Contribution à l'élaboration d'une méthodologie de spécification, de vérification et de génération semi-automatique d'interfaces homme-machine : Application à l'outil Ergo-Conceptor+

Meriem Riahi

Numéro d'ordre : 04-18

THESE

pour l'obtention du

Doctorat de l'Université de Valenciennes et du Hainaut-Cambrésis

Spécialité Automatique et Informatique des Systèmes Industriels et Humains

Mention Informatique

Par

Mme Meriem RIAHI épouse CHEBBI

Contribution à l'élaboration d'une méthodologie de spécification, de vérification et de génération semi-automatique d'interfaces homme-machine : Application à l'outil Ergo-Conceptor+

Pour une soutenance prévue le 28 Septembre 2004 devant le jury composé de :

MM.

Patrick GIRARD	(Professeur, LISI-ENSMA, Université de Poitiers)	Rapporteur
Christophe KOLSKI	(Professeur, LAMIH, UVHC)	Co-Directeur de Thèse
Ahmed MAHJOUB	(Professeur, Faculté des Sciences de Tunis)	Président
Mohamed MOALLA	(Professeur, Faculté des Sciences de Tunis)	Co-Directeur de Thèse
Faouzi MOUSSA	(Maître-Assistant, Faculté des Sciences de Tunis)	Co-Directeur de Thèse
Pascal YIM	(Professeur, Ecole Centrale de Lille)	Rapporteur

Résumé :

La recherche effectuée vise l'élaboration d'une démarche globale de conception des interfaces homme-machine qui s'appuie sur l'utilisation d'outils formels pour l'identification, la spécification et l'analyse du comportement des objets intervenant dans les interactions homme-machine. Les problèmes ergonomiques posés, liés à l'utilisation de l'interface graphique, concernent principalement *quoi* présenter à l'opérateur (en fonction du contexte de fonctionnement et des tâches opérateurs) et *comment* le présenter.

L'approche proposée vise à satisfaire les points suivants : (1) une analyse globale du système homme-machine (SHM) couvrant l'analyse de la tâche, des fonctionnements et des dysfonctionnements du système à contrôler, permettant la déduction des besoins informationnels de l'opérateur (BIO) ; (2) une modélisation formelle du comportement du SHM permettant la gestion automatique du dialogue et la vérification des bonnes propriétés de l'interface ; (3) une spécification graphique de l'interface intégrant les recommandations ergonomiques nécessaires de représentation et regroupement des objets au niveau des différentes vues de l'interface ; (4) une génération semi-automatique de l'interface à partir des spécifications préétablies.

Pour répondre à ces besoins, on propose une combinaison de méthodes d'analyse et outils formels de modélisation. On procède par une analyse systémique utilisant une combinaison de méthodes d'analyse en mode de fonctionnement normal (SADT) et en mode de dysfonctionnement (AMDE et AdD) pour une première modélisation du système homme-machine. On utilise ensuite, un formalisme basé sur les réseaux de Petri Interprétés (RdPI) pour la modélisation du comportement du SHM, exprimant les différentes interactions possibles entre l'opérateur et l'interface et permettant ainsi la déduction des objets de l'interface et la gestion automatique du dialogue homme-machine. Finalement, un outil manipulant les recommandations ergonomiques (Tools For Working With GuideLines) est proposé. Il se base sur un système à base de connaissances intégrant les recommandations ergonomiques nécessaires pour la matérialisation graphique des différents objets identifiés lors de l'analyse du système pour leur affichage et leur manipulation par l'utilisateur.

Mots clés : Interaction Homme-Machine (IHM), Système Homme-Machine (SHM), Spécification formelle, Réseaux de Petri Interprétés (RdPI), Génération semi-automatique d'interface.

Abstract :

This research work is carried out in the context of design and semi-automatic generation of graphical Interfaces for process control. It studies a new approach for ergonomic Interface design, using a formal language for the specification of interface objects and their behaviour. In fact, several ergonomic problems related to the use of grafical user interfaces are noticed. These problems mainly concern *WHAT* to present to the operators according to the functional context and the corresponding human tasks, and *HOW* to present the information graphically

The proposed approach aims to cover these main points: (1) an analysis of the HMS to identify the different functioning states of the system and the possible interventions of the human operator, expressing the interaction of the operator within the interface according to different functioning contexts of the system allowing the deduction of the user requirements; (2) a formal modelling of the Human-Machine System behaviour making possible the validation of the specifications before going on to the generation of the interface ; (3) a graphical specification of the interface integrating ergonomic criteria for the presentation and the dialogue of the interface; (4) a semi-automatic generation of the interface regarding to the generated specifications.

For this fact, we propose to use a combination of methods and formal modelling tools. We perform a hierarchical decomposition of the HMS using SADT method and we analyse the dysfunctionning of the system with FMEA and Fault Tree methods. The Interpreted Petri nets are then proposed for modelling the human machine-system behaviour. Finally, we propose to use, for UI generation, a model-based tool which is able to decide on the appropriate displays. It takes into consideration, on the one hand the results of the previous analysis and on the other hand, specific formalised guidelines stored in its knowledge bases.

Key words : Human-Computer Interaction, Man-Machine System, Formal specification, Interpreted Petri Nets, Semi-Automatic Interface generation.

Avant-Propos

Le travail présenté dans ce mémoire a été effectué en collaboration entre le Laboratoire d'Automatique et de Mécanique Industrielles et Humaines (LAMIH), U.M.R C.N.R.S 8530, à l'Université de Valenciennes et du Hainaut-Cambrésis dans le groupe de recherche « Raisonnement Automatique et Interaction Homme-Machine » (RAIHM) dirigé par le Professeur Christophe KOLSKI et le Laboratoire d'Informatique, Productique et Parallélisme (LIP2) de la Faculté des Sciences de Tunis sous la direction du Professeur Mohamed MOALLA, et de Monsieur Faouzi MOUSSA.

Je tiens, tout d'abord, à exprimer ma profonde gratitude au Professeur Christophe KOLSKI et au Professeur Mohamed MOALLA, pour m'avoir accueillie et intégrée dans leurs équipes de recherche. Qu'ils trouvent, ici, l'expression de ma profonde reconnaissance pour leur aide, les judicieux conseils et les remarques pertinentes qu'ils m'ont prodigués tout le long de la réalisation de ce travail.

Mes remerciements les plus vifs vont naturellement à Monsieur Faouzi MOUSSA qui, en plus de son soutien amical constant, m'a encadrée, guidée et conseillée avec enthousiasme et rigueur tout au long de cette thèse. Je lui suis en outre très reconnaissante pour m'avoir fourni les moyens de mener mes recherches dans d'excellentes conditions humaines et matérielles.

Je suis très reconnaissante envers Monsieur Patrick GIRARD, Professeur à l'Université de Poitiers (LISI-ENSMA), et Monsieur Pascal YIM, Professeur à l'Ecole Centrale de Lille, de m'avoir fait l'honneur d'examiner ce travail et d'en être les rapporteurs.

Je remercie également le Professeur Ahmed MAHJOUB, Président Directeur Général de Tunisie Télécom, pour l'honneur qu'il me fait d'accepter de présider le jury de soutenance de ce mémoire.

Enfin, je ne saurais oublier tous ceux qui m'ont aidée, de près ou de loin, à poursuivre et achever ce travail. Je m'adresse spécialement :

- à mes sœurs Hasna, Hédia et Karima pour leur soutien, leur compréhension et leurs encouragements durant toutes les années de mes études,
- à mon cher frère Habib à qui je souhaite beaucoup de succès et de bonheur dans sa vie,
- à mon mari Naoufel pour sa patience, sa tolérance et ses encouragements et pour m'avoir supporté ces derniers mois,
- à ma belle famille, en particulier, à Souad, Haifa et Chawki pour leur soutien moral et matériel.
- à toutes mes amies pour leur permanent soutien et encouragement : Aida, Asma, Fatma, Héla, Houda et Sonia.

A mes parents...

Sommaire

Introduction générale **1**

Chapitre 1 : Aspects méthodologiques liés à l'analyse, la conception et la génération d'IHM dans les systèmes automatisés de production **3**

I. Du rôle de l'homme dans la supervision et la conduite des systèmes automatisés de production 4

II. Aspects généraux de l'interaction homme-machine en salle de contrôle 6
II.1- La modélisation de l'opérateur humain 6
II.1.1- Le modèle du processeur humain 7
II.1.2- La théorie décisionnelle de Rasmussen 8
II.1.3- La théorie de l'action de Norman 9
II.2- L'analyse de la tâche humaine 10
II.2.1- Travaux de Scapin 11
II.2.2- Travaux de Abed et al. 12
II.2.3- Travaux de Paterno et al. 13
II.3- Connaissances provenant de l'ergonomie cognitive pour la conception et l'évaluation des IHM 15

III. Des cycles de vie classiques aux cycles de vie enrichis sous l'angle de l'IHM 17
III.1- Modèles généraux de génie logiciel 17
III.1.1- Modèle en cascade 19
III.1.2- Modèle en V 19
III.1.3- Modèle en spirale 20
III.1.4- Modèle par incréments 21
III.1.5- Synthèse sur les modèles classiques 22
III.2- Modèles enrichis pour les IHM 22
III.2.1- Modèle de Long 22
III.2.2- Modèle en étoile 23
III.2.3- Modèle ∇ (Nabla) 23
III.2.4- Conclusion sur les modèles classiques et enrichis 24
III.3- Démarche globale en U dédiée aux systèmes industriels complexes 24
III.3.1- Phase conceptuelle descendante de conception d'un SHM 25
III.3.2- Phase ascendante d'évaluation d'un SHM 28

IV. Approches et méthodologies de conception des IHM proposées dans la littérature 30
IV.1- Travaux de Palanque et Bastide : le modèle ICO 30
IV.2- Travaux de De Rosis : le formalisme XDM 31
IV.3- Travaux de Tabary et al. : la méthode TOOD 32
IV.4- Travaux de Ait Ameur et al. : la méthode B 33
IV.5- Travaux de Rodriguez : le projet ALACIE 34
IV.6- Travaux de Bodart et al. : le projet TRIDENT 35
IV.7- Travaux de Moussa : le système Ergo-Conceptor 36

V. Synthèse et identification des nouveaux besoins 37
V.1- Tableau récapitulatif 37
V.2- Guide méthodologique 38

Conclusion 39

Chapitre 2 : Méthodes, outils, modèles et formalismes de base venant en support des approches méthodologiques 41

I. Méthodes et techniques d'analyse des systèmes en mode de fonctionnement normal 42
I.1- Approches fonctionnelles (cartésiennes) 42
I.1.1- La méthode SA 42
I.1.2- La méthode SADT 43
I.1.3- La méthode SART 44
I.1.4- La méthode des graphes de fluence 44
I.2- Approches systémiques 45
I.3- Approches objets 46

II. Méthodes et techniques d'analyse des systèmes en mode de fonctionnement anormal 47
II.1- Méthode d'Analyse Préliminaire des Dangers (APD) 47
II.2- Méthodes d'analyse et regroupement des pannes 48
II.2.1- La méthode AMDE 48
II.2.2- La méthode MCPR 49
II.2.3- La méthode MTV 49
II.3- Méthodes d'analyse détaillée causes/effets d'une panne 50
II.3.1- La méthode MDS 50
II.3.2- La méthode AdD 50
II.3.3- La méthode MACQ 51
II.3.4- la méthode MDCC 52

III. Outils formels de modélisation utilisés dans les systèmes homme-machine 52
III.1- Apports et limites des modèles formels 52
III.2- Outils formels proposés pour la spécification des IHM 54
III.2.1- La logique temporelle 54
III.2.2- Les langages ensemblistes 55
III.2.3- Les automates d'état fini 56
III.2.4- Le langage Lotos 57
III.2.5- Le langage Lustre 57
III.2.6- Les Réseaux de Petri 58

IV. Synthèse et choix des méthodes et outils 59

Conclusion 63

Chapitre 3 : Approche proposée pour la spécification, la vérification et la 64

génération semi-automatique d'IHM dans les systèmes industriels complexes

I. Présentation générale de l'approche proposée 65

II. Présentation détaillée de l'approche 66
II.1- Analyse du SHM 66
II.2- Modélisation de l'interaction homme-machine 68
II.2.1- Composition séquentielle 69
II.2.2- Composition parallèle 70
II.2.3- Composition alternative 72
II.2.4- Composition choix opérateur 73
II.2.5- Composition itérative 75
II.2.6- Composition de fermeture 76
II.2.7- Principe de construction du modèle global de l'interaction homme-machine 76
II.3- Déduction des Besoins informationnels des opérrateurs (BIO) 78
II.4- Spécification de l'interface 79
II.4.1- Spécification de la présentation assistée par un système à base de connaissances 79
II.4.2- Spécification du dialogue 82
II.5- Vérification des propriétés de l'interface 85
II.5.1- Les propriétés à vérifier 85
II.5.2- Principe de vérification 86
II.6- Génération de l'interface 88
II.6.1- La génération graphique 88
II.6.2- la gestion du dialogue homme-machine 89
II.7- Intégration et évaluation de l'IHM intégrée 91

