des variables internes dans un contexte d'hypothèses précises. Par exemple, si l'on suppose que ADD-1 n'est pas défaillant et que la valeur de X est bonne ($X = 6$). Les règles du flux inférentiel permettent alors d'estimer que $Y = 4$. Cette valeur, injectée dans la règle exprimant la contrainte de fonctionnement de ADD-2 nous donnerait que $G = 10$. Cela introduit une nouvelle violation de contrainte par rapport à G (puisque $G = 12$ est la valeur observée). On peut répéter le raisonnement tenu par rapport à F. Ainsi si on continue dans cette voie on aboutit à la conclusion que les ensembles de composants défaillants possibles sont { MULT-2, ADD-2 } et { MULT-2, MULT-3 }.

Le mécanisme de résolution utilisée par Davis est donc l'analyse systématique de toutes les hypothèses issues de la violation d'une contrainte de fonctionnement. Aucune hypothèse n'est éliminée par des critères probabilistes ou statistiques. Dans l'exemple précédent on aboutit ainsi à donner plusieurs solutions pour expliquer le comportement défaillant de la carte. Ces solutions sont les ensembles de composants suivants :

{ ADD-1 }, { MULT-1 }, { MULT-2, ADD-2 }, { MULT-2, MULT-3 }.

Nous utiliserons cet exemple pour présenter les autres approches de la théorie logique.

Approche de M.R. Genesereth [GEN84].

Elle a pour fondement, l'utilisation des données de conception (issues d'une CAO), pour diagnostiquer automatiquement les systèmes ou processus physiques. Elle a conduit au développement du programme DART.

DART est sensé assurer le diagnostic en coopération avec un programme de test automatique.

L'originalité des travaux de Genesereth se situe dans l'utilisation de la logique du premier ordre lors de la modélisation des systèmes.

a - Modélisation.

Genesereth distingue deux phases dans la modélisation : la modélisation pour la conception du système et la modélisation pour le diagnostic du système.

En conception, la modélisation concerne deux axes : la description structurelle et la description du comportement.

Au niveau structurel, chaque composant est typé à l'aide de prédicat logique. Si on se réfère au circuit électronique de la figure C.3, on peut distinguer les connaissances suivantes :

* Connaissances structurelles relatives aux composants.

SD1 : (MULT M1)

SD2 : (MULT M2)

SD3 : (MULT M3)

SD4 : (ADD A1)

SD5 : (ADD A2)

* Connaissances structurelles relatives aux liens inter-composants.

SD6 : (CONN (IN 1 F1) (IN 1 M1))

SD7 : (CONN (IN 3 F1) (IN 2 M1))

SD8 : (CONN (IN 2 F1) (IN 1 M2))

SD9 : (CONN (IN 4 F1) (IN 2 M2))

SD10 : (CONN (IN 3 F1) (IN 1 M3))

SD11 : (CONN (IN 5 F1) (IN 2 M3))

SD12 : (CONN (OUT M1) (IN 1 A1))

SD13 : (CONN (OUT M2) (IN 2 A1))

SD14 : (CONN (OUT M2) (IN 1 A2))

SD15 : (CONN (OUT M3) (IN 2 A2))

F1 représente la carte électronique primaire.

Une connaissance comme SD6 s'interprète de la manière suivante : l'entrée 1 de la carte F1 (IN 1 F1) est connecté (CONN) à l'entrée 1 du multiplicateur M1 (IN 1 M1).

Le comportement du système et de chacun de ses composants est également décrit par des règles utilisant des prédicats du langage. Là aussi, Genesereth distingue deux types de connaissances :

* Les connaissances modélisant le comportement du système tel que défini dans le cahier des charges (Behavioral Data).

Ces connaissances font abstraction de la structure interne du système.

Exemple : Circuit F1 avec A=3, B=2, C=2, D=3, E=3

BD1 : (IF (AND (VAL (IN 1 F1) t 3) (VAL (IN 2 F1) t 2)
(VAL (IN 3 F1) t 2) (VAL (IN 4 F1) t 3) (VAL (IN 5 F1) t 3))
(AND (VAL (OUT 1 F1) t 12) (VAL (OUT 2 F1) t 12)))

La règle BD1 donne les valeurs de sortie de F1 en fonction de celles de ses entrées.

* Les connaissances relatives à des composants de base vérifiant les lois de la physique (Theoretical Data).

Par exemple le comportement théorique d'une porte logique AND pourrait être modélisé par la règle suivante :

TH1 : (IF (AND (ANDG d) (VAL (IN 1 d) t ON) (VAL (IN 2 d) t ON))
(VAL (OUT 1 d) t ON))

Pour le diagnostic, les données issues de la conception doivent être enrichies par des connaissances de deux nouveaux types :

* Les connaissances relatives à la description de l'environnement du système (Achieval and Observable Data).

ACi : (VAL (IN i F1) t x)

La variable "x" représente la valeur de l'entrée "i" de F1 à la date t.

OBj : (VAL (OUT j F1) t y)

La variable "y" représente la valeur de la sortie "j" de F1 à la date t.

* Les connaissances relatives aux hypothèses de défaillance (ou de non défaillance) introduites par l'utilisateur (Diagnostic Assumption).

DA1 : (IF (NOT (ADD A1)) (AND (ADD A2) (MULT M1) (MULT M2) (MULT M3)))

La règle précédente traduit l'hypothèse de défaillance de l'un des composants du système (NOT (ADD A1)) .

b - Résolution.

Elle est activée par la donnée à une date, d'observations ne correspondant pas à celles attendues (observations négatives). Ces observations sont alors interprétés comme des symptômes. A partir d'une inférence en chaînage arrière, l'ensemble des composants suspectés est établi.

Le fonctionnement de DART repose sur l'hypothèse de défaillance unique. Aussi, si l'ensemble des suspects est réduit à un composant, le diagnostic est terminé. Dans le cas contraire, DART prend en compte les observations positives fournies en même temps que les symptômes pour réduire l'ensemble des suspects. Ensuite, il tente de générer une série de tests permettant la discrimination du composant défaillant. Si la génération est réussie, la procédure de diagnostic est appliquée itérativement à partir des observations produites par les tests. Sinon, DART conclue en donnant l'ensemble minimal des composants suspectés.

c - Conclusion.

L'intérêt de l'approche proposée par Genesereth se situe au niveau de l'utilisation de la logique pour modéliser les systèmes dans le cadre du diagnostic automatique.

Toutefois, on peut douter de l'efficacité de DART notamment pour le diagnostic de système complexe autre que les circuits électroniques. Cela est sensible, notamment dans la modélisation du comportement normal du système à diagnostiquer (Behavioral Data). Malgré l'utilisation de prédicats logiques on a le sentiment qu'on peut être confronté à une combinatoire importante dans le cadre de la modélisation de comportements complexes.

Par ailleurs l'hypothèse de défaillance unique reste assez restrictive. En effet, que vaut le programme DART lorsqu'en cours de génération de tests discriminatoires, il survient une nouvelle défaillance ?

Ces critiques trouvent leur réponse dans l'approche proposée par Reiter [REI87].

Approche de J. de Kleer [KLE80], [KLE84], [KLE87], [FRO90]

Les premiers travaux de Johan de Kleer date de 1976 avec l'approche "Violated Expectations". Cette approche concernait uniquement le diagnostic des systèmes digitaux. Elle avait pour fondement l'hypothèse de défaillance unique. En 1987, de Kleer proposa l'approche "General Diagnosis Engine" (GDE) [KLE87] qui généralise ses travaux antérieurs. Comme son nom l'indique, elle se pose comme approche universelle pour diagnostiquer tout type de systèmes ou processus.

a - Principe.

de Kleer conçoit le diagnostic comme une tâche itérative. Son but est d'identifier la défaillance à partir des mesures courantes, effectuées sur le processus. Si l'identification échoue la tâche de diagnostic doit déterminer les mesures additionnelles à effectuer. Dans cette approche le diagnostic utilise également un modèle explicite du procédé.

b - Modélisation

Le modèle du système à diagnostiquer est défini par la donnée de :

- la description de sa structure physique,
- le modèle de chacun de ses constituants.