Conclusion 91

Chapitre 4 : Application de la méthodologie sur une application industrielle : de l'analyse du SHM à la vérification des propriétés de l'IHM **93**

I. Présentation de l'application industrielle 94

II. Application de la démarche 96
II.1- Analyse du SHM 96
II.1.1- Décomposition fonctionnelle du système par SADT 96
II.1.2- Analyse des dysfonctionnements du système 99
II.1.2.1- Analyse par la méthode AMDE 99
II.1.2.2- Elaboration de l'arbre des défaillances 106
II.1.3- Analyse de la tâche opérateur 113
II.2- Modélisation de l'interaction homme-machine par les RdP 114
II.3- Déduction des BIO 117
II.4- Spécification de l'interface 118
II.5- Vérification des propriétés de l'interface 118

Conclusion 118

Chapitre 5 : Conception et réalisation de l'outil Ergo-conceptor+ supportant la génération de l'IHM et perspectives de recherche 119

I. Architecture conceptuelle et fonctionnelle de l'outil Ergo-Conceptor+
I.1- Rappel de l'architecture informatique de l'outil Ergo-Conceptor 120
I.2- présentation de l'outil Ergo-Conceptor+ 120
I.3- Ergo-Conceptor versus Ergo-Conceptor+ 121
II. Conception détaillée de l'outil Ergo-Conceptor+ 123
II.1- Description du procédé 124
II.2- Définition du modèle de l'interaction homme-machine 124
II.3- Génération graphique de l'IHM 125
II.4- Gestion de la dynamique du dialogue homme-machine 126
II.5- Extrait de la modélisation objet avec UML 128
II.5.1- Diagramme de classes 131
II.5.2- Diagramme de séquence 131
II.6- Environnement technique de réalisation 133
II.6.1- Environnement matériel 134
II.6.2- Environnement logiciel 134
II.7- Conclusion sur l'outil Ergo-Conceptor+ 134

III. Perspectives de recherche 135
III.1- Intégration de l'approche multi-agents dans la démarche de spécification et de génération des IHM 135
III.2- Couplage d'un système de prédiction et d'aide à la conduite 137
III.3- Prise en considération de l'aspect profil utilisateur 137
III.4- Validation et évaluation sur cas réel 137

Conclusion 137

Conclusion générale 139

Références bibliographiques 141

Liste des Figures

Figure 1.1 : Exemple de salles de contrôle 4
Figure 1.2 : Exemples de vues de supervision 5
Figure 1.3 : Architecture d'un système homme-machine 5
Figure 1.4 : Modèle du processeur humain 7
Figure 1.5 : L'échelle de décision de Rasmussen 8
Figure 1.6 : La théorie de l'action de Norman 9
Figure 1.7 : Approche d'analyse de la tâche selon Stammers 11
Figure 1.8 : Une arborescence de tâches MAD 12
Figure 1.9 : Une boîte SADT relative à une tâche humaine interactive dont le fonctionnement est décrit par un RdPS 13
Figure 1.10 : MCD d'une modélisation CTT 15
Figure 1.11 : Démarche traditionnelle de développement d'un logiciel 18
Figure 1.12 : Modèle en cascade (waterfall) 19
Figure 1.13 : Modèle en V 20
Figure 1.14 : Modèle de la spirale 21
Figure 1.15 : Modèle par incréments 22
Figure 1.16 : Modèle proposé par Long et al. 23
Figure 1.17 : Modèle en étoile 23
Figure 1.18 : Modèle ∇ (Nabla) 24
Figure 1.19 : Phase descendante de conception du système Homme-Machine 26
Figure 1.20 : Phase ascendante d'évaluation du système Homme-Machine 29
Figure 1.21 : Le modèle ICO 30
Figure 1.22 : Structure Méthodologique de TRIDENT 35
Figure 1.23 : Démarche d'Ergo-Conceptor 36
Figure 1.24 : Squelette méthodologique de l'approche proposée 39

Figure 2.1 : Exemple de DFD pour une fonction analysée par SA 43
Figure 2.2 : Structure hiérarchique d'une analyse par SADT 43
Figure 2.3 : Exemple de graphe de fluences 44
Figure 2.4 : Exemple d'un MCD pour une agence de location de films vidéo 45
Figure 2.5 : Exemple de diagramme de classe des objets participant à une collaboration "ouvrir marché" modélisé par UML 47
Figure 2.6 : Automate à états finis décrivant une opération de calcul dans une calculatrice 56
Figure 2.7 : Arbre en Lotos 57
Figure 2.8 : Exemple de l'état d'un RdP avant et après le franchissement d'une transition 58
Figure 2.9 : Schéma récapitulatif des principales méthodes développées pour la sûreté de fonctionnement des systèmes industriels 62

Figure 3.1 : Squelette méthodologique de l'approche proposée 65
Figure 3.2 : cas (a) d'un sous-système sans nécessité de duplication, et (b) d'un sous-système dupliqué 67
Figure 3.3 : Réseau de Petri Interprété (RdPI) 68
Figure 3.4 : Structure élémentaire modélisant l'état d'un opérateur face à une action élémentaire 69
Figure 3.5 : Composition séquentielle 70
Figure 3.6 : Structure PAR1 71

Figure 3.7 : Structure PAR2 71
Figure 3.8 : Composition parallèle 71
Figure 3.9 : Structure ALT 72
Figure 3.10 : Composition Alternative 72
Figure 3.11 : Structure élémentaire de décision du choix opérateur 73
Figure 3.12 : Composition de choix opérateur exclusif 73
Figure 3.13 : Composition Choix opérateur inclusif 74
Figure 3.14 : Réseau I 75
Figure 3.15 : Composition itérative 75
Figure 3.16 : Structure de fermeture 76
Figure 3.17 : Composition de fermeture 76
Figure 3.18 : Exemple d'un modèle d'interaction homme-machine 77
Figure 3.19 : Exemple d'un modèle de l'interaction homme-machine avec les BIO 79
Figure 3.20 : Processus de spécification de la présentation de l'interface 81
Figure 3.21.a : Exemples de recommandations ergonomiques pour la génération des vues graphiques 81
Figure 3.21.b : Exemples de recommandations ergonomiques pour la sélection des objets de l'interface 81
Figure 3.22 : Structures de contrôle des objets de l'interface 83
Figure 3.23 : Processus de communication entre les différents RdP 84
Figure 3.24 : Suppression d'une SEAO 87
Figure 3.25 : Système à base de connaissances de l'outil Ergo-Conceptor+ 89
Figure 3.26 : Joueur ou interpréteur de RdPI 90
Figure 3.27 : Structures des matrices « Pre » et « Post » 90

Figure 4.1 : Façonnage et remplissage de godets 94
Figure 4.2 : Décomposition du système par SADT (Actigramme A0) 96
Figure 4.3 : Décomposition du système par SADT (Actigramme A1) 97
Figure 4.4 : Elaboration de l'AdD pour l'événement redouté 'arrêt système' 107
Figure 4.5 : Développement de l'AdD pour les deux événements : 'Pb doasge P' et 'Pb dosage S' 108
Figure 4.6 : Développement de l'AdD pour l'événement 'Perturbation niveau cuve' 109
Figure 4.7 : Développement de l'AdD pour l'événement 'arrêt système remplissage' 110
Figure 4.8 : Développement de l'AdD pour l'événement 'Perturbation niveau réservoir' 111
Figure 4.9 : Développement de l'AdD pour l'événement 'atteinte seuil max réservoir' 112
Figure 4.10 : Modèle de l'interaction homme-machine déduit de l'AdD 116
Figure 4.11 : le sous réseau SRP12 117

Figure 5.1 : Architecture Informatique du système Ergo-Conceptor 120
Figure 5.2 : Démarche suivie par le système Ergo-Conceptor+ 122
Figure 5.3 : Architecture informatique du système Ergo-Conceptor+ 123
Figure 5.4 : Module de description du procédé 125
Figure 5.5 : Module de définition du modèle de l'interaction homme-machine 125
Figure 5.6 : Les matrices Pre et Post édités à l'aide de l'outil Ergo-Conceptor+ 126
Figure 5.7 : Module de génération graphique de l'IHM 127
Figure 5.8 : Une image écran du système Ergo-conceptor+ pour l'initialisation de sa base de connaissance 128
Figure 5.9 : Module de gestion de la dynamique du dialogue homme-machine 129
Figure 5.10 : Une vue graphique générée à l'aide de l'outil Ergo-Conceptor+ reflétant un état de fonctionnement normal du système 130

Figure 5.11 : Une vue graphique générée à l'aide du système Ergo-Conceptor+ reflétant un état de dysfonctionnement du système (relatif à la variable ID-PROD-E) 131
Figure 5.12 : Diagramme des classes 132
Figure 5.13 : Diagramme de séquence 133
Figure 5.14 : Système multi-agent de génération temps réel d'IHM 136

Liste des Tableaux

Tableau 1.1 : Evaluation de quelques approches proposées pour la conception de l'interface 37

Tableau 2.1 : Evaluation informelle des méthodes d'analyse et des outils de modélisation des systèmes 60

Tableau 4.1 : Analyse du système par l'AMDE 100

Tableau 5.1 : Tableau comparatif des deux systèmes Ergo-Conceptor et Ergo-Conceptor+ 124

Introduction générale

Durant les dernières décennies on a constaté un développement technologique considérable marqué spécialement par l'avènement de l'automatisation. Cet évènement a évolué timidement au fil des années jusqu'au début des années 70 avec l'apparition des microprocesseurs, la révolution de la microélectronique et par suite du monde de l'informatique. On croyait alors pouvoir se passer de l'être humain et pouvoir projeter «l'usine sans homme». Cependant, on a vite constaté l'erreur de cette pensée, principalement avec l'accroissement significatif des applications industrielles à haut degré de sécurité, dans les années 90, telles que les processus nucléaires, les industries chimiques ou encore les systèmes de transport. La présence de l'opérateur humain reste, désormais, nécessaire dans de tels systèmes. Son rôle s'est transformé d'abord en coordonnateur des sous-systèmes, puis en superviseur et contrôleur du système automatisé et il reste indispensable pour gérer l'imprévisible.

En effet, l'automatisation peut prévoir un certain nombre de défauts et les traiter mais elle ne peut jamais prévoir tous les défauts possibles et dès qu'une panne, donc une situation imprévisible survient, l'automatisation ne marche plus et c'est l'homme qui devrait intervenir et chercher à résoudre le problème le plus efficacement possible.

Pour cela il devrait être assisté pour bien résoudre les problèmes. L'objectif est donc de l'aider à réagir derrière une bonne interface bien conçue pour ce fait.

Dans ce but, les systèmes automatisés de commande ont été dotés de systèmes de surveillance, centralisant les informations issues des capteurs, dans une salle de contrôle ou sur le poste de conduite [Millot, 90][Kolski, 97].

Lorsque le système est en régime permanent et en fonctionnement normal, les calculateurs assurent la conduite sans nécessité d'intervention humaine. En revanche, lors des changements de régime ou en situation de dysfonctionnement, les opérateurs restent indispensables pour assurer les tâches de décision que l'on ne sait pas, ou pas complètement, automatiser.

De là vient la problématique de mener une conception orientée assistance de l'opérateur dès le début. Un point important évoqué au niveau de la conception des interfaces concerne le côté ergonomique de la présentation. En effet, les interfaces graphiques présentent le moyen privilégié pour : (i) informer l'opérateur de l'évolution du procédé et (ii) l'assister durant ses tâches mentales de résolution de problèmes [Rasmussen, 86][Johannsen, 95][De Keyser et al., 92].

Mais si ces informations sont mal présentées à l'écran et non bien organisées, ça risque de faire perdre l'opérateur humain et l'induire en erreur. Or dans de tels domaines, les erreurs humaines peuvent avoir des conséquences dangereuses sur le système de production, l'environnement, la sécurité voire même les vies humaines [Reason, 90].

Des outils de plus en plus puissants pour la création graphique des IHM sont disponibles sur le marché. La problématique réside donc au niveau méthodologique plutôt que pratique. Pour résumer, ces problèmes concernent principalement QUOI présenter à l'opérateur en fonction du contexte de fonctionnement et des tâches opérateurs, QUAND présenter ces informations et enfin COMMENT présenter celles-ci sur les écrans graphiques.

C'est dans ce cadre que s'insèrent nos travaux de recherche. Notre contribution vise, en effet, l'élaboration d'une approche globale de spécification, de vérification et de génération automatique des interfaces homme-machine (IHM).

Plusieurs travaux de recherche ont été élaborés au fil des années étudiant cette problématique et proposant des solutions différentes. Cependant, on a constaté que chacun se focalise sur une partie de la problématique de la conception des IHM. Un groupe de ces chercheurs s'est concentré sur l'étude des aspects ergonomiques. Un autre s'est focalisé sur les aspects d'analyse de la tâche humaine. D'autres chercheurs ont étudié l'intégration des outils formels dans le processus de spécification.

De même, la plupart des travaux effectués ont été étudiés sur des exemples de navigation aérienne, de contrôle de trafic routier où la modélisation du système est relativement simple par rapport aux systèmes industriels complexes. L'objectif de nos travaux consiste, donc, à proposer une démarche couvrant toutes les étapes de la conception des IHM, et intégrant des outils et techniques formelles de modélisation.

Ce mémoire présente les résultats de notre recherche. Il est composé de cinq chapitres.

Le premier commence par introduire les notions de base et les aspects généraux de l'interaction homme-machine. Il présente par la suite une synthèse d'approches et de méthodologies de conception d'IHM proposées dans la littérature. Cette synthèse aboutit à l'élaboration d'un cahier des charges d'une «approche globale» de conception des IHM qui nous sert comme guide méthodologique dans nos travaux de recherche.

Le deuxième chapitre propose une étude bibliographique d'un ensemble de méthodes, outils et modèles existants dans la littérature, supportant différentes approches méthodologiques de conception des IHM. Ils sont en fin de chapitre globalement discutés, et un choix est effectué à la fin, en terme d'articulation de ceux-ci, pour supporter l'approche proposée dans le chapitre suivant.

La présentation détaillée de cette approche fait l'objet du troisième chapitre et le quatrième s'intéresse à son illustration à travers un exemple d'application industrielle.

Finalement, le cinquième chapitre propose une maquette d'un outil informatique appelé "Ergo-Conceptor+", proposé pour supporter notre approche. Cet outil est appliqué à l'exemple proposé au quatrième chapitre. Les résultats de cette application sont évalués et discutés à la fin de ce chapitre.

Chapitre 1

Aspects méthodologiques liés à l'analyse, la conception et la génération d'IHM dans les systèmes automatisés de production

Plan du chapitre

I. Du rôle de l'homme dans la supervision et la conduite des systèmes automatisés de production
II. Aspects généraux de l'interaction homme-machine en salle de contrôle
- II.1- La modélisation de l'opérateur humain
- II.2- L'analyse de la tâche humaine
- II.3- Connaissances provenant de l'ergonomie cognitive pour la conception et l'évaluation des IHM

III. Des cycles de vie classiques aux cycles de vie enrichis sous l'angle de l'IHM
- III.1- Modèles généraux de génie logiciel
- III.2- Modèles enrichis pour les IHM
- III.3- Démarche globale en U dédiée aux systèmes industriels complexes

IV. Approches et méthodologies de conception des IHM proposées dans la littérature
- IV.1- Travaux de Palanque et Bastide : le modèle ICO
- IV.2- Travaux de De Rosis : le formalisme XDM
- IV.3- Travaux de Tabary et al. : la méthode TOOD
- IV.4- Travaux de Ait Ameur : la méthode B
- IV.5- Travaux de Rodrigez : le projet ALACIE
- IV.6- Travaux de Bodart : le projet TRIDENT
- IV.7- Travaux de Moussa : le système Ergo-Conceptor

V. Synthèse et identification des nouveaux besoins
- V.1- Tableau récapitulatif
- V.2- Guide méthodologique

Conclusion

De nombreux chercheurs ayant travaillé sur les IHM se sont orientés vers une conception centrée sur l'être humain. Ils se sont intéressés à l'étude des différents aspects de l'interaction homme- machine impliquant l'ergonomie et la psychologie cognitive pour la modélisation de l'opérateur humain, la mise en œuvre de méthodes d'analyse des tâches et activités humaines, la définition de différentes méthodes d'analyse et de conception ergonomique d'IHM. Des cycles de vie de génie logiciel enrichis pour les IHM ont été également proposés à cette fin.

Dans la première partie de ce chapitre, nous reprenons certains des aspects et concepts qui nous ont paru les plus importants. Nous enchaînons, par la suite, par le survol d'un ensemble d'approches et de méthodologies proposées dans la littérature pour la conception et la génération des IHM. Ces approches sont discutées et synthétisées à la fin du chapitre, identifiant ainsi de nouveaux besoins et proposant un cahier des charges d'une approche articulant analyse, conception et génération des IHM.

I. Du rôle de l'homme dans la supervision et la conduite des systèmes automatisés de production

Avec le développement accru de l'automatisation des systèmes de production, le rôle de l'opérateur humain est en train d'évoluer vers la manipulation « immatérielle », dans le sens qu'il intervient de moins en moins par une action physique directe pour la commande des machines et de plus en plus par une action de supervision et de conduite à travers une interface informationnelle. Les tâches confiées à l'opérateur deviennent, de là, plutôt « intellectuelles », exigeant la maîtrise du fonctionnement du procédé physique et de son automatisation ainsi que les moyens appropriés, au niveau de l'interface informationnelle, pour pouvoir suivre l'évolution du fonctionnement du système et agir chaque fois qu'une intervention est nécessaire [Cambacau, 98][Cassar et al., 94][Kolski, 01][Vicente et al., 01].

Le rôle de l'opérateur est encore à la fois plus important et plus difficile quand il s'agit d'intervenir, non pas simplement pour corriger des consignes de commande, mais pour résoudre des situations de dysfonctionnement imprévisibles. L'opérateur humain se trouve ainsi replacé au centre du système. Son interface de supervision et de conduite peut aller de la simple console avec écran à des grandes salles de contrôle (figure 1.1) et l'on substitue, dès lors, volontiers à l'appellation systèmes automatisés de production l'appellation systèmes homme-machine (SHM) [Sheridan, 85].

Figure 1.1 : Exemple de salle de contrôle

L'opérateur travaille donc à travers l'image mentale qu'il se fait, à chaque instant, du système et ce, grâce à l'interface graphique qui lui est présentée dans la salle de contrôle. L'affichage graphique peut être constitué d'un ensemble de vues représentant différentes informations statiques (indépendantes de l'état d'avancement de l'activité du procédé) ou dynamiques (évolutives en fonction de l'état d'avancement de l'activité du procédé). Ces vues permettent de synthétiser l'état du système et d'aider l'opérateur humain dans ses tâches de supervision et de commande (figure 1.2) [Marcus, 97].

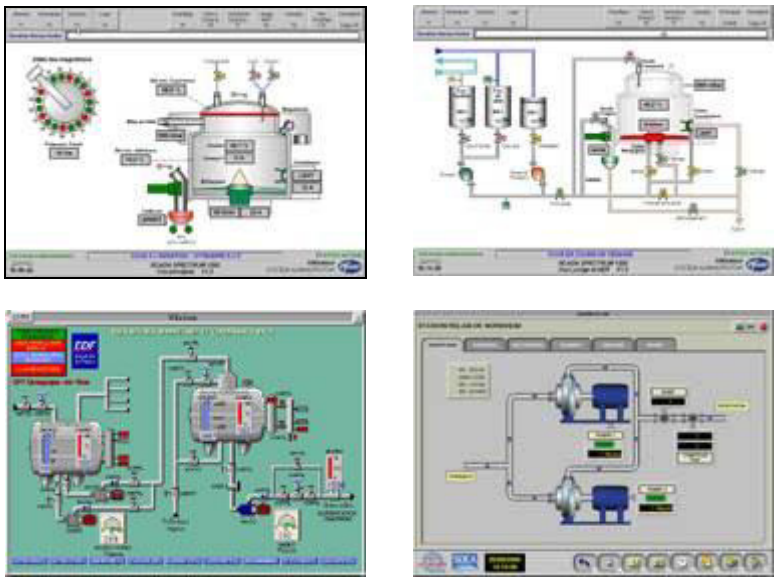


Figure 1.2 : Exemples de vues de supervision

L'architecture évoluée d'un système homme-machine peut être celle illustrée par la figure 1.3. Les trois couches "commande locale", "coordination" et "pilotage automatisé" constituent la commande temps réel. Souvent, on y adjoint un système de surveillance automatisé. L'interface homme-machine (IHM) gère les interactions de l'opérateur aussi bien avec le système de pilotage (informations sur l'avancement de l'activité du système) qu'avec le système de surveillance automatisé (informations sur l'état de fonctionnement des composants du système). Elle peut être soutenue par un système d'assistance de l'opérateur (spécifications de références, procédures de diagnostic, etc.).

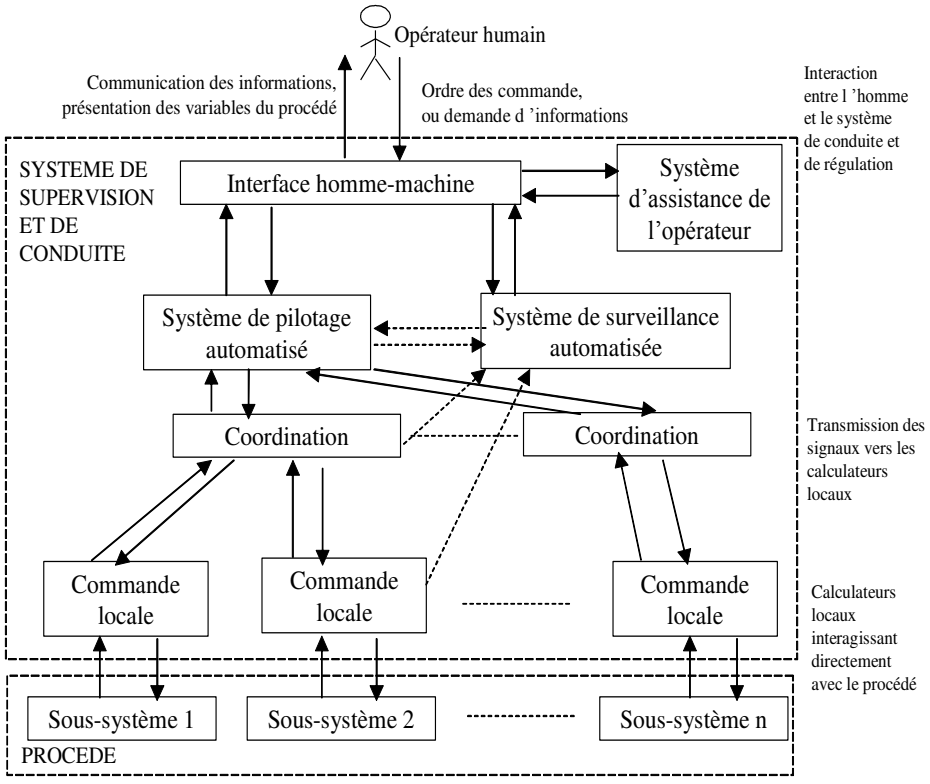


Figure 1.3 : Architecture d'un système homme-machine

Dans une telle architecture, les informations élaborées à l'intention de l'opérateur peuvent être diversifiées en contenu, en forme d'affichage et en volume et la problématique de l'interaction est donc de trouver la réponse aux trois questions classiques : Quand afficher ? Quoi afficher et Comment ?

Ainsi, la conception de l'interface homme-machine doit passer par la résolution des problèmes suivants :

- définir clairement les tâches affectées à l'opérateur humain,
- identifier les besoins informationnels de l'opérateur pour chacune de ces tâches,
- étudier le processus de leur communication à travers l'interface, tant sur le plan ergonomique que sur le plan des interactions que l'opérateur humain doit avoir au niveau de l'interface.

II. Aspects généraux de l'interaction homme-machine en salle de contrôle

Trois aspects ont marqué les débuts de recherche dans le domaine de l'interaction homme-machine :
- la modélisation de l'opérateur humain,
- l'analyse de la tâche humaine,
- l'ergonomie cognitive.

II.1. La modélisation de l'opérateur humain

Dans le domaine de l'interaction homme-machine, deux approches de modélisation de l'opérateur humain se distinguent :

- *L'approche fiabiliste :* elle a été initiée par Swain en 1963 [Swain, 63]. Son objectif est d'appliquer à l'opérateur les mêmes techniques d'évaluation de la fiabilité des matériels pour aboutir par la suite à des évaluations de la fiabilité globale du système homme-machine. Comme exemple de cette approche, nous pouvons citer THERP (Technique for Human Error Rate Prediction) [Swain, 64] qui quantifie la fiabilité humaine en passant par la probabilité d'occurrence des erreurs des opérateurs au même titre que les dysfonctionnements des systèmes [Swain et al.,81]. Plus tard, SHERPA, une variante de THERP, est venue ajouter à l'évaluation quantitative une évaluation qualitative dans l'espoir de rendre cette approche plus humaine [Embrey, 86].

- *L'approche cognitiviste* : elle vise à comprendre la genèse des erreurs de l'opérateur à partir des concepts tirés de la psychologie cognitive et s'appuie, pour cela, sur une description des mécanismes décisionnels humains mis en jeu pour accomplir les tâches. Elle facilite ainsi la déduction des besoins informationnels et des besoins d'assistance, utiles pour la spécification de l'interface. Contrairement, donc, à l'approche fiabiliste, l'approche cognitiviste met davantage l'accent sur la construction plutôt que sur l'évaluation.