Nous voyons là que le concept du modèle est un concept récursif.

Dans la conception de "de Kleer", la notion de constituant est plus large que celle généralement admise. En fait elle intègre à la fois des aspects structurels et fonctionnels du système. Ainsi, l'ensemble des constituants peut inclure les composants physiques de système, les processus et même les étapes d'une inférence logique. Cette notion est donc un concept central pour comprendre le formalisme de GDE.

Exemple : Le système à diagnostiquer est de nouveau la carte électronique de la figure C.3.

Les constituants du circuit sont : M1, M2, M3, A1 et A2.

Le comportement de chaque composant est donné par son équation logique :

M1 : $X = A * C$

M2 : $Y = B * D$

M3 : $Z = C * E$

A1 : $F = X + Y$

A2 : $G = Y + Z$

c - Résolution.

Elle s'inscrit dans le cadre de l'hypothèse de défaillances multiples. Elle est initiée par les écarts détectés entre les variables mesurées (appelées Observations ou OBS) et leur prédiction. La résolution dans GDE s'appuie sur trois concepts de base :

*** L'Environnement (ENV).**

Il désigne un ensemble de composants (donnés entre accolades { }) supposés non défaillants et utilisés pour prédire la valeur d'une variable.

{A1, M1, M2} est l'environnement utilisé pour calculer F.

*** Les Conflits (< >).**

Un conflit est un ensemble de constituants dont l'un au moins est défaillant (disjonction).

Les conflits sont détectés en lançant une inférence C sur la base des observations OBS et de l'environnement associé ENV (C(OBS, ENV)).

Dans l'exemple de la carte électronique, C({F = 10}, { A1, M1, M2 }), aboutit à une incohérence (Fmesuré = 10 et Fprédit = 12). Les éléments de l'environnement constituent donc un conflit noté < A1, M1, M2 >.

*** Candidats ([]).**

Un candidat est un ensemble de constituants défaillants (conjonction).

Les candidats correspondent à la conjonction des conflits minimaux. Dans l'exemple précédent, le rejet des valeurs mesurées de F et G conduit respectivement aux conflits $C1 = \{ A1, M1, M2 \}$ et $C2 = \{ A1, A2, M1, M3 \}$.

Le diagnostic est donné par l'ensemble des candidats minimaux :

[A1], [M1], [A2, M2], [M2, M3].

Le mécanisme de résolution de GDE est basé sur une coopération avec ATMS [KLE87] (Assumption based Truth Maintenance System). Ce système est chargé du maintien de la vérité des hypothèses émises par GDE dans le cadre de la génération de tests optimaux pour l'enrichissement de l'ensemble OBS.

**REFERENCES
BIBLIOGRAPHIQUES**

INTRODUCTION GENERALE

[BOI91] S. Bois, Intégration de la gestion des modes de marche dans le pilotage d'un système automatisé de production, Thèse de de Doct. de l'Université, Lille, 28 Novembre 1991.

[BRU90] J. Brunet, D. Jaume, M. Labarrère, A. Rault, M. Vergé, Détection et diagnostic de panne : approche par la modélisation, Hermès, 1990.

[CAR90] J. Cardoso, Sur les réseaux de petri avec marquages flous, Thèse de Doct. de l'Université, Université Paul Sabatier, Toulouse, Octobre 1990.

[SAH87] A.E.K. Saharaoui, Contribution à la surveillance et à la commande d'atelier, Thèse de Doct. de l'Université, Toulouse 1987.

[SEM86] V. Sembugamoorthy, B. Chandrasekaran, "*Functional Representation of Devices and Compilation of Diagnostic Problem Solving Systems*", in J.L. Kolodner and C.K. Riesbeck, *Experience, Memory, and Reasoning (Ed.)*, Lawrence Erlbaum Associates, 1986.

CHAPITRE I: LA SURVEILLANCE EN LIGNE

[ALA86] P. Alanche, P. Salvi, G. Morel, M. Roesch, P. Lhoste, M. Salim, "*Modélisation de la Partie Opérative et Application à la structuration de la commande des systèmes automatisés*", Journées AFCET "Méthodes et Outils Modernes de conception et d'exploitation de la commande des procédés discontinus complexes", Montpellier, Mars 1986.

[AMA90] S. Amar, E. Castelain, J.C. Gentina, "*Modélisation des moyens de production par langages orientés objet en vue de la conception de la commande d'un système de production flexible*", Congrès CIM 90, Bordeaux, Juin 1990.

[AMA92] S. Amar, E. Craye, J.C. Gentina, "*A method of Hierarchical Specification and Prototyping of Flexible Manufacturing Systems (FMS)* ", dans les proceedings de IFAC'92 , pp. 44-49, Melbourne, Australie, Aout 1992.

[ARC91] B. Archimède, C. Berard, L. Pun, G. Doumeingts, "*Real-time dynamical diagnosis in the control of flexible manufacturing systems*", in the Proceedings of IMACS '91, pp. 627-631, Lille, 1991.

[AUS92] C. Ausfelder, E. Catelain, J.C. Gentina, "*An Object Oriented Simulation Tool to Validate the Dynamic Behaviour of Flexible Manufacturing Systems*", in the Proceedings of ESM 92, pp. 528-532, York, June 1992.

[BEN89] A. Benvéniste, Les langages synchrones : des logiciels pour la spécification et la conception des systèmes temps-réel de traitement de l'information, C2A, Rapport n°1 bis, mai 1989.

[BOI91/a] A. Boidin, "*Tendance nouvelle dans l'utilisation des vibrations pour la maintenance*", Revue Technique Apave, N°255, pp. 39-40, JUL/AOU/SEP 1991.

[BOI91/b] S. Bois, Intégration de la gestion des modes de marche dans le pilotage d'un système automatisé de production, Thèse de de Doct. de l'Université, Lille, 28 Novembre 1991.

[BOU84] A. Bourjault, Contribution à une approche méthodologique de l'assemblage automatisé : élaboration automatique des séquences opératoires, Thèse d'Etat, Université de Franche-Comté, Besançon, 1984.

[BOU88] J.P. Bourey, Structuration de la partie procédurale du système de commande de cellules de production flexibles dans l'industrie manufacturière, Thèse de Doct. de l'Université, Lille, 23 Mars 1988.

[BRU90] J. Brunet, D. Jaume, M. Labarrère, A. Rault, M. Vergé, Détection et diagnostic de panne : approche par la modélisation, Hermès, 1990.

[CAR90] J. Cardoso, Sur les réseaux de petri avec marquages flous, Thèse de Doct. de l'Université, , Université Paul Sabatier, Toulouse, Octobre 1990.

[CAS87] E. Castelain, Modélisation et simulation interactive de cellules de production flexibles dans l'industrie manufacturière, Thèse de Doct. de l'Université, Lille, 26 Février 1987.

[CET85] CETIM, La surveillance automatique des outils de coupe, Recueil de conférences, Senlis, 13 Novembre 1985.

[CEI74] Commission Electronique Internationale, Liste des termes de base, définitions et mathématiques applicables à la fiabilité, Référence 271, 1974.

[CHA83] B. Chandrasekaran, "Towards a taxonomy of problem solving types", AI Magazine, 4 (1), 1983, pp. 9-17.

[CHA89] B. Chandrasekaran, J.W. Smith, J. Sticklen, "Deep Models and Their Relation to Diagnosis" Artificial Intelligence in Medecine, Vol. 1, 1989, pp. 29-40.

[COM90] M. Combacau, M. Courvoisier, "A hierarchical and modular structure for FMS control and monitoring", in the proceedings of 1st International Conference on AI Simulation and Planning in high autonomy systems, Tucson, Arizona, Mars 1990.

[COM91] M. Combacau, "Commande et surveillance des systèmes à événements discrets complexes : application aux ateliers flexibles", Thèse de Doct. de l'Université, Université Paul Sabatier, Toulouse 1991.

[COR79] D. Corbeel, "Schéma de cablage et schéma de contrôle. Application à la simulation et à la gestion des processus industriels", Thèse de Doct. de Spécialité, Lille 1979.

[COR80] D. Corbeel, "Formal description of processes systems and exception handling", Mini&Micro, Proc. pp335-339, Budapest, 9-11 September 1980.