C'est cette dernière approche qui nous a servi de base au démarrage de nos travaux. En effet, dans le contexte qui nous intéresse, la psychologie cognitive a pour but de bien connaître l'opérateur humain pour pouvoir mieux l'assister. Cette science consiste à étudier les caractéristiques typiquement humaines et les processus par lesquels l'homme acquiert, traite et interprète les informations de son environnement.

Parmi les résultats des travaux menés dans ce domaine, nous distinguons principalement :
- le modèle du processeur humain de Card et al. [Card et al., 83];
- la théorie décisionnelle de Rasmussen proposant un cadre de modélisation du comportement de l'opérateur humain dans le cas de la surveillance des procédés industriels [Rasmussen, 86];
- la théorie de l'action de Norman offrant un modèle de réalisation de tâche humaine [Norman, 86].

Nous proposons, ci-dessous, un bref rappel des concepts clés de ces modèles.

II.1.1- Le modèle du processeur humain

Le modèle du processeur humain, proposé par S. Card et al. en 1983, peut être considéré comme le premier modèle de la psychologie cognitive étudié dans le cadre des systèmes interactifs. Il consiste à représenter l'individu comme un système de traitement d'information régi par des règles. L'objectif est de pouvoir construire, à partir d'une théorie technique, des modèles qui pourraient répondre aux questions que le concepteur se poserait sur des phénomènes précis de l'interaction Homme-Machine.

Le modèle du processeur humain consiste en la combinaison de trois sous-systèmes interdépendants (le système sensoriel, le système moteur et le système cognitif) qui possèdent, chacun, une mémoire et un processeur (figure 1.4). Le système sensoriel est spécialisé dans les traitements des stimuli. Le système moteur est responsable des mouvements et manipulations des unités physiques de commande. Le système cognitif, lui, consiste à contrôler le comportement de l'individu et à activer en conséquence mnèmes mis à sa disposition dans la mémoire à court terme. Un mnème n'est autre qu'une unité cognitive symbolique. C'est une abstraction d'information en provenance de la mémoire à long terme.

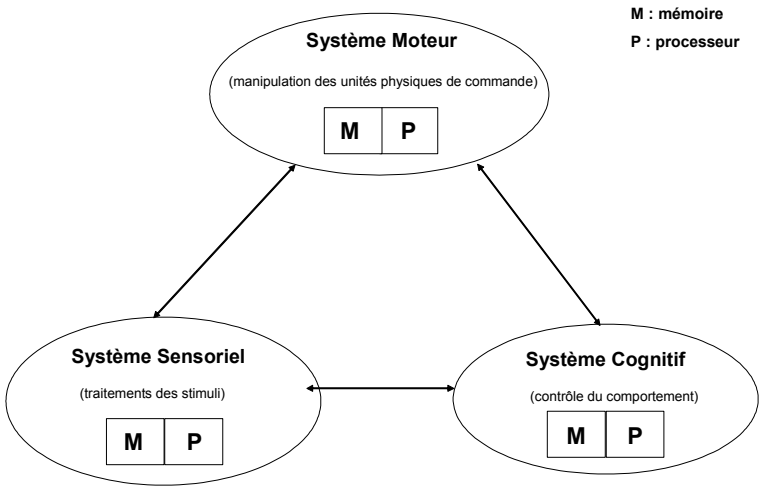


Figure 1.4 : Modèle du processeur humain

L'utilité dans ce modèle est d'expliquer et de prédire les performances de l'utilisateur. En effet, les contraintes de performance sont relatives aux mémoires et processeurs de chacun des trois systèmes : moteur, sensoriel et cognitif. Elles peuvent être mesurées à partir des paramètres tels que : le cycle de base du processeur, la capacité de la mémoire, sa persistance ou encore le type d'information mémorisé.

Ce modèle constitue un cadre fédérateur à la diversité des connaissances en psychologie utilisant une terminologie de l'informaticien.

II.1.2- La théorie décisionnelle de Rasmussen

La théorie décisionnelle de Rasmussen, appelée aussi "échelle de décision", vise à expliquer la démarche de résolution de problème suivie par un utilisateur d'imagerie en salle de supervision. Ce modèle a constitué une ouverture très importante dans la modélisation de l'opérateur humain.

En effet, le modèle mental qualitatif de Rasmussen (figure 1.5) illustre bien la démarche générale de résolution de problème suivie par l'opérateur. Il comprend quatre étages séquentiels de traitement d'information :

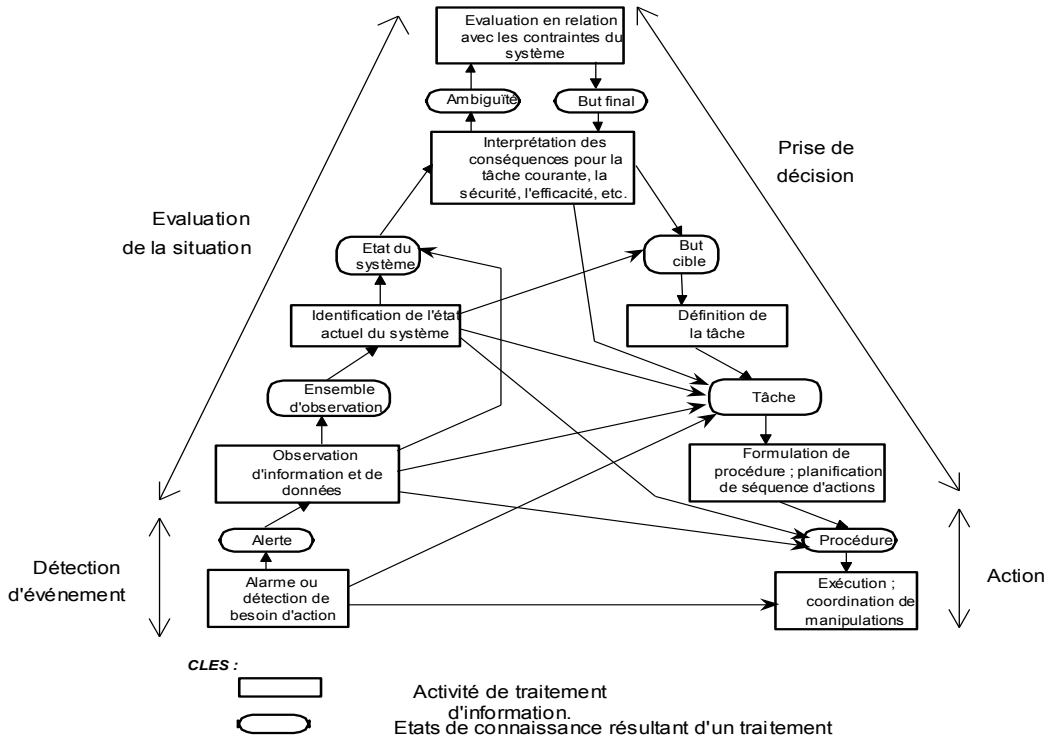


Figure 1.5 : L'échelle de décision de Rasmussen

- détection d'un événement anormal mettant l'opérateur en état d'alerte (détection par une alarme ou par une observation de l'évolution anormale d'une ou de plusieurs variables),
- évaluation de la situation en observant l'ensemble des informations utiles de façon à identifier l'état du procédé,

- définition d'une stratégie générale de correction, compte tenu de l'état identifié et des objectifs de conduite qui lui sont assignés, puis décomposition en tâches et procédures d'actions,
- exécution des actions.

Bien entendu, lorsqu'il se trouve face à une situation connue, l'opérateur peut sauter une ou plusieurs étapes selon son expérience, ses connaissances et selon la gravité de la situation qui se présente.

Cette approche de modélisation constitue une source précieuse d'enseignement et de réflexion pour les concepteurs des interfaces homme-machine et des outils d'assistance. En effet, elle permet de mieux positionner et comprendre les besoins de l'utilisateur pour les différentes tâches qu'il a à accomplir et cela dans les différentes situations normales et anormales du système. Notons aussi qu'elle a fait l'objet d'extensions par Hoc et Amalberti sous l'angle de la dynamique des situations [Hoc et al., 95].

II.1.3- La théorie de l'action de Norman

L'hypothèse de la théorie de l'action de Norman est que l'individu élabore des modèles conceptuels. Ces modèles sont les données déterminantes de son comportement. En fait, le dit "modèle conceptuel" n'est autre qu'une représentation mentale qui dépend de la connaissance déjà acquise et de la compréhension de la situation présente. Il est donc incomplet, imprécis et il évolue avec l'expérience.

Selon Norman, la réalisation d'une tâche met en jeu au moins sept activités (figure 1.6) :

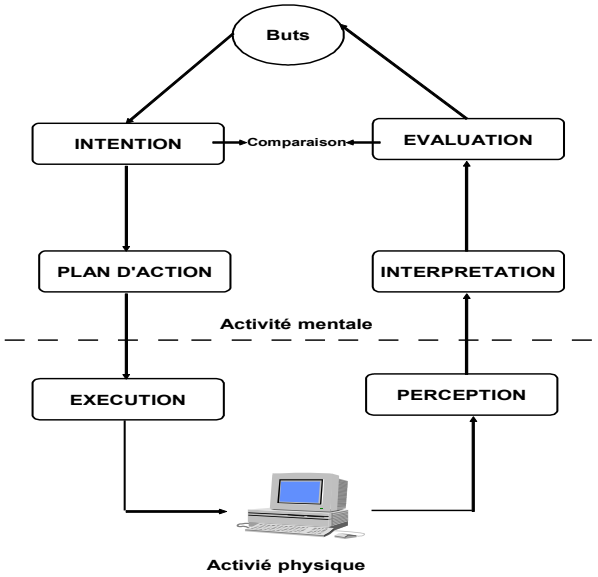


Figure 1.6 : La théorie de l'action de Norman

1- l'établissement d'un but qui n'est autre qu'une représentation mentale de l'état du système que l'utilisateur souhaite atteindre ;
2- la formation d'une intention, c'est-à-dire la décision d'agir de façon à atteindre le but ;

3- la spécification d'une suite d'actions nécessaires pour atteindre le but ;
4- l'exécution du plan d'action en agissant sur les dispositifs d'entrée ;
5- la perception du nouvel état du système ;
6- l'interprétation des modifications effectuées aboutissant à une nouvelle représentation mentale de l'état du système ;
7- l'évaluation de l'état du système et sa comparaison avec le but préétabli.

Le dénominateur commun de ces modèles issus de la psychologie cognitive est l'analyse des tâches opérateurs. Cependant, leur inconvénient majeur est le manque de support formel. En effet, ces modèles mettent l'accent sur une description à haut niveau d'abstraction à l'aide de la notion de tâche. Ils ne peuvent donc pas être appliqués d'une manière directe et systématique pour la conception des interfaces homme-machine. De nombreux chercheurs travaillent depuis des années sur la formalisation de ce concept clé de la psychologie cognitive : "la tâche opérateur". Une synthèse de plusieurs travaux représentatifs est maintenant présentée.

II.2- L'analyse de la tâche humaine

Dans les salles de contrôle, les opérateurs sont généralement isolés des installations industrielles et ont généralement à résoudre des problèmes impliquant des centaines voire des milliers de variables du système (telles que températures, pressions, débits,…) [Hammouche, 93][Jeffries, 97][Marti, 98]. Les tâches humaines nécessitent donc, à cet effet, un haut niveau de connaissance et de savoir faire. Elles sont regroupées selon Rouse [Rouse, 83] en quatre principales classes :

- *Les tâches de supervision et d'optimisation :* elles correspondent au rôle principal de contrôle assigné à l'opérateur humain. L'IHM doit simplifier la tâche de la supervision du procédé afin de permettre : (1) l'optimisation de la production moyennant de fins ajustements et (2) la détection et l'anticipation de l'apparition des anomalies possibles.

- *les tâches de transition :* elles correspondent aux procédures prédéfinies qui assurent les transitions d'état du fonctionnement du procédé industriel (marche/arrêt, changement de point de fonctionnement, etc.). L'IHM devrait simplifier : (1) l'accomplissement de ces procédures prédéfinies, (2) le jugement concernant leur répercussion sur la totalité du procédé et (3) la supervision de l'évolution du procédé.

- *Les tâches de détection de défaillances et de diagnostic :* à l'apparition d'une alarme et/ou observation d'une évolution anormale de certaines variables du procédé, l'opérateur humain doit être capable de détecter les défauts et d'analyser le problème. On considère que cette tâche peut être facilitée si l'interface présente à l'opérateur les interrelations causes/effets entre les variables.

- *Les tâches de correction et de compensation :* elles concernent le rétablissement du fonctionnement normal du procédé. L'IHM devrait pouvoir aider les opérateurs humains à décider quelles actions ils devraient accomplir et à estimer quels effets auraient ces actions sur le procédé.

L'analyse de ces tâches humaines nécessite une collaboration étroite et pluridisciplinaire entre les différents intervenants du processus de développement des applications interactives [Norman, 86][Meinadier, 91].

Selon Stammers [Stammers, 90], le processus d'analyse de la tâche peut être décomposé en trois étapes essentielles (figure 1.7) :

1- la première étape consiste à recenser les données de la tâche et à regrouper la documentation nécessaire,
2- la seconde étape consiste à formaliser une description de la tâche à partir des données collectées et/ou de tâches sous-jacentes aux précédentes,
3- la dernière étape applique les résultats de la modélisation pour la conception ou l'évaluation du système.

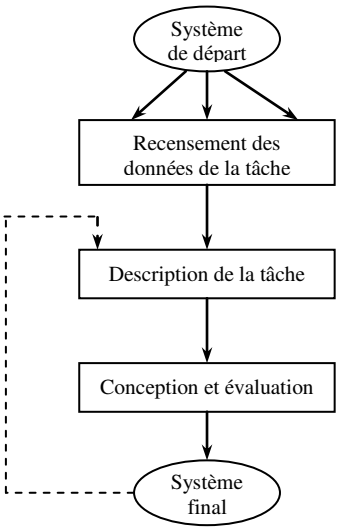


Figure 1.7 : Approche d'analyse de la tâche, selon Stammers

Toutes les données recueillies lors de l'analyse de la tâche doivent donc faire l'objet d'une modélisation en vue d'en tirer les recommandations nécessaires pour la conception des applications interactives [Robert et al., 97] [Sheridan et al., 97] [Terwilliger et al., 97][Shepherd, 98]. Nous résumons ici, à titre d'exemple, trois de ces modèles de tâches, issus des travaux de Scapin à l'INRIA, de ceux de Abed et al. au LAMIH et de Paterno et al. au CNUCE. De nombreux autres exemples peuvent être trouvés dans l'ouvrage de Diaper et Stanton [Diaper et Stanton, 01].

II.2.1- Travaux de Scapin

Le modèle MAD (Méthode Analytique de Description de Tâches) proposé par Scapin en 1989 est un formalisme de description de tâches. Des objets tâches sont conçus comportant chacun [Scapin, 89][Scapin et al., 01] :

- *un état initial (I) :* sous-ensemble de l'état du monde de l'application, constitué de la liste des objets et arguments d'entrée de la tâche ;
- *un état final (F) :* sous-ensemble de l'état du monde de l'application, constitué de la liste des objets et arguments de sortie de la tâche ;

- *un but (B) :* sous-ensemble de l'état final indiquant explicitement le but recherché dans l'exécution de la tâche ;
- *des pré-conditions (C.N) :* ensemble de prédicats exprimant les contraintes qui doivent être satisfaites par l'état initial pour le déclenchement de l'exécution de la tâche ;
- *des post-conditions (P.C) :* ensemble de prédicats exprimant les contraintes qui doivent être satisfaites par l'état final, après l'exécution de la tâche.

MAD distingue deux types d'objets tâches : les tâches élémentaires indécomposables correspondant aux actions et les tâches composées.

Plusieurs opérateurs ont été définis au niveau de ce modèle tels que : SEQ pour décrire le séquencement des tâches, PAR pour exprimer le parallélisme, ALT pour spécifier l'exécution alternative, BOUCLE pour préciser l'itération etc. (figure 1.8). On remarque par contre l'absence de constructeurs conditionnels du style IF.

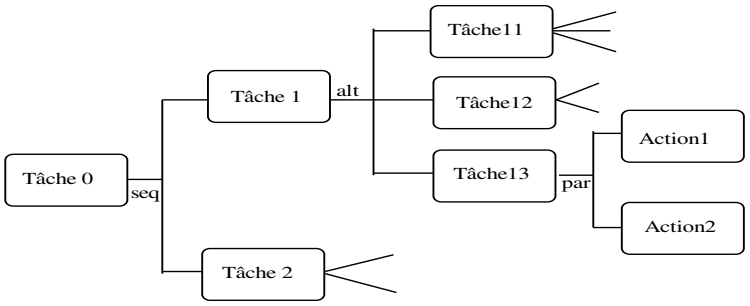


Figure 1.8 : Une arborescence de tâches MAD

Ce modèle fournit une bonne représentation des tâches, compréhensible facilement aussi bien du point de vue de l'ergonome que de celui de l'informaticien. Cependant, il décrit le travail indépendamment de la répartition des tâches entre l'homme et la machine, ce qui peut rendre difficile son utilisation pour la conception de certaines applications interactives et, plus particulièrement, pour la conception d'applications industrielles où les tâches peuvent être manuelles, partageables ou totalement automatisées.

II.2.2- Travaux de Abed et al.

Abed et al. proposent une démarche de description de la tâche basée sur une combinaison de la méthode SADT (Structured Analysis and Design Technique) et des réseaux de Petri. Cette démarche est décrite en quatre étapes [Abed, 01][Abed et al., 01]:

1- Utiliser la méthode SADT pour décomposer hiérarchiquement le comportement du système homme-machine (SHM) en tâches.
2- Identifier les tâches élémentaires au dernier niveau de la décomposition et les répartir entre machine et opérateur.
3- Analyser les tâches humaines et distinguer entre les tâches interactives qui définissent des routines d'action et les tâches non interactives de simple surveillance.

4- Décrire le fonctionnement et la dynamique des tâches élémentaires interactives en utilisant les réseaux de Petri Synchronisés (RdPS).

Pour chaque tâche élémentaire dont on a défini le scénario à l'aide des réseaux de Petri, ces auteurs expliquent comment il est possible d'identifier les besoins en information qui sont en fait les entrées et les contraintes imposées sous forme de données de contrôle des boîtes actigrammes de SADT (figure 1.9).

Ce principe de décomposition SADT/Petri permet de décrire d'une manière formelle et rigoureuse, aussi bien l'aspect statique que l'aspect dynamique du SHM, et conduit à identifier les besoins informationnels des opérateurs humains.

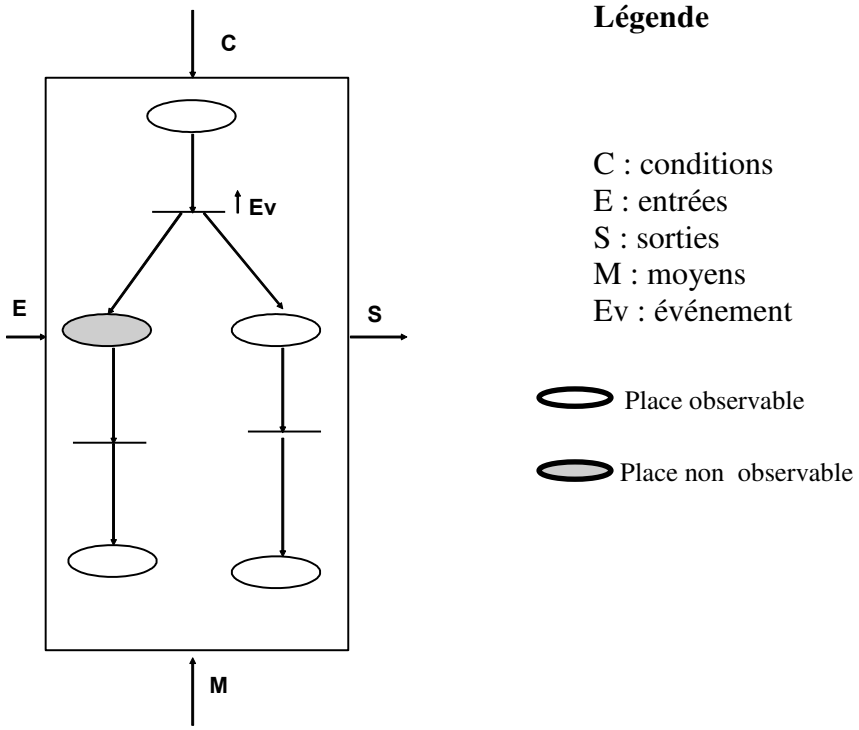


Figure 1.9 : Une boîte SADT relative à une tâche humaine interactive dont le fonctionnement est décrit par un RdPS

II.2.3- Travaux de Paterno et al.

Paterno et al. proposent une démarche et un outil de modélisation des tâches nommé CTT (Concur Task Trees) [Paterno, 01][Mori et al., 02]. CTT a pour source d'inspiration les premières études développées au sein de l'équipe autour de la notation Lotos pour la spécification des interfaces graphiques.

CTT se base sur la notation UAN (User Action Notation) développée au début des années 80, par Antonio Siochi, H. Rex Hartson et Deborah Hix. Son objectif est de fournir un formalisme de modélisation de l'interaction homme-machine compréhensible par l'équipe de conception et de développement des interfaces [Hix et Hartson, 93].

UAN se base sur une notation textuelle modélisant les différentes tâches de l'interaction. Les tâches élémentaires sont décrites par des tables précisant : l'ensemble des actions opérateur, les réponses (feedback) du système et les éventuelles modifications d'états comme suit :

Tâche: <nom de la tâche >		
Actions de l'utilisateur	**Réponse (FEEDBACK)**	**Etat de l'interface**

Les notations utilisées pour décrire ces trois éléments du tableau sont laissées au soin du concepteur. Cependant, un standard est proposé et utilisé par Hix et al.

UAN ne décrit que des tâches asynchrones. Elle propose une description de l'interface en termes de tâches non ordonnées. UAN a été enrichie par la suite par plusieurs opérateurs permettant d'exprimer les relations temporelles entre les tâches tels que :

- T1|||T2 : les actions de T1 et T2 peuvent être effectuées dans n'importe quel ordre.
- T1|[]|T2 : T1 et T2 doivent se synchroniser sur certaines actions pour échanger des informations.
- T1>>T2 : quand T1 est terminée T2 devient active.
- T1 []>>T2 : quand T1 se termine elle fournit des informations à T2 et l'active.
- T1[>T2 : quand une action de T2 se produit, la tâche T1 est désactivée.
- [T1] : T1 est facultative pour atteindre le but (la tâche de niveau supérieur).
- T1* : T1 peut être répétée autant de fois que possible pour atteindre le but.
- T1{n} : T1 doit être répétée n fois pour atteindre le but.

Sa notation textuelle et l'absence d'un outil de support de UAN, a fait que son utilisation reste limitée dans la pratique. Cependant, elle a servi de base à la notation graphique CTT pour la modélisation des tâches concurrentes.

En effet, l'objectif principal de CTT, est de fournir une notation facile de modélisation des tâches supportant la conception des applications interactives. Cette notation distingue quatre types de tâches :

- les tâches utilisateur : relatives à des activités cognitives de réflexion, sélection, choix, etc. ;
- les tâches système (ou tâches application) : relatives à des traitements informatiques et la génération de résultats ;
- les tâches interactives : relatives à des actions de l'opérateur et des "feedbacks" du système ;
- les tâches abstraites (ou tâches non élémentaires) : relatives à une composition de tâches de différents types.

CTT se focalise sur une modélisation des activités moyennant une structure hiérarchique. Elle se base sur une syntaxe graphique arborescente et sur un ensemble d'opérateurs pour exprimer les différentes relations temporelles (figure 1.10).

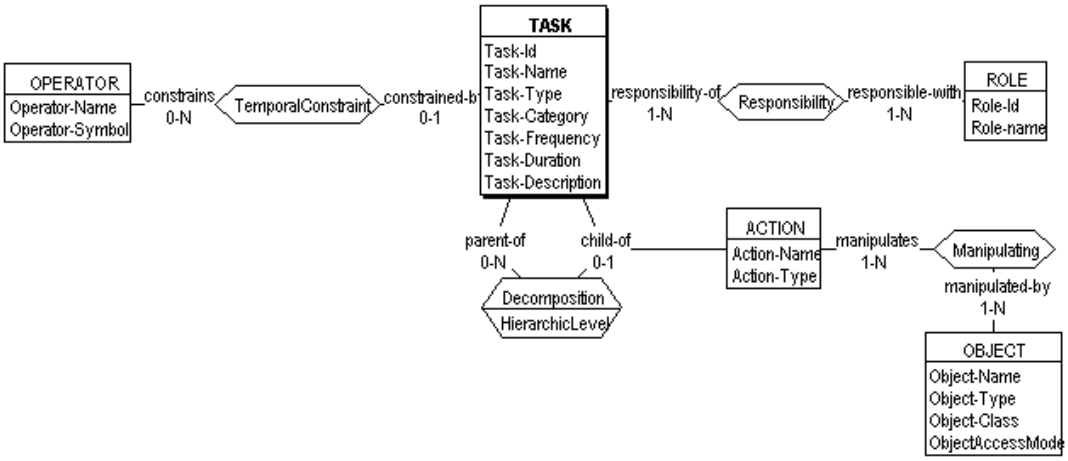


Figure 1.10 : MCD d'une modélisation CTT

II.3- Connaissances provenant de l'ergonomie cognitive pour la conception et l'évaluation des IHM

Nous avons présenté jusqu'ici deux concepts clés de la modélisation de l'interaction homme-machine, à savoir le concept de modèle de l'opérateur humain (ou utilisateur dans le contexte qui nous préoccupe, en l'occurrence celui des systèmes industriels complexes) et celui de l'analyse et de la modélisation de la tâche humaine. Un autre aspect primordial de l'interaction homme-machine prend la forme de connaissances provenant de l'ergonomie cognitive pour la conception et l'évaluation des IHM [Amalberti et al., 98] [Jacko and Sears, 03]. Cet aspect est maintenant présenté.