[COR84] D. Corbeel, J.C. gentina, C. Vercauter, "Adaptive Petri-nets for real-time applications", IMACS Digiteh'84, Patras (GRECE), 1984.

[CRA89] E. Craye, "De la modélisation à l'implantation automatisée de la commande hiérarchisée de cellules de production flexibles dans l'industrie manufacturière", Thèse de Doct. de l'Université, Lille, 13 Janvier 1989.

[CRU91] D. Cruette, " Méthodologie de conception des systèmes à événements discrets : application à la conception et la validation hiérarchisée de la commande de cellules flexibles de production dans l'industrie manufacturière ", Thèse de Doct. de l'Université, Lille, Février 1991.

[DEV91] P. Devémy, "Un instrument de maintenance conditionnelle au service de l'industrie", *Revue Technique Apave*, N°255, pp. 50-50, JUL/AOU/SEP 1991.

[DUB83] M.B. Dubuisson, "Surveillance multiparamètre de processus industriels complexes : l'approche reconnaissance des formes ", pp. 37-39, Janv./Fév. 1983.

[ELK92/a] S. Elkhatabi, A. Toguyéni, " Surveillance de la commande et du procédé", Rapport interne, Séminaire sur la surveillance, Lille, 23 janvier 1992.

[ELK92/b] S. Elkhatabi, D. Corbeel, J.C. Gentina, "Integration of dependability in the conception of FMS ", *Proceedings of INCOM'92*, Vol. 1, pp. 249-254, Toronto, may 1992.

[FIN85] P.K. Fink, J.C. Lusth, J.W. Duran, "A General Expert System Design for Diagnostic Problem Solving", *IEEE Transactions on Pattern Analysis And Machine Intelligence*, vol. PAMI-7, No. 5, September85.

[HAM91] S. Hammadi, " Une méthode d'ordonnancement minimisant les temps d'attente et de transit dans les systèmes de production flexibles de type job-shop ", Thèse de Doct. de l'Université, Lille, 18 décembre1991.

[HEB92] A. Hebrard, "Etude et réalisation d'un module de pilotage et d'un outil de représentation graphique appliqués à la conception d'un système de production flexible", Thèse de Doct. de l'Université, Lille , 7 Février1992.

[HEI87] T. Heinzerling, "Adaptation of the Petri-net of the pregraph defined in CASPAIM project in order to make the prototypage and a preliminary evaluation of the parallel architecture of the control in the field of C.I.M.", *Diplomarbeit TU München / IDN Lille*, 1988.

[KAP88] M. Kapusta, "Génération assistée d'un graphe fonctionnel destiné à l'élaborationstructurée du modèle de la partie commande pour les cellules de production flexibles dans l'industrie manufacturière", Thèse de Doct. de l'Université, Lille 1988.

[LAP84] J.C. Laprie, "Dependability evaluation of software systems in operation", *IEEE Trans. Soft. Eng.*, Vol. 10, N°6, November 1984.

[LAP85] J.C. Laprie, "sûreté de fonctionnement des systèmes informatiques et tolérances aux fautes : concepts de base ", TSI, Vol. 4, N°5, 1985.

[LAU91] C. Laurent, "Suivi vibratoire des machines tournantes par système informatisé", Revue Technique Apave, N°255, pp. 29-30, JUL/AOU/SEP 1991.

[LEC91] J.C. Lecoufle, G. Hatier, "L'organisation de la maintenance un préalable à une bonne efficacité", Revue Technique Apave, N°255, pp. 15-18, JUL/AOU/SEP 1991.

[LEV91] B. Lévi, "L'analyse vibratoire : du nouveau et encore du nouveau", Revue Technique Apave, N°255, pp. 27-28, JUL/AOU/SEP 1991.

[LHO91] P. Lhoste, "Surveillance des M.S.A.P. : les Atouts de la Modélisation de Comportement", Journées d'Etudes du Groupe GT2 du Pôle SED, Paris, 7 Février 91.

[MIL88] P. Millot, "Supervision des procédés automatisés et ergonomie", Collection Traité des Nouvelles Technologies, Editions Hermès, Paris, 1988.

[NIE92] E. Niel, "Aspects of working conditions and safety in the use of industrial robots : an overview", in Computer Integrated Manufacturing Systems, Butterorth-Heinemann Ltd Ed., Vol. 5, N°1, pp. 69-77, February 1992.

[PET80] J.L. Peterson, "A note on coloured Petri-nets", Information processing letters, Vol. 11, n°1, pp. 40-43, 1980.

[RAC91] Reliability Analysis Center, None Electronic Parts Reliability Data, 1991.

[SAH87] A.E.K. Saharaoui, "Contribution à la surveillance et à la commande d'atelier", Thèse de Doct. de l'Université, Toulouse 1987.

[SAH92] A.E.K. Saharaoui, "*Vers une approche de surveillance des systèmes à événements discrets* ", APII, Vol. 26, N°2, 1992

[SEY89] W. Seyfferth, An analytical approach to the fast evaluation of different configurations of Flexible Manufacturing Systems Based on the pregraph in the context of CASPAIM, Diplomarbeit TU München / IDN Lille, 1989.

[SFA89] A. Sfalcin, M. Roesch, P. Lhoste, G. Morel, "*Fiabilité, Disponibilité, et Sécurité des installations intégrant des Filtres de Comportement*", 2ème Colloque annuel du Club FIABEX, I.N. des Télécommunications, Evry, Novembre 1989.

[SIB88] C. Sibertin-Blanc, "*Le prototypage des applications interactives à l'aide des Réseaux de Petri*", Séminaire d'Informatique de l'I.I.E., Réseaux de Petri, 26-27 Octobre 1988.

[SIB90] C. Sibertin-Blanc, "*L'approche à objets pour la surveillance et la mise en oeuvre des SED*", Réunion S.E.D., GT2, CNRS, Paris, 11 Janvier 1990.

[TAR90] E. Tardif d'Hamonville, Contribution à la maintenance prédictive des machines tournantes, Thèse de Doctorat, ENSAM, Janvier 1990.

[TOG88] A.K.A. Toguyeni, Conception assistée du système de commande d'une cellule flexible, DEA de l'Université, Lille, 5 Septembre 1988.

[TOG90] A.K.A. Toguyeni, E. Craye, J.C. Gentina, "*A method of temporal analysis to perform on-line diagnosis in the context of Flexible Manufacturing System*", Proceedings of IECON'90, Vol. 1, pp. 445-450, Pacific Grove-California, november 90.

[TOG91] A.K.A. Toguyeni, E. Craye, J.C. Gentina, "*From the traitement of failures to the management of working modes*", Proceedings of IMACS MCTS 91, Vol. 1, pp. 595-601, Lille, may 1991.

[VIL88] A. Villemeur, Sûreté de fonctionnement des systèmes industriels, Ed. Eyrolles, Col. DER-EDF, Vol. 67, 1988.

[ZAN90] C. Zanne, J. F. Aubry, "*Une méthode de conception des systèmes de commande numérique pour une classe de processus rapides*", APII, Vol. 24, N°5, 1990.

CHAPITRE II : LE DIAGNOSTIC - EXPRESSION DES BESOINS.

[AMA90] S. Amar, E. Castellain, J.C. Gentina, "Modélisation des moyens de production par langages orientés objet en vue de la conception de la commande d'un système de production flexible", Congrès CIM 90, Bordeaux, Juin 1990.

[AUD90] E. Audureau, P. Enjalbert, L. Farinas del Cerro, "Logique temporelle, Sémantique et validation de programmes parallèles", Ed. Masson, Col. ERI, 1990.

[AYG86] P. Aygalinc, Application de la reconnaissance de formes à l'aide au diagnostic médical. Sélection multi-critère de variables explicatives, Thèse de Doct. de l'Université, Lille I, 1986.

[BES89] H. Bestougeff, G. Ligozat, "*Outils logiques pour le traitement du temps, de la linguistique à l'intelligence artificielle*", Ed. Masson, Col. ERI, 1989.

[BRU90] J. Brunet, D. Jaume, M. Labarrère, A. Rault, M. Vergé, Détection et diagnostic de panne : approche par la modélisation, Hermès, 1990.