L'ergonomie est un mot qui vient du grec (*ergo* : travail ; *nomie* : règles, lois). C'est donc la science des règles de travail. Elle est définie comme étant le rassemblement des connaissances sur le fonctionnement de l'homme en activité afin de les appliquer à la conception des tâches, des machines, des outillages, des bâtiments et des systèmes de production. Pour ce qui est de la conception des SHM, il s'agit de concevoir les postes de travail, au sens large ; en fonction des caractéristiques physiologiques et psychologiques de l'opérateur humain et de son comportement en situation de travail, [Bastien, 96][Scapin and Bastien, 97][Bastien et Scapin, 01][1].

L'ergonomie cognitive pour la conception et l'évaluation des IHM, peut être considérée comme une approche légère de conception venant en complément de celle de la psychologie cognitive. C'est une approche principalement expérimentale qui tente de s'opérationnaliser en se basant sur la transformation de l'expérience acquise dans divers domaines en règles ergonomiques applicables [Vanderdonckt, 99]. Sous l'angle de l'utilisabilité généralement vue comme venant en complément du concept d'utilité pour l'ergonomie des logiciels [Smith, 86] [Nielsen, 93] [Shneiderman, 98] [Grammenos et al., 99], ces règles peuvent être groupées en huit classes de critères (pouvant se décomposer en sous-critères) comme suit [Scapin, 89][Bastien, 96] [Reed et al., 99] :

[1] Voir aussi [De Montmollin, 95]

1- *Le guidage* : c'est l'ensemble des moyens permettant de conseiller, orienter, informer et conduire l'utilisateur lors de ses interactions avec l'ordinateur (messages, alarmes, labels, etc.).
2- *La charge de travail* : c'est l'ensemble des éléments de l'interface qui ont pour rôle de réduire la charge perceptive ou mnésique des utilisateurs et d'augmenter l'efficacité du dialogue.
3- *Le contrôle explicite* : c'est la prise en compte par le système des actions explicites des utilisateurs tout en leur laissant le contrôle sur le traitement de leurs actions. Il intègre donc l'anticipation des actions utilisateur et l'offre des options appropriées.
4- *L'adaptabilité* : c'est la flexibilité du système et sa capacité à réagir selon le contexte, et selon les besoins, les préférences et le niveau d'expérience des utilisateurs.
5- *La gestion des erreurs* : c'est l'ensemble des moyens permettant d'une part d'éviter ou de réduire les erreurs, et d'autre part de les corriger lorsqu'elles surviennent en intégrant des messages d'erreur brefs avec des termes aussi spécifiques que possible.
6- *L'homogénéité et la cohérence* : c'est la façon avec laquelle les choix de conception de l'interface sont conservés pour des contextes identiques, et sont différents pour des contextes différents (présenter toujours la même information au même endroit, utiliser toujours le même format de champs d'entrée de données,…).
7- *La signifiance des codes et les dénominations* : c'est l'adéquation entre l'objet ou l'information affichée ou entrée et son référent pour faciliter le rappel et la reconnaissance.
8- *La compatibilité* : c'est l'accord pouvant exister entre les caractéristiques des utilisateurs (mémoire, perceptions, habitudes, compétences, attentes, âge, etc.) et des tâches, d'une part, et l'organisation des sorties, des entrées et du dialogue d'une application donnée, d'autre part. Ça concerne également le degré de similitude entre divers environnements ou applications.

Dans la pratique, l'exploitation des connaissances ergonomiques se fait selon trois approches principales :

1- *Le recours à un expert humain* : cette méthode donne souvent de bons résultats. L'expert est cependant, trop souvent mis à contribution à une phase de conception déjà avancée et son intervention consiste à aménager le poste de travail et évaluer et organiser les différentes vues de l'IHM selon son expertise. Son impact peut être relatif si les décisions les plus importantes ont été déjà prises. L'idéal serait, donc, de le faire participer dès le début du projet.

2- *L'utilisation de guides ergonomiques* : Il existe de plus en plus de guides dans la littérature [Vanderdonckt, 94][Bastien, 96]. Ces guides ont permis d'améliorer considérablement les conditions de travail grâce à la vulgarisation de l'ergonomie en touchant un public de plus en plus large. Toutefois, les recommandations ergonomiques de ces guides ne sont pas toujours directement applicables. Elles présentent parfois un niveau de recommandations trop générales ou plutôt spécifiques à des domaines particuliers. Aussi, la structuration de ces recommandations dans les guides n'est souvent pas adéquate vis-à-vis des attentes des concepteurs de l'IHM ce

qui rend leur utilisation assez difficile. Ces guides peuvent exister sous une forme papier ou électronique.

3- *Le recours à des systèmes informatisés* : ces systèmes tentent de formaliser les recommandations ergonomiques des experts humains et disponibles dans les guides, tout en cherchant à automatiser leur application dans le processus de conception des IHM. Cette approche consiste à faire appel à des techniques de l'intelligence artificielle pour mettre en place des systèmes à base de connaissances (SBC), prenant la forme de « tools for working with guidelines » (TFWWG) [Vanderdonckt, 00][Iwc, 99][Iwc,00]. Nous pouvons citer à ce sujet, comme exemple, le système SYNOP [Kolski et al., 91][Kolski et al., 96], un système expert développé pour l'aide à l'évaluation et l'amélioration de vues graphiques ; ou encore le système à base de connaissances ERGO-CONCEPTOR [Moussa, 92], conçu pour l'aide à la conception ergonomique des interfaces de supervision des procédés industriels (notre recherche se situant dans la lignée des travaux relatifs à SYNOP et ERGO-CONCEPTOR).

Ayant présenté des aspects primordiaux de l'interaction homme-machine à savoir : la modélisation de l'opérateur et de la tâche humaine et la prise en considération de connaissances ergonomiques, il s'avère nécessaire d'introduire à ce niveau, des exemples représentatifs de méthodologies de conception des IHM intégrant pour certaines ces aspects.

Dans ce qui suit, nous commençons par donner un bref aperçu des principaux modèles généraux du génie logiciel, puis sur plusieurs modèles enrichis sous l'angle de l'IHM. Nous présentons, par la suite, un cadre méthodologique de conception et d'évaluation des SHM, faisant office de source d'inspiration dans notre travail de recherche.

III. Des cycles de vie classiques aux cycles de vie enrichis sous l'angle de l'IHM

III.1- Modèles généraux de génie logiciel

L'approche traditionnelle de génie logiciel distingue généralement six phases dans la vie d'un logiciel (figure 1.11) :

1. *La phase de pré-analyse* (ou étude d'opportunité) : elle répond à la question **pourquoi** : pourquoi faut-il réaliser un certain logiciel, quels sont les besoins ? Pourquoi a-t-on besoin du logiciel ? Y a-t-il de meilleures alternatives ? Le logiciel sera-t-il satisfaisant pour les utilisateurs ? Y a-t-il un marché pour le logiciel ? Etc. [Birrell, 85]. Durant cette phase se pose également la question des ressources nécessaires pour réaliser le logiciel (budget, personnel, matériel).

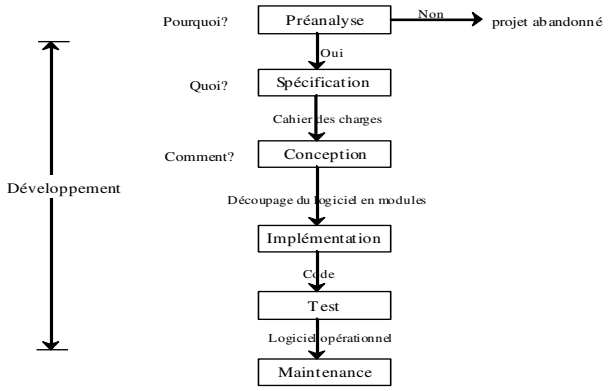


Figure 1.11 : Démarche traditionnelle de développement d'un logiciel

2. *La phase de spécification* : elle va permettre de définir précisément quelles sont les fonctions à réaliser par le logiciel. La phase de spécification (spécification des besoins) répond à la question **quoi** ?. Le résultat de cette phase est le cahier des charges du logiciel.

3. *La phase de conception* : elle répond à la question **comment** ? : comment réaliser le logiciel défini par le cahier des charges. Le résultat de cette phase est l'architecture du logiciel, autrement dit la décomposition en modules, avec spécification des interfaces de chacun des modules et détail des algorithmes de chacune des procédures des modules.

4. *La phase d'implémentation* : elle s'occupe du codage de chacun des modules définis lors de la phase de conception.

5. *La phase de test* : elle commence généralement par le test séparé de chacun des modules et finit par le test complet de tout le logiciel. Le résultat de cette phase est en principe un logiciel opérationnel.

6. *La phase de maintenance* : on distingue dans cette phase trois types de maintenance : la *maintenance corrective*, qui s'occupe de corriger les bugs découverts une fois le logiciel remis au client ; la *maintenance adaptative* qui adapte le logiciel à un environnement (logiciel et matériel) qui évolue comme par exemple l'adaptation à de nouvelles versions d'un système d'exploitation ; et la *maintenance perfective*, qui s'occupe d'étendre le logiciel pour y inclure de nouvelles possibilités, de nouvelles fonctions, etc.

En fait, cette vue est bien trop simpliste, car dans la pratique on a toujours tendance à revenir sur certains points identifiés et certains choix faits au niveau des phases précédentes du cycle de vie du logiciel. Une façon plus réaliste consiste à montrer l'enchaînement des différentes phases du cycle avec le *feed-back* possible d'une phase sur une autre. Dans la littérature, nous trouvons principalement quatre modèles qui illustrent cet aspect. Ces modèles sont maintenant présentés.

III.1.1- Modèle en cascade

Le modèle en cascade [Boehm, 88] définit une réalisation séquentielle des étapes du processus de développement du logiciel avec des retours possibles vers les étapes précédentes afin de rectifier les lacunes éventuellement identifiées (figure 1.12). Ce modèle propose donc des étapes de validation ou de vérification des résultats d'une certaine phase du cycle avant d'aborder la phase suivante.

Le modèle en cascade est séduisant de par sa simplicité. Cependant, les retours arrières qu'il propose se limitent à des retours à une phase immédiatement antérieure. Le contrôle ainsi imposé à la fin de chaque étape ne favorise donc pas les changements et tend à stabiliser rapidement le logiciel.

Concernant le développement des systèmes interactifs, on remarque qu'aucune analyse et modélisation des tâches utilisateur n'est préconisée. Cette notion n'est considérée, lors de la première étape, que d'une manière informelle selon le bons sens et l'expérience du concepteur. L'aspect utilisateur n'intervient trop souvent qu'implicitement dans les étapes finales d'évaluation du produit réalisé [Tabary, 01].

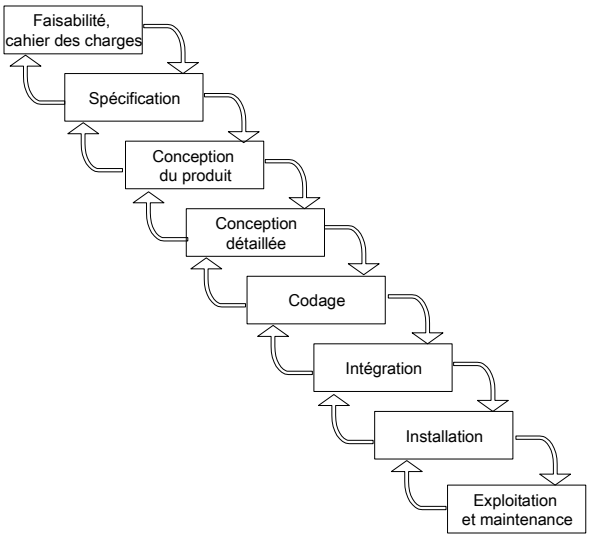


Figure 1.12 : Modèle en cascade (waterfall)

III.1.2- Modèle en *V*

Le modèle en V [McDermid et al., 84] est plus proche de la réalité de l'articulation entre les activités de spécification et de conception, et celles de validation et de vérification (figure 1.13). Contrairement au modèle en cascade, ce modèle fait apparaître le fait que le début du processus de développement conditionne ses dernières étapes.

En effet, le modèle en V structure les étapes du cycle de vie d'un logiciel en deux phase :

- une phase descendante pour la spécification, la conception et le codage, et
- une phase ascendante pour les validations, l'intégration et les tests.

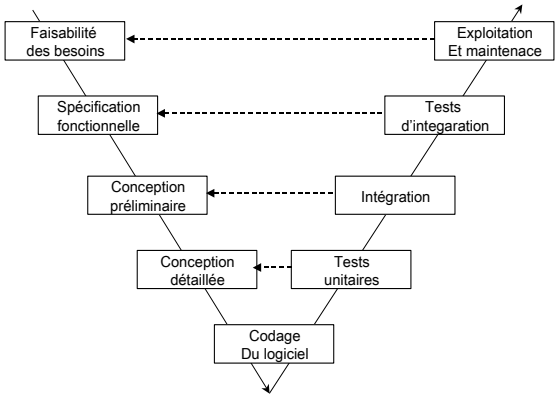


Figure 1.13 : Modèle en V

Ainsi, dans chaque étape de la phase descendante doivent être prévus le plan, les moyens et les méthodes permettant d'évaluer et de valider cette étape. Ce qui permet une évaluation le plus en amont possible du système. Ainsi les étapes de la branche droite du V peuvent être mieux préparées et planifiées.

En raison de l'importance accordée à l'évaluation dans ce modèle, l'utilisateur peut jouer un rôle plus déterminant que dans le modèle en cascade. Néanmoins, les aspects de l'analyse et la modélisation des tâches humaines et des utilisateurs restent toujours non positionnés dans ce modèle. Ce qui demande un raffinement spécial en cas de développement de logiciels fortement interactifs.

III.1.3- Modèle en spirale

Le modèle en spirale proposé par Boehm et al. en 1984 [Boehm, 88], (figure 1.14) met l'accent sur l'analyse des risques.

Le développement dans ce modèle est décrit comme un processus itératif de prototype selon quatre phases (une par cadran), illustré par le parcours depuis l'intérieur jusqu'à l'extérieur de la spirale :

i- déterminer les objectifs, les alternatives et les contraintes à partir des résultats du cycle précédent (pour le premier cycle, à partir d'une analyse préliminaire des besoins) ;
ii- analyser les risques, évaluer les alternatives, avec éventuellement prototypage, simulation et essais ;
iii- développer et vérifier la solution retenue (suivant les étapes du modèle en cascade ou en *V*) s'il ne reste plus de risques sinon poursuivre la spirale vers la phase suivante ;
iv- fixer les moyens à mettre en œuvre pour poursuivre la spirale par revue des résultats et planification du cycle suivant.

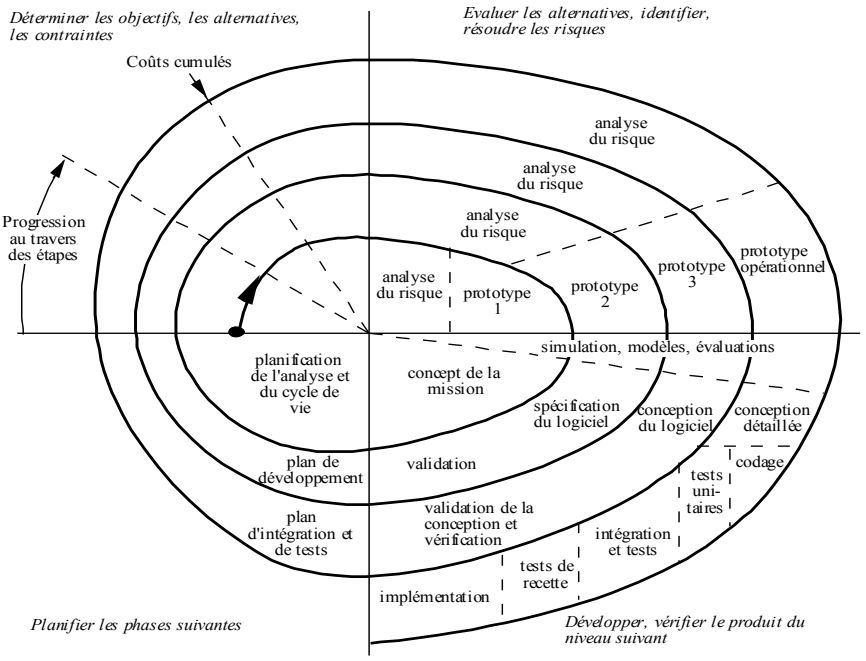


Figure 1.14 : Modèle de la spirale

Le modèle en spirale utilise systématiquement des prototypes exploratoires afin de guider la conception. Il est plus particulièrement adapté aux projets innovants, à risques et dont les enjeux sont importants.

Ce modèle possède un avantage certain vis-à-vis du développement de logiciels fortement interactifs. En effet, les besoins sont formulés progressivement et les différents risques rencontrés sont résolus au fur et à mesure. Ce n'est qu'une fois les risques stabilisés, qu'on procède à une étape plus fine de développement et d'élaboration d'éléments logiciels.

De nombreux auteurs tels James [James, 91], Nielsen [Nielsen, 93] ou Lichter [Lichter et al., 94] s'accordent à penser que le prototypage, à défaut d'une méthode de conception-évaluation universelle, systématique et rigoureuse, propose une solution très pratique pour éviter ou au moins limiter les défauts de conception et les erreurs ergonomiques, pour accroître la qualité des produits et abaisser les coûts de production, de mise au point et de maintenance.

III.1.4- Modèle par incréments

Dans les modèles en spirale, en V ou en cascade, il est implicite qu'après une décomposition en composants, ceux-ci sont développés indépendamment les uns des autres, en parallèle ou en séquence selon les ressources disponibles. Dans le modèle par incréments, seul un sous-ensemble est développé à la fois. Dans un premier temps un *logiciel noyau* est développé, puis successivement, les incréments sont développés et intégrés (figure 1.15).

Avec ce modèle, les intégrations sont progressives et chaque développement est moins complexe que le précédent. Il peut y avoir des livraisons et mises en oeuvre

après chaque incrément et l'effort est constant dans le temps par opposition au pic pour les conceptions détaillées pour les modèles en cascade ou en V.

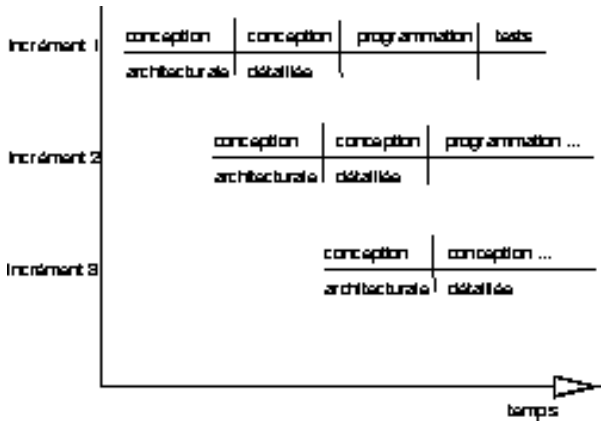


Figure 1.15 : Modèle par incréments

Néanmoins, cette pratique comporte des risques importants comme par exemple la remise en cause du noyau de départ, ou celle des incréments précédents ; ou encore l'impossibilité d'intégrer un nouvel incrément. Afin de réduire ces risques, il est nécessaire de définir les incréments au début du projet et ce de façon globale et il faut que ces incréments soient les plus indépendants possible, aussi bien sur le plan fonctionnel qu'au niveau de la séquence de développement.

III.1.5- Synthèse sur les modèles classiques

L'ensemble de ces modèles classiques de génie logiciel, offre un cadre général très utile pour les concepteurs. Cependant, les interfaces homme-machine n'y sont pas citées même si la démarche les sous-entend. L'analyse et la modélisation des utilisateurs et des tâches humaines ne sont pas préconisées et sont laissées à l'appréciation des concepteurs. Afin de remédier à ces lacunes, d'autres modèles enrichis pour les IHM ont été mis au point pour le développement des applications hautement interactives. Certains de ces modèles sont maintenant présentés.

III.2- Modèles enrichis pour les IHM

L'idée d'enrichissement des cycles de vie pour les IHM, repose sur l'intégration, d'un point de vue méthodologique, des aspects fondamentaux de l'interaction homme-machine comme la modélisation des tâches humaines, la réalisation itérative des prototypes ou l'évaluation des systèmes interactifs. Nous avons choisi de présenter ici trois de ces modèles : le modèle de Long et al., le modèle en étoile, et le modèle ∇ (Nabla). D'autres modèles enrichis sont décrits dans [Kolski et al., 01].

III.2.1- Modèle de Long

Ce modèle [Long et al., 90] est proche du modèle en cascade du génie logiciel classique (figure 1.16), tout en positionnant l'IHM dans l'étape de conception, et en insistant sur l'importance de l'évaluation et des itérations lors du projet.

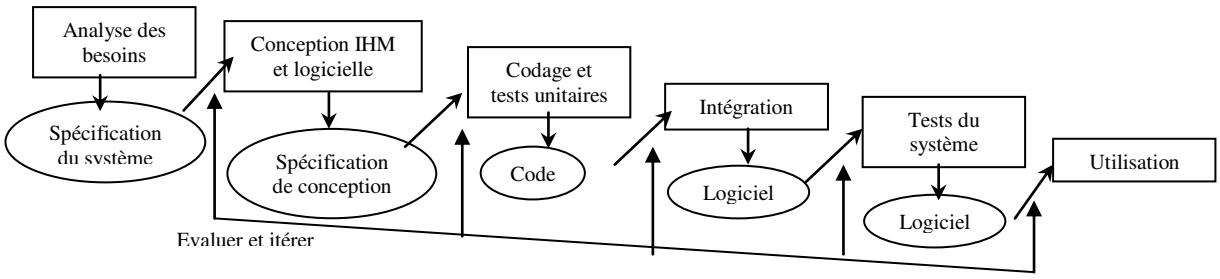


Figure 1.16 : Modèle proposé par Long et al.

Loin d'être parfait, ce modèle s'avère toutefois incitateur sous l'angle des interactions homme-machine, tout en restant très proche d'un modèle classique.

III.2.2- Modèle en étoile

Le modèle proposé par Hartson et al. [Hartson et al., 89], appelé modèle en étoile (figure 1.17) situe l'évaluation au centre même du cycle complet, montrant ainsi des interactions/itérations possibles entre chacune des autres étapes. L'étape d'évaluation est vue comme une étape intermédiaire permettant de protéger l'équipe de développement d'un rejet terminal. Ce modèle se situe assez loin d'un modèle classique mais cette idée le rend intéressant.

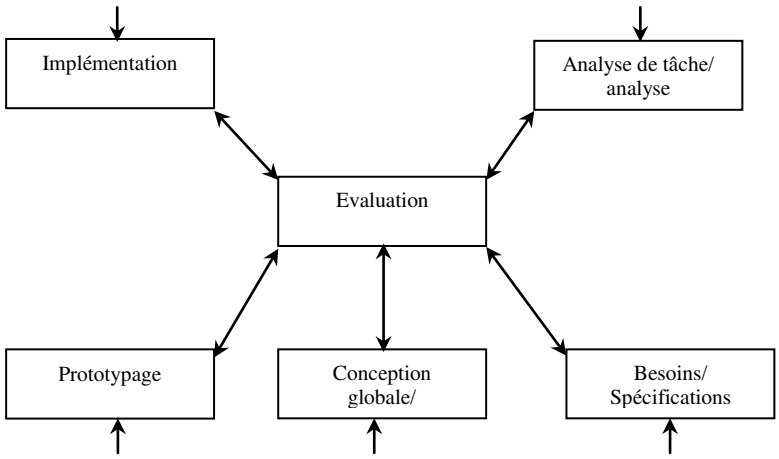


Figure 1.17 : Modèle en étoile

Il n'impose pas *a priori* d'ordre dans l'accomplissement des étapes du processus, bien qu'en pratique, les activités de développement soient reportées en fin de cycle. Il sous-entend une conception participative visant la détection précoce de problèmes d'utilisabilité, requérant une forte adhésion de l'utilisateur par cette implication centrale.

III.2.3- Modèle ∇ (Nabla)

Le modèle ∇ [Kolski, 97] (figure 1.18) a pour objectif de situer les différentes étapes du génie logiciel nécessaires pour développer un système interactif, tout en différenciant l'interface proprement dite (partie gauche du modèle) des modules

applicatifs ou d'aide éventuellement accessibles à partir de ceux-ci (partie droite). Une des caractéristiques marquantes du modèle est de positionner des étapes, inexistantes dans les modèles classiques du génie logiciel, où les facteurs humains devront être considérés par l'équipe de développement.