[CAR90] J. Cardoso, Sur les réseaux de Petri avec marquage flou, Thèse de Doctorat, Université Paul Sabatier, Octobre 1990.

[CHA83/a] B. Chandrasekaran, "*Towards a taxonomy of problem solving types*", AI Magazine, 4 (1), 1983, pp. 9-17.

[CHA83/b] B. Chandrasekaran, S. Mittal, "*Deep compiled knowledge approaches to diagnostic problem-solving*", in Int. J. of Man-Machine Studies, Vol.19, 1983, pp. 425-436.

[CHA86] B. Chandrasekaran, J. Josephon, A. Keuneke, "*Functional Representation as a Basis For Generating Explanation*", Proc. IEEE Int'l Conf. Systems, Man and Cybenatics, 1986, pp. 726-731

[CHA89] B. Chandrasekaran, J.W. Smith, J. Sticklen, "*Deep Models and Their Relation to Diagnosis*" Artificial Intelligence in Medecine, Vol. 1, 1989, pp. 29-40.

[DEC 89] P. Declerck, M. Staroswiecki, "*Analytical redundancy in non linear interconnected systems by means of structural analysis*", in the Proceedings of AIPAC'89, Vol. 2, Nancy 1989, pp. 23-27.

[FIN85] P.K. Fink, J.C. Lusth, J.W. Duran, "*A General Expert System Design for Diagnostic Problem Solving*", IEEE Transactions on Pattern Analysis And Machine Intelligence, vol. PAMI-7, No. 5, September85.

- [FRO90] E. Froger, Diagnostic de systèmes complexes : modélisation des raisonnements, architecture multi-expert et validation., Thèse de Doctorat, Nantes, 1990.
- [KLE80] J. de Kleer, J.S. Brown, "A Mental Models of Physical Mechanisms and their Acquisition," in *Cognitive Skills and their Acquisition*, J.R. Anderson, ed., Lawrence Erlbaum Associates, Hillsdale, N.J., 1980, pp. 285-309.
- [KLE84] J. de Kleer, J.S. Brown, "A Qualitative Physics Based on Confluences", *Artificial Intelligence*, Vol. 24, 1984, pp.7-83.
- [KLE87] J. de Kleer, B.C. Williams, "Diagnosing Multiple Faults", *Artificial Intelligence*, 1987, vol. 32, n°1, pp 97-130.
- [KLE90] J. de Kleer, "Using Crude Probability Estimates to Guide Diagnosis", *Artificial Intelligence*, 1990, vol. 40, pp 381-392.
- [KRA91] M.A. Kramer, "First International Workshop on Principles of Diagnosis", *IEEE Expert*, June 1991, pp 86-91.
- [REG83] J. A. Reggia, D.S. Nau, P.Y. Wang. "Diagnostic expert systems based on a set covering model" ; *Int. J. of Man Machine Studies*, 1983, vol . 19, pp 437-460.
- [REI87] R. Reiter, "A Theory of Diagnosis from First Principles", *Artificial Intelligence*, 1987, vol. 32, n°1, pp 57-95.
- [SEM86] V. Sembugamoorthy, B. Chandrasekaran, " Functional Representation of Devices and Compilation of Diagnostic Problem Solving Systems", in J.L. Kolodner and C.K. Riesbeck, *Experience, Memory, and Reasoning (Ed.)*, Lawrence Erlbaum Associates, 1986.
- [TOR89] P. Torasso, L. Console, "Diagnostic Problem Solving, combining heuristic, approximate and causal reasoning", North Oxford Academic, 1989.
- [VIL88] A. Villemeur, "Sûreté de fonctionnement des systèmes industriels", Ed. Eyrolles, Col. DER-EDF, Vol. 67, 1988.
- [VOY87] R. Voyer, "Moteurs de systèmes Experts", Editions EYROLLES, 1987.

[YIM86] P. Yim, "Systèmes experts en maintenance industrielle : une synthèse ", in Proc. of Automatic Test Equipment'86, Paris, ATE, oct. 1986.

CHAPITRE III : LE DIGANOSTIC - MODÉLISATION ET RAISONNEMENT.

[ALL81] J.F. Allen, " An Interval-Based Representation of Temporal Knowledge ", Proceedings of International Conference on Artificial Intelligence (IJCAI), pp 221-226, Vancouver, 1981.

[ALL89] J.F. Allen, P.J. Hayes, " Moments and points in an interval-based temporal logic ", In Comput. Intell. 5, pp 225-238, 1989.

[AMA90] S. Amar, E. Castelain, J.C. Gentina, "Modélisation des moyens de production par langages orientés objet en vue de la conception de la commande d'un système de production flexible", Congrès CIM 90, Bordeaux, Juin 1990.

[BES89] H. Bestougeff, G. Ligozat, "Outils logiques pour le traitement du temps, de la linguistique à l'intelligence artificielle", Ed. Masson, Col. ERI, 1989.

[BOI91] S. Bois, "Intégration de la gestion des modes de marche dans le pilotage d'un système automatisé de production", Thèse de de Doct. de l'Université, Lille, 28 Novembre 1991.

[BOU88] J.P. Bourey, "Structuration de la partie procédurale du système de commande de cellules de production flexibles dans l'industrie manufacturière", Thèse de Doct. de l'Université, Lille, 23 Mars 1988.

[BRE91] G. Brès, " Panorama sur les Généraux Bysantins ", T.S.I., Vol. 10, N°5, 1991.

[BRU90] J. Brunet, D. Jaume, M. Labarrère, A. Rault, M. Vergé "Détection et diagnostic de panne : approche par la modélisation", Hermès, 1990.

[BUL90/a] S.A. Bull, "KOOLv2 : Guide d'utilisation (1) ", 95 F2 89SM, 1990

[BUL90/b] S.A. Bull, "KOOLv2 : Guide d'utilisation (2) ", 95 F2 24SR, 1990

[BUL90/c] S.A. Bull, "KOOLv2 : Manuel de référence", 95 F2 98SM, 1990

[CHA83/a] B. Chandrasekaran, "Towards a taxonomy of problem solving types", AI Magazine, 4 (1), 1983, pp. 9-17.

[CHA83/b] B. Chandrasekaran, S. Mittal, "Deep compiled knowledge approaches to diagnostic problem-solving", in Int. J. of Man-Machine Studies, Vol.19, 1983, pp. 425-436.

[CRA89] E. Craye, "De la modélisation à l'implantation automatisée de la commande hiérarchisée de cellules de production flexibles dans l'industrie manufacturière", Thèse de Doct. de l'Université, Lille, 13 Janvier 1989.

[DEA91] A. Dean, "Using Functionnal Models in Automatic Debugging", IEEE EXPERT, December 1991, pp 13-18.

[DIC89] Chang S.J., F. DiCesare, and G. Goldbogen, "The generation of Diagnostic heuristics for Automated error recovery in Manufacturing Workstations," Proceeding of the 1989 IEEE International Conference on Robotics and Automation, pp 38-43, September 1989

[FAH91] A. Farah, " Détection et diagnostic de pannes dans les ateliers de production flexibles de l'industrie manufacturière", Rapport de DEA, Université de Lille1, Juillet 1991.

[FRO90] E. Froger, " Diagnostic de systèmes complexes : modélisation des raisonnements, architecture multi-expert et validation.", Thèse de Doctorat, Nantes, 1990

[GON79] M. Gondran , M. Minoux, " Graphes et Algorithmes", Editions Eyrolles, 1979.

[KLE80] J. de Kleer, J.S. Brown, "A Mental Models of Physical Mecanisms and their Acquisition," in *Cognitive Skills and their Acquisition*, J.R. Anderson, ed., Lawrence Erlbaum Associates, Hillsdale, N.J., 1980, pp. 285-309.

[MAS89] G. Masini, A. Napoli, D. Colnet, D. Léonard, K. Tombre, "Les langages à objets : langages de classes, langages de frames, langages d'acteurs", Editions InterEditions, 1989.

[RAC91] Relibility Analysis Center, None Electronic Parts Reliability Data, 1991.

[SEM86] V. Sembugamoorthy, B. Chandrasekaran, " Functional Representation of Devices and Compilation of Diagnostic Problem Solving Systems", in J.L. Kolodner and C.K. Riesbeck, *Experience, Memory, and Reasoning (Ed.)*, Lauwrence Erlbaum Associates, 1986.

[STA91/a] M. Staroswiecki, "*Présentation des travaux du groupe Actionneurs Intelligents*", APIST-Journée d'Instrumentation Intelligente, Paris, 18 avril 1991.

[STA91/b] M. Staroswiecki, "*Fonctionnalités internes des actionneurs dans un système à intelligence distribuée*", Automation 91, Paris la Défense, 14-16 mai 1991.

[TOG90] A.K.A. Toguyeni, E.Craye, J.C. Gentina, "*A method of temporal analysis to perform on-line diagnosis in the context of flexible manufacturing system*", Proceedings of IECON'90, November 90, Vol. 1, pp 445-450, Pacific Grove-California.

[TOG91] A.K.A. Toguyeni, E. Castelain, E.Craye, " *From the treatment of failure to the management of working degraded modes*", Proceedings of IMACS'91, May 91, pp 595-601, Lille-France.

[VIL88] A. Villemeur, Sûreté de fonctionnement des systèmes industriels, Ed. Eyrolles, Col. DER-EDF, Vol. 67, 1988.

CHAPITRE IV : SPECIFICATION FONCTIONNELLE DE LA COMMANDE D'UN ATELIER FLEXIBLE, DANS LE CADRE DE L'INTEGRATION DES MECANISMES DE RECOUVREMENT DES DEFAILLANCES DU PROCEDE

[BOI91] S. Bois, "Intégration de la gestion des modes de marche dans le pilotage d'un système automatisé de production", Thèse pour l'obtention du grade de Docteur de l'Université, Lille1, novembre 1991.

[BOU87] D. Bouteille, N. Bouteille, S. Chantreuil, R. Collot, J.P. Frachet, H. Le Gras, C. Merlaud, J. Selosse, A. Sfar, "Les automatismes programmables", Cepadues-Edition, Septembre 1987.

[BOU88] J.P. Bourey, "Structuration de la partie procédurale du système de commande de cellules de production flexibles dans l'industrie manufacturière", Thèse de Doct. de l'Université, Lille, 23 Mars 1988.

[COM91] M. Combacau, " Commande et surveillance des systèmes à événements discrets complexes : application aux ateliers flexibles", Thèse pour l'obtention du grade de Docteur de l'Université, Université Paul Sabatier, décembre 1991.

[CRA89] E. Craye, "De la modélisation à l'implantation automatisée de la commande hiérarchisée de cellules de production flexibles dans l'industrie manufacturière", Thèse pour l'obtention du grade de Docteur de l'Université, Lille1, janvier 1989.

[CRU91] D. Cruette, Méthodologie de conception des systèmes complexes à événements discrets : application à la conception et la commande de cellules flexibles de production dans l'industrie manufacturière, Thèse pour l'obtention du grade de Docteur de l'Université, Lille1, février1991.

[HAM91] S. Hammadi, Une méthode d'ordonnancement minimisant les temps d'attente et de transit dans les systèmes de production flexibles de type job-shop, Thèse de Doct. de l'Université, Lille, 18 décembre1991.

[HUV92] B. Huvenoit, E. Craye, J.C. Gentina, "*Elaboration de la commande de cellules de production flexibles dans l'industrie manufacturière*", dans Automatisation Industrielle, Montréal, Juin 1992.

[MAS89] G. Masini, A. Napoli, D. Colnet, D. Léonard, K. Tombre, Les langages à objets : langages de classes, langages de frames, langages d'acteurs, Editions InterEditions, 1989.

[PUJ87] G. Pujolle, D. Seret, D. Dromard, E. Horlait, Réseaux et Télématique, Tome1, Editions Eyrolles, 1987.

[SAH87] A.E.K. Sahraoui , Contribution à la surveillance et à la commande d'atelier, Thèse pour l'obtention du grade de Docteur de l'Université, Université Paul Sabatier, 1987.

[SAH92] A.E.K. Sahraoui, " *Vers une approche globale de la surveillance des systèmes à événements discrets*", APII, Vol. 26 -2-, pp 91-106, 1992

[TAW90] R. Tawegoum, Le routage des pièces par la gestion dynamique des chemins reliant les lieux physiques d'un atelier flexible, Mémoire de DEA, Université de Lille1, Juillet 1990.

[TOG91] A.K.A. Toguyeni, E. Craye, J.C. Gentina, " *From the traitement of failures to the management of working modes* ", Proceedings of IMACS MCTS 91, Vol. 1, pp. 595-601, Lille, may 1991.

ANNEXES

[DAV84] R.Davis, " Diagnosis Reasoning Based on Structure and Behavior " , Artificial Intelligence, Vol.24, No. 3, 1984, pp.347-410.

[FRO90] E. Froger, " Diagnostic de systèmes complexes : modélisation des raisonnements, architecture multi-expert et validation.", Thèse de Doctorat, Nantes, 1990.

[GEN84] M.R. Genesereth, "The Use of Design Descriptions in Automated Diagnosis" , Artificial Intelligence, 1984, vol. 24, pp 411-436.

[KLE80] J. de Kleer, J.S. Brown, "A Mental Models of Physical Mecanisms and their Acquisition," in *Cognitive Skills and their Acquisition*, J.R. Anderson, ed., Lawrence Erlbaum Associates, Hillsdale, N.J., 1980, pp. 285-309.

[KLE84] J. de Kleer, J.S. Brown, " A Qualitative Physics Based on Confluences " , *Artificial Intelligence* , Vol. 24, 1984, pp.7-83.

[KLE87] J. de Kleer, B.C. Williams, "Diagnosing Multiple Faults", Artificial Intelligence, 1987, vol. 32, n°1, pp 97-130.

[REI87] R. Reiter, "A Theory of Diagnosis from First Principles", Artificial Intelligence, 1987, vol. 32, n°1, pp 57-95.

[SIB88] C. Sibertin-Blanc, "Le prototypage des applications interactives à l'aide des Réseaux de Petri", Séminaire d'Informatique de l'I.I.E., Réseaux de Petri, 26-27 Octobre 1988.

[VIL88] A. Villemeur, "Sûreté de fonctionnement des systèmes industriels", Ed. Eyrolles, Col. DER-EDF, Vol. 67, 1988.

**TABLE DES
MATIERES**

AVANT-PROPOS.....	5
SOMMAIRE.....	9

INTRODUCTION GÉNÉRALE

INTRODUCTION GENERALE.....	15

CHAPITRE I

Introduction.....	21
I. Positionnement et contexte du problème.	22
I.1 Introduction.....	22
I.2 Place de la surveillance dans la méthodologie CASPAIM.	22
I.2.1 Un peu d'histoire.....	22
I.2.2 Présentation des principaux travaux de CASPAIM II.	24
I.2.2.1 Elaboration de la commande de coordination [CRU91].	24
I.2.2.1.1 Principe	24
I.2.2.1.2 Les étapes de la méthodologie CASPAIM-2	25
I.2.2.1.3 Gammes logiques et gammes opératoires.....	27
I.2.2.2 Conception de l'interface [CRU91], [HEB92].	31
I.2.3 Conclusion.....	32
I.3 Surveillance et supervision.	33
I.3.1 Systèmes de supervision des processus continus.....	33
I.3.2 Coopération homme-calculateur [MIL88].	35
I.3.2.1 Coopération verticale	35
I.3.2.2 Coopération horizontale	35
I.3.3 Conclusion	37
I.4 Surveillance et maintenance.....	38
I.4.1 Introduction.	38
I.4.2 La sûreté de fonctionnement.....	38
I.4.2.1 Introduction.....	38
I.4.2.2 Terminologie.....	39
I.4.2.3 Classification des défaillances	41

I.4.2.3.1 Classification en fonction de la rapidité de leur manifestation	42
I.4.2.3.2 Classification en fonction de l'amplitude.....	42
I.4.2.3.3 Classification en fonction de la rapidité de leur manifestation et de leur amplitude	42
I.4.2.3.4 Classification en fonction des causes.....	43
I.4.2.4 Présentation des méthodes d'analyse de la sûreté de fonctionnement	45
I.4.2.4.1 L'Analyse des Modes de Défaillances et de leurs Effets (AMDE).	46
I.4.2.4.2 La Méthode de l'Arbre des Causes (MAC).	48
I.4.3 Classification de la maintenance.	53
I.4.4 Conclusion.	54
I.5 Conclusion.....	55
II. Choix d'une politique de surveillance adaptée à la maintenance des SFP.....	56
II.1 Fonctionnalités d'un système de surveillance : terminologie et principaux outils.....	56
II.2 Les différentes formes de surveillance : définitions et classifications.....	59
II.2.1 Classification en fonction du mode d'acquisition des données.	60
II.2.1.1 Surveillance continue.....	60
II.2.1.2 Surveillance périodique.....	61
II.2.1.3 Surveillance apériodique.	61
II.2.1.4 Conclusion.....	62
II.2.2 Classification en fonction de la partie surveillée.	62
II.2.2.1 Surveillance du procédé.....	62
II.2.2.1.1 Surveillance des produits.	62
II.2.2.1.2 Surveillance des moyens de production.....	63
II.2.2.1.3 Surveillance des opérations.....	63
II.2.2.2 Surveillance de la commande.....	65
II.3 Influence de la nature du système sur le type de surveillance et de maintenance.....	70
II.3.1 Les systèmes continus.	70
II.3.2 Les systèmes discrets.	71
II.4 Conclusion.....	71
III. Structuration de la surveillance dans le contexte des ateliers flexibles.....	73
III.1 Rappels sur nos hypothèses de travail.....	73

III.2 Structuration de la surveillance dans le contexte des ateliers flexibles.....	75
III.2.1 Approche du LAAS.....	75
III.2.2 Approche du CRAN/LACN.....	80
III.2.3 Proposition de structuration dans le cadre de la méthodologie CASPAIM.....	84
III.2.3.1 Le module de détection.....	86
III.2.3.2 Le module de diagnostic.....	87
III.2.3.3 Le module de recouvrement.....	88
III.3 Conclusion.....	89
IV. Détection	90
IV.1 Introduction.....	90
IV.2 Etat de l'art.....	90
IV.3 Principe du filtrage des comptes-rendus.	96
IV.4 Modèle des opérations dans le cadre de la détection [TOG90], [TOG91].	96
IV.4.1 Modèle temporel associé.....	96
IV.4.2 Classes de symptômes.....	98
IV.4.3 Exemple : Détection des symptômes liés à la défaillance d'un axe d'avance sur une M.O.C.N.	100
IV.5 Outils temporels pour la spécification du mécanisme de filtrage- perception.....	102
IV.5.1 Les RdP temporisés.....	102
IV.5.2 Les RdP temporels [CAR90].....	104
IV.5.3 Mécanisme de filtrage par RDPI temporels [TOG91].	105
IV.5.4 Les RdP temporels Objets.....	107
IV.5.4.1 Définition formelle des RdP temporels Objets [CAR90].....	107
IV.5.4.2 Application au filtrage.	110
IV.6 Classification des symptômes.....	112
IV.7 Conclusion.....	113
Conclusion.....	114

CHAPITRE II

Introduction.....	117
I. Les formes classiques du diagnostic.....	117
I.1 Introduction.....	117
I.2 Le diagnostic procédural.....	118
I.3 Le diagnostic à partir des systèmes experts.....	119
I.4 Conclusion.....	122
II. Contraintes.....	123
II.1 Contraintes liées à une approche système expert classique.	123
II.1.1 Manque d'expertise.....	124
II.1.2 Contrainte temporelle.....	124
II.1.3 Données incomplètes.....	125
II.1.3.1 Complétude de la base de règles.....	125
II.1.3.2 Par rapport à la base de faits.....	126
II.1.3.2.1 Causes.....	126
II.1.3.2.2 Conséquences.....	128
II.2 Le diagnostic en ligne et l'incertain.....	130
II.2.1 Nature de l'incertain dans les systèmes réels [TOR89].	130
II.2.2 Différentes approches.....	130
II.2.3 Contexte de la surveillance en ligne.	131
II.3 Conclusion.....	132
III. Modélisation dans le cadre du diagnostic.....	132
III.1 Introduction : objectif de la modélisation.....	132
III.2 Classification des modèles de diagnostic.....	133
III.2.1 Modèle mathématique, modèle qualitatif.....	133
III.2.2 Modèle de représentation, modèle de connaissance [BRU90].	
.....	135
III.2.3 Modèle du comportement normal, modèle du comportement	
défaillant.....	135
III.3 Conclusion.....	135
IV. Modélisation fonctionnelle : analyse de l'existant.	136
IV.1 Introduction.....	136
IV.2 Analyse de l'existant.....	138
IV.2.1 Raisonnements causaux ou associatifs : le système CHECK	
[TOR89].....	138
IV.2.1.1 Réseau causal.....	138

IV.2.1.2	Résolution causale ou raisonnement associatif	141
IV.2.1.3	Conclusion.....	142
IV.2.2	La théorie logique.....	142
IV.2.2.1	Introduction.....	142
IV.2.2.2	Approche de Reiter [REI87],[FRO90]	143
IV.2.2.2.1	Modélisation.....	143
IV.2.2.2.2	Résolution.	144
IV.2.2.3	Conclusion.....	148
IV.2.3	Fonctionnel, structurel et comportemental.....	148
IV.2.3.1	Approche de S. Amar [AMA90].	149
IV.2.3.2	Approche de B. Chandrasekaran [CHA83/a], [CHA83/b], [CHA86], [CHA89].....	151
IV.2.3.3	Conclusion.....	157
Conclusion.....		158

CHAPITRE III

Introduction.....	161
I. Modelisation fonctionnelle pour le diagnostic en ligne.	161
I.1 Définitions des concepts de base.....	161
I.2 Modèle fonctionnel et modèle structurel : définitions et différenciation.....	164
I.3 Méthode de conception d'un modèle fonctionnel	166
I.3.1 Notion de fonction.....	166
I.3.2 Les différentes phases de la méthode.....	168
I.3.1 Exemple : modèle fonctionnel d'un centre d'usinage.....	170
I.3.4 Terminologie liée aux fonctions.....	176
I.4 Modélisation comportementale.	177
I.4.1 Introduction	177
I.4.2 Principe	178
I.4.3 Le concept de la redondance temporelle [TOG91].....	179
I.4.3.1 La redondance analytique [BRU90].	179
I.4.3.2 Enoncé du concept.....	179
I.4.4 Les Signatures Temporelles Causales (STC).....	180
I.4.4.1 Introduction au concept de STC.....	180
I.4.4.2 Structure temporelle	181
I.4.4.2.1 Choix d'une structure temporelle.	181
I.4.4.2.2 Définition des entités temporelles associées.	
.....	181
I.4.4.3 Opérateurs et prédicats.....	183
I.4.4.4 Sémantique d'une STC.....	183
I.4.5 Ecritures des STC d'après les relations d'Allen.....	184
I.4.5.1 Présentation des relations d'Allen [ALL81],[BES89].	
.....	184
I.4.5.2 Expression des contraintes temporelles.....	185
I.4.5.3 Extensions et discussions.	188
I.4.5.4 Exemple : Retour sur le convoyeur de la figure II.4.	
.....	189
I.4.6 Reconnaissance de séquences dans une approche centralisée	
[TOG90], [FAH91].....	191
I.4.7 Conclusion	193
II. Exploitation du modèle fonctionnel.	194

II.1	Positionnement des capteurs.....	195
II.1.1	Propriétés du graphe fonctionnel.	195
II.1.2	Identification et observabilité.....	196
II.1.2.1	Terminologie liée à l'identification.	197
II.1.2.2	Formalisation du problème de l'identification	198
II.1.2.3	Lien entre identification et observabilité	198
II.1.2.4	Définitions liées à l'observabilité.....	199
II.1.2.5	Propriétés à vérifier du graphe fonctionnel dans le cadre de l'identification.	200
II.1.3	Algorithme de positionnement n°1.....	203
II.2	Mécanismes de raisonnement.....	205
II.2.1	Principe de résolution.	205
II.2.2	Observabilité totale et partielle.....	205
II.2.2.1	L'observabilité totale.....	206
II.2.2.2	L'observabilité partielle.....	206
II.2.2.3	Algorithme de positionnement n°2.	207
II.2.3	Gestion d'hypothèses.	208
II.2.3.1	Nature des hypothèses générées.....	208
II.2.3.2	Stratégies d'analyses.....	209
II.2.3.3	Le principe	211
II.2.3.4	Mécanisme d'identification.....	212
II.2.3.5	Mécanisme de validation.	212
II.2.4	Mécanisme de pronostic.	213
II.2.4.1	Condition de déclenchement.	213
II.2.4.2.	Conditions d'arrêts.....	214
II.2.4.3	Mise en œuvre.....	214
II.2.5	Classification des noeuds du modèle fonctionnel.....	215
II.2.5.1	Introduction.....	215
II.2.5.2	Classification.	215
II.2.5.3	Comportement d'un noeud.....	216
II.2.6	Conclusion	219
III.	Spécifications informatiques	220
III.1	Principe	220
III.2	Hiérarchie des noeuds.....	222
III.3	Classes de symptômes et de STC.....	226
III.3.1	Classes de symptômes.	226
III.3.1.1	Méthodes personnelles à la classe "Symptome".....	226

III.3.1.2 Attributs du schéma d'instance d'un symptôme.....	226
III.3.1.3 Méthodes du schéma d'instance d'un symptôme.....	227
III.3.1.4 Démons associés aux attributs de la classe symptôme.	227
III.3.1.5 Transcription des spécifications en KOOL.....	227
III.3.2 Classe de STC.	228
III.3.2.1 Attribut du schéma d'instance.	228
III.3.2.2 Méthodes du schéma d'instance.....	229
III.3.2.3 Démons liés aux attributs.....	229
III.3.2.4 Transcription des spécifications en KOOL.....	230
III.3.3 La classe d'attribut SymptomeContraint.	231
III.3.3.1 Attributs de SymptomeContraint.	231
III.3.3.2 Transcription des spécifications en KOOL.....	231
III.4 Exemple : Spécification informatique du GF du SAS.	232
III.5 Mécanisme d'ensemble.....	238
III.6 Conclusion.....	241
Conclusion.....	241

CHAPITRE IV

Introduction.....	247
I. Le recouvrement : principe et classification.	248
I.1 Présentation du GEMMA [BOU87].	248
I.1.1 Les procédures de fonctionnement.	249
I.1.2 Les procédures d'arrêt et de remise en route.	250
I.1.3 Les procédures suite à une défaillance.....	250
I.2 Les différents aspects du recouvrement.	251
I.2.1 Classification en fonction des moyens d'actions.....	251
I.2.1.1 Moyens d'actions sur le procédé.	252
I.2.1.2 Mécanismes d'actions sur la commande.....	253
I.2.2 Classification en fonction du but et de la réactivité.	256
I.2.2.1 Recouvrement immédiat.	256
I.2.2.2 Recouvrement différé.....	257
I.2.3 Classification en fonction de l'étendue et de la durée.....	258
I.2.3.1 Recouvrement local à court terme.	259
I.2.3.2 Recouvrement global à moyen terme.	259
I.2.3.3 Recouvrement global à long terme.	259
I.2.4 Conclusion	259
II. Gestion des modes de marches et d'arrêts des SFP [BOI91].	261
II.1 Modélisation du comportement des moyens de production.....	261
II.2 Structuration et fonctionnement du gestionnaire.	265
II.3 Conclusion.....	268
III. Recouvrement au niveau commande : Exigence d'une approche fonctionnelle	269
III.1 Flexibilité et gestion des modes de marche.	269
III.2 Gestion des modes de marche et pilotage.	270
III.3 Conclusion.....	273
IV. Structuration en couches de la commande.....	274
IV.1 Classification de la flexibilité du procédé.	274
IV.2 Limites des spécifications de CASPAIM-2.	275
IV.3 Présentation du modèle OSI [PUJ87].	278
IV.4 Présentation des couches du modèle de commande.....	281
IV.4.1 Commandes actionneurs.....	282
IV.4.2 Commande des moyens de production.....	282
IV.4.3 Gestion du système de transport.	283

IV.4.3.1 Le routage.....	286
IV.4.3.2 Le contrôle de l'accessibilité directe entre lieux de stockage.	288
IV.4.3.3 La synchronisation des moyens de production.	288
IV.4.4 Allocation des moyens de production.	288
IV.4.4.1 Machines logiques.	288
IV.4.4.2 Pilotage.....	290
IV.4.4.3 Recouvrement.	291
IV.4.4.4 Conclusion.....	292
IV.4.5 Les gammes opératoires.	294
IV.5 Conclusion.....	297
V Intégration du recouvrement dans l'architecture de commande.....	298
V.1 Ordonnancement et pilotage dans le contexte de CASPAIM-2 [HAM91].	298
V.2 Nouvelle proposition adaptée à la structure en couches.	301
Conclusion.....	304

CONCLUSION GÉNÉRALE

CONCLUSION GENERALE.....	305

ANNEXES

ANNEXE A : Sûreté de fonctionnements - Définitions des principaux concepts et des paramètres associés [VIL88]	311
ANNEXE B : Les Réseau de Petri à Objets ou RdPO [SIB88].	314
ANNEXE C : Complément sur la théorie logique.....	317
Approche de R. Davis [DAV84].	317
a- Modèle de connaissance.....	317
b- Diagnostic des défaillances.....	318
Approche de M.R. Genesereth [GEN84].	320
a - Modélisation.....	320
b - Résolution.....	323
c - Conclusion.	323
Approche de J. de Kleer [KLE80], [KLE84], [KLE87], [FRO90].....	324
a - Principe.....	324
b - Modélisation	324
c - Résolution.....	325

BIBLIOGRAPHIE

Introduction générale	329
Chapitre I : La surveillance en ligne.....	329
Chapitre II : Le diagnostic - Expression des besoins.....	334
Chapitre III : Le diagnostic - Modélisation et Raisonnement.....	336
Chapitre IV : Spécification fonctionnelle de la commande d'un atelier flexible, dans le cadre de l'intégration des mécanismes de recouvrement des défaillances du procédé	339
Annexes.....	340

TABLE DES FIGURES

CHAPITRE I

Figure I.1	
Schéma de principe d'un SFP d'après [CRU91].	25
Figure I.2	
Synoptique de CASPAIM-2 d'après [CRU91]	26
Figure I.3	
Gamme logique d'usinage avec flexibilité de gamme.	28
Figure I.4	
Cellule de transformation.	29
Figure I.6	
Structuration du niveau hiérarchique pour l'intégration de la surveillance.	32
Figure I.7	
Modèle de résolution de problème adapté de Rasmussen [MIL88].	34
Figure I.8	
Répartition dynamique des tâches entre opérateur et calculateur d'après P. MILLOT [MIL88].	36
Figure I.9	
Chronogramme liant les concepts de défaillance, de panne et de réparation.	39
Figure I.10	
Synoptique du système broche/mandrin d'une M.O.C.N.	41
Figure I.11	
Exemple de défaillance en cascade.	45
Figure I.12	
Extrait de l'AMDE du système broche/mandrin d'une machine outil à broche horizontale.	47
Figure I.13	
Représentation des événements de l'arbre des causes.	49
Figure I.14	
Portes logiques utilisées pour relier les événements.	50
Figure I.15	
Exemple d'arbre de défaillance.	52
Figure I.16	
Techniques de maintenance.	54
Figure I.17	
Test d'hypothèses.	58

Figure I.18	
Suivi des efforts de coupe.	60
Figure I.19	
Positionnement des capteurs utilisés pour la surveillance d'une opération de chargement de pièce.	64
Figure I.20	
Illustration de la sûreté de séquence pour le chargement d'une machine outil.	65
Figure I.21	66
Figure I.22	
Surveillance implicite de la commande dans CASPAIM-1.....	67
Figure I.23	
Graphe de comportement du manipulateur (i).	68
Figure I.24	
Proposition de structuration de la fonction surveillance de la commande d'après [ELK92/b].	69
Figure I.25	
Schéma de synthèse relatif à la maintenance sur état du procédé	72
Figure I.26	
Discrétisation binaire d'un signal analogique	74
Figure I.27	
Modélisation d'une activité par RdPO.....	75
Figure I.28	
Structuration d'un niveau de commande-surveillance d'après M. COMBACAU [COM91].....	78
Figure 29	
Rainurage.	79
Figure I.30	
Passage d'une vision atomique à une vision moléculaire de l'interface PC-PO... ..	81
Figure I.31	
Modèle du comportement normal d'un vérin double-effet.	81
Figure I.32	
Utilisation en émulation.	82
Figure I.33	
Surveillance par prévision du comportement.	83
Figure I.34	
Structuration de la PC d'après [LHO91].....	83

Figure I.35	Structuration de la fonction surveillance et positionnement par rapport à la PC et au procédé.....	86
Figure I.36	Technique du chien de garde.	91
Figure I.37	Représentation d'une macro-action.	92
Figure I.38	Convoyeur à bande.	94
Figure I.39	Modélisation de la commande de 2 transferts exclusifs d'après CASPAIM-1.....	95
Figure I.40	Schéma de principe de la fonction filtrage-perception.	96
Figure I.41	Modèle associé à une opération [TOG90].....	97
Figure I.42	Filtrage à partir de la modélisation temporelle.	98
Figure I.43	Définition des symptômes et contexte opérationnel associé.....	99
Figure I.44	Schéma d'une jonction.	100
Figure I.45	Principe de la surveillance d'axe.	101
Figure I.46	Mécanisme de tir dans un RdP temporisé.....	102
Figure I.47	Modélisation d'un transfert par RdP temporisé au niveau des arcs d'après [SEY89].....	103
Figure I.48	Résolution d'un indéterminisme par réseau de Petri temporel d'après [CAR90].	104
Figure I.49	Mécanisme de filtrage.	106
Figure I.50	Résolution d'indéterminismes transition-substitution dus au flux de gamme....	109
Figure I.51	Mécanisme de filtrage par Réseau de Petri Temporel à Objets.....	111

CHAPITRE II

Figure II.1	
Schéma de principe du mécanisme de diagnostic	118
Figure II.2.....	120
Figure II.3	
Tableau récapitulatif des techniques d'inférence utilisées, en fonction du mode d'initialisation et du but du diagnostic.....	122
Figure II.4	
Convoyeur.....	125
Figure II.5	
Séquence de détection de symptômes.	126
Figure II.6	
Schéma d'un vérin muni de capteurs.	127
Figure II.7.....	128
Figure II.8	
Synoptique d'un vérin hydraulique.....	134
Figure II.9	
Classification des systèmes experts par approche de représentation de la connaissance.....	137
Figure II.10	
Graphe causal.	138
Figure II.11	
Structure d'un arc CAUSAL.	139
Figure II.12	
Tableau récapitulatif des relations entre noeuds.....	140
Figure II.13.	143
Figure II.14	
Un HS-tree simplifié pour le circuit électronique [REI87].....	147
Figure II.15	
Classification fonctionnelle d'après [AMA90].	149
Figure II.16	
Association de fonctions aux composants d'un centre d'usinage d'après [AMA90].....	150
Figure II.17	
Modèle comportemental d'un moteur à double sens de marche.....	151

Figure II.18	
Sonnerie électronique.....	152
Figure II.19	
Composantes d'une représentation fonctionnelle d'après Chandrasekaran.	154
Figure II.20	
Modélisation de la fonction "sonner".....	155
Figure II.21	
Modélisation par RdPI du "comportement1" relatif à la fonction sonner de la sonnerie.	156

CHAPITRE III

Figure III.1.....	162
Figure III.2.....	163
Figure III.3	
Sémantique des liens inter-composants.	164
Figure III.4.....	167
Figure III.5	
Méthodologie de conception du modèle fonctionnel d'un système existant.....	168
Figure III.6.....	171
Figure III.7	
Schéma du SAS et modèle structurel associé.....	171
Figure III.8.....	172
Figure III.9.....	174
Figure III.10.....	175
Figure III.11	
Extrait du GF d'un système de production.	176
Figure III.12	
Modélisation du comportement d'une fonction de transfert.....	177
Figure III.13.....	178
Figure III.14.....	181
Figure III.15.....	182
Figure III.16.....	182
Figure III.17	
Interprétation des contraintes.	184
Figure III.18	
Relations d'Allen.....	185
Figure III.19.....	185
Figure III.20.....	186
Figure III.21/a.....	186
Figure III.21/b.....	187
Figure III.22.....	187
Figure III.23	
Occurrence d'un symptôme de type I a l'intérieur de l'intervalle de validation	
d'un symptôme de type II.....	188
Figure III.24	

Modèle temporel utilisé par le module de détection.	189
Figure III. 25	
Séquence d'entrée.....	192
Figure III.26	
Reconnaissance de séquences.	193
Figure III.27.	197
Figure III.28.	200
Figure III.29.....	202
Figure III.30	
Diagnostic local lié au concept observabilité partielle.....	207
Figure III.31	
Extrait du graphe fonctionnel du SAS du CU60.....	208
Figure III.32.	210
Figure III.33.....	210
Figure III.34.	211
Figure III.35.	211
Figure III.36	
Les étapes du pronostic.	213
Figure III.37.	218
Figure III.38.....	218
Figure III.39	
Tableau récapitulatif des comportements par type de noeuds.....	219
Figure III.40	
Schéma de principe.	221
Figure III.41	
Hiérarchie des noeuds.....	226
Figure III.42.....	236
Figure III.43	
Spécification des mécanismes d'un noeud initiateur.....	240

CHAPITRE IV

Figure IV.1	
Le GEMMA.	249
Figure IV.2	
Evolution des efforts de coupe en fonction du degré d'usure d'un outil.	252
Figure IV.3	
Reconfiguration dynamique dans CASPAIM-1	255
Figure IV.4	
Système de convoyage des pièces.	257
Figure IV.5	
Réactivité du recouvrement en fonction du but.	258
Figure IV.6	
Graphe de comportement d'une machine effective complexe d'après [BOI91].	262
Figure IV.7	
Graphe de comportement d'une machine virtuelle.	264
Figure IV.8	
Exemple de représentation finale d'un atelier d'après [BOI91].	265
Figure IV.9	
Structuration du Niveau Hiérarchique et du gestionnaire des modes de marche d'après [BOI91].	266
Figure IV.10	
Contrainte de coopération.	267
Figure IV.11	
Organigramme de fonctionnement du gestionnaire adapté d'après [BOI91].	268
Figure IV.12	269
Figure IV.13	272
Figure IV.15	
Réseau maillé.	279
Figure IV.16	
Modèle OSI et MiniMap.	281
Figure IV.17	
Décomposition en couches de la commande d'un atelier flexible.	281
Figure IV.18	
Commande des moyens de production.	282
Figure IV.19	283

Figure IV.20	
Réseau de transport d'un SFP.....	284
Figure IV.21	
Table des liaisons.	285
Figure IV.23	
Gamme opératoire d'usinage linéaire.	294
Figure IV.24	
Gamme opératoire flexible.	295
Figure IV.25	
Passage de données entre la couche gestion des gammes opératoires et la couche gestion de l'allocation.....	297
Figure IV. 26	
Affectation - Ordonnancement.....	299
Figure IV.27	
Notion de marge d'opération.....	299
Figure IV.28	
Utilisation de la flexibilité.....	302
Figure IV.29	
Intégration du recouvrement dans l'architecture de commande-surveillance d'un SFP.....	303

ANNEXES

Figure A.1

Représentation temporelle des principaux paramètres de fiabilité d'après
A.VILLEMEUR [VIL88]..... 313

Figure C.1

Principaux termes utilisés dans la description de la structure d'un système.... 317

Figure C.2

Modélisation structurelle d'un additionneur 4 bits..... 318

Figure C.3

Exemple de diagnostic utilisant la violation des contraintes..... 319