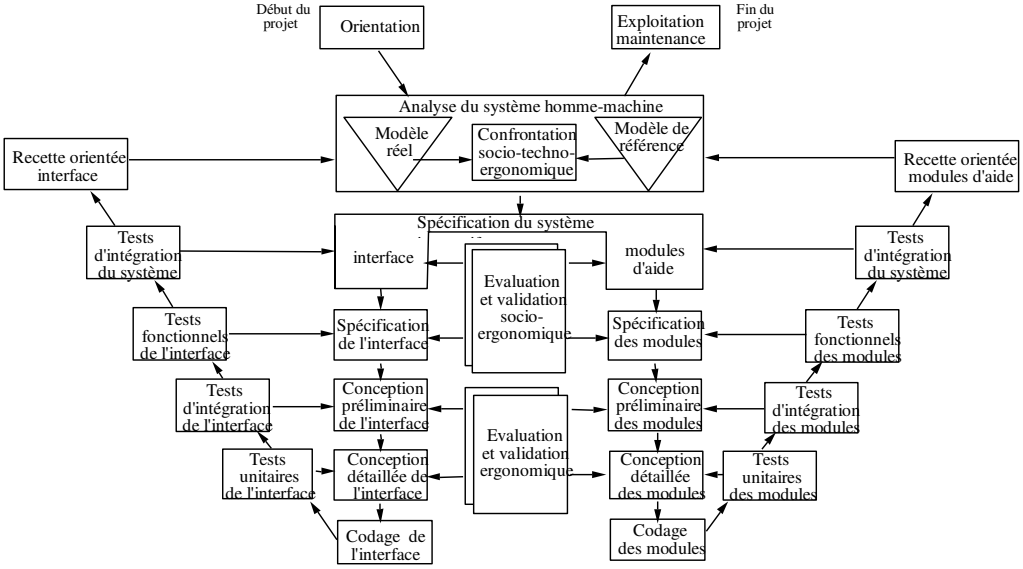


Figure 1.18 : Modèle ∇ (Nabla)

III.2.4- Conclusion sur les modèles classiques et enrichis

Comme nous l'avons vu jusqu'ici, la tâche de conception des SHM et plus particulièrement des IHM constitue un processus assez complexe, plus ou moins facilitée par les cycles de vie proposés, qu'ils soient classiques ou enrichis. Plusieurs démarches peuvent, en effet, être suivies dans la phase de conception ; ces démarches diffèrent selon le type d'application et les problèmes qui lui sont inhérents.

Pour le cas des applications industrielles, parmi les méthodologies de conception existantes, la méthodologie globale de conception et d'évaluation des systèmes homme-machine (SHM) proposée par Millot [Millot, 90][Abed 01] [Lepreux et al., 03], constitue un cadre général très intéressant dans le contexte de nos recherches. La démarche de conception et de génération automatique d'interface adoptée pour ce travail de recherche peut, en effet, entrer dans le cadre général de cette méthodologie. Il importe donc de commencer par sa présentation.

III.3- Démarche globale en U dédiée aux systèmes industriels complexes

Dans la démarche de conception et d'évaluation des SHM proposée par Millot et Abed, plusieurs problèmes fondamentaux liés respectivement à la modélisation du comportement décisionnel humain, à la coopération homme-machine dans les tâches décisionnelles et à la réalisation d'interfaces ergonomiques sont soulevés. Il propose d'intégrer deux systèmes :

(i) un système de surveillance, pour aider l'opérateur dans ses tâches de supervision ;
(ii) un système d'aide à la décision pour l'assister dans ses tâches de diagnostic et de correction.

La démarche méthodologique de conception des SHM comprend deux phases séquentielles :
- la première est une phase conceptuelle descendante de modélisation du SHM qui aboutit à la réalisation du système,
- la deuxième est ascendante ; elle vise à évaluer le système conçu et réalisé dans l'étape précédente.

Nous présentons dans ce qui suit les démarches à suivre dans chacune de ces deux phases de la méthodologie en s'attardant plus particulièrement sur la première relative à la modélisation du SHM puisqu'elle concerne plus particulièrement nos travaux.

III.3.1- Phase conceptuelle descendante de conception d'un SHM

Elle a pour objet de commencer par la modélisation du SHM, pour aboutir à une mise en oeuvre sur site ou à une simulation. La démarche consiste d'abord à analyser le procédé et son système de commande pour en déduire les modes de fonctionnement et de dysfonctionnements prévisibles (figure 1.19).

Ceci conduit à la définition des tâches humaines pour l'exécution desquelles il convient ensuite d'évaluer les capacités humaines à partir du modèle de l'opérateur. Cette évaluation des ressources humaines en regard des contraintes techniques à satisfaire conduit à la définition d'outils d'aide et à l'interface homme-machine [Millot, 99]. Cette étape se compose de cinq étapes.

Etape 1 : Analyse du procédé et du système de commande
A partir d'une description du procédé (en terme de caractéristiques techniques et d'objectifs), il s'agit d'analyser celui-ci et son système de commande selon les différents contextes de fonctionnements prévisibles (normal ou anormal).
On débouche ainsi sur la définition des tâches à effectuer qu'il faudra analyser par la suite. En fait, plusieurs types de tâches peuvent être identifiés à ce niveau : les tâches automatisées prises en compte par des boucles auto-régulées, les tâches partageables pouvant être affectées indifféremment aux automatismes ou à l'opérateur humain et les tâches manuelles spécifiquement humaines et généralement non automatisables suite à des problèmes de disponibilité d'information par exemple ou de difficulté technique ou théorique d'automatisation. Seuls ces deux derniers types de tâches seront considérés dans ce qui suit, dans la mesure où les tâches totalement automatisées ne concernent pas l'intervention de l'opérateur humain et par suite l'IHM.

Etape 2 : Analyse des tâches prescrites
Lors de l'analyse des tâches, une vérification de l'adéquation des capacités humaines aux contraintes engendrées par l'exécution des tâches, doit être faite à partir d'un modèle de l'opérateur (décrivant en particulier ses limites et ressources cognitives). Ce modèle doit permettre, en effet, de rapprocher les ressources humaines disponibles ainsi que les limites intrinsèques des opérateurs sur les plans perceptuels (acquisition et traitement des informations) et physiques (exécution des actions).

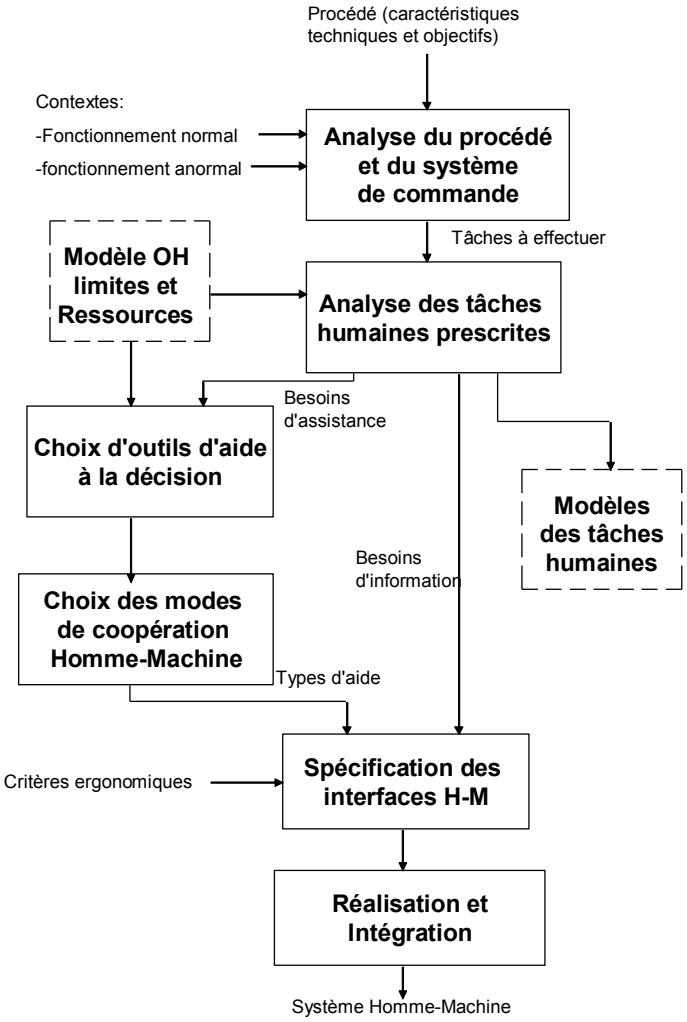


Figure 1.19 : Phase descendante de conception du système Homme-Machine (version de Millot [Millot, 90])

Etape 3 : Spécification des outils d'assistance

Parmi les tâches humaines, certaines tâches peuvent se révéler selon les procédés industriels extrêmement complexes et pourraient être assistées par des outils d'aide (par exemple à la synthèse d'information, au diagnostic, à la reprise de défauts,…). Ces besoins d'assistance déduits à partir de l'analyse de la tâche forment un cahier des charges pour le système d'aide à la décision qu'il faudra intégrer aux systèmes de supervision. Un mode d'intégration devrait alors être défini en précisant les niveaux d'automatisation (choix des outils d'aide) et les modes de coopération entre l'opérateur et l'outil d'aide choisi.

Etape 4 : Spécification de l'Interface Homme-Machine

Cette phase peut être menée, en partie, en parallèle avec la phase précédente de spécification des outils d'assistance. Elle consiste à analyser les besoins informationnels déduits à partir de l'analyse de la tâche pour déduire les informations à afficher sur les écrans de supervision et pour structurer les images correspondantes selon les contextes opérationnels du procédé, en se référant toujours à des critères ergonomiques.

Notons l'existence d'un document de synthèse très intéressant sur la spécification des systèmes interactifs, dressant un bilan des méthodes disponibles, sans lien direct avec un domaine d'application particulier [Jambon et al., 01].

Etape 5 : Réalisation et intégration

C'est la dernière phase de l'étape descendante de conception des SHM. Elle consiste en l'implémentation réelle de l'interface et son intégration avec les différents outils d'assistance et d'automatisation pour constituer ainsi le SHM. Dans la pratique, de manière générale, il existe un vaste choix d'outils de construction des interfaces. Cela est dû aux progrès technologiques, à l'apparition de différents constructeurs et à la diversité des besoins en applications interactives (informatique de gestion, applications internet, applications pour les systèmes temps réel, etc.) [Fekete et al., 01]. Différentes classifications existent dans la littérature. Nous distinguons, ici, principalement : les boîtes à outils, les squelettes d'applications et les générateurs d'interfaces, que nous présentons rapidement :

- *Les boites à outils :* les boîtes à outils restent le support de base pour le développement des interfaces. Elles offrent au programmeur une collection de composants logiciels de base, réutilisables lui permettant de construire son interface [Coutaz, 90]. Une boîte à outils offre un ensemble de composants prédéfinis : des boutons, des menus, des conteneurs (zones scrollables, gestionnaires de positionnement, des fenêtres, des palettes, des canevas, etc.), des composants complexes (zones de listes, zones de texte, zones de dessin, …), des composants de dialogue (outils d'alertes, sélecteurs de couleurs, …). Toute boîte à outils repose sur un langage, un système d'exploitation et un environnement d'utilisation (GUI du système hôte). Plusieurs exemples existent tels que : IlogViews [Ilog, 94], X/Motif [Scheifler et al., 86], Java AWT [Geary et al., 97], MacOS [Apple, 94], etc.

- *Les squelettes d'applications :* un squelette d'application appelé également gestionnaire de dialogue ou serveur d'interface se présente comme un assemblage logiciel réutilisable et extensible qui assure la plupart des fonctions de l'interaction avec l'utilisateur. Contrairement aux boîtes à outils, les squelettes d'application imposent une architecture prédéfinie partageable par plusieurs applications. Cette architecture regroupe un module de contrôle (le noyau d'exécution) et un module de présentation (les services et techniques d'interaction). Le programmeur aura à raffiner quelques parties et/ou en définir d'autres pour une application particulière. Les squelettes d'application reposent sur la programmation objet. Le premier squelette d'application était MacApp [Schmucker, 87] qui définit les classes abstraites Application, Document et View. Les plus connus actuellement sont : Java Swing [Geary et al., 97], Microsoft Foundation Classes (MFC) [Microsoft, 00] et OpenStep [Next, 92] d'Apple.

- *Les générateurs d'interfaces :* bien que les squelettes d'application réduisent sensiblement le coût du développement des interfaces par rapport aux boîtes à outils, ils restent toutefois inadaptés à des cas particuliers d'applications. Ce qui nécessitera le recours aux services de la boîte à outils autour de laquelle le squelette a été créé. La tâche du programmeur devient alors assez difficile. Les générateurs d'interfaces viennent à son secours en lui permettant la spécification de l'interface. Leur principe consiste à générer les différents composants de

l'interface ainsi que leurs liens à partir des spécifications de l'application exprimées au moyen de formalismes. Deux classes de générateurs d'interfaces existent : les générateurs interactifs et les générateurs automatiques.

- Les générateurs interactifs, reconnus en général sous le nom de UIMS (User Interface Management System), offrent à l'utilisateur la possibilité de créer son interface en choisissant les fenêtres et les différents composants, avec leurs attributs tels que la couleur, la taille, etc. En effet, les UIMS ne font que traduire l'interface créée sous forme de code réutilisable par l'application correspondante. Ils ne permettent donc de générer qu'un seul composant de l'interface : la présentation. Ils ne touchent, cependant, ni le niveau syntaxique (l'aspect dialogue), ni le niveau sémantique de l'application.

- Les générateurs automatiques (ou semi-automatiques) se distinguent des générateurs interactifs par le fait que c'est le système lui même qui peut choisir pour l'utilisateur tel type de fenêtre, de menu ou de composant et ce, à partir des spécifications fournies par le concepteur. Ils essayent de couvrir les trois composants de l'interface et spécialement le composant du contrôleur de dialogue. Ils sont le plus souvent basés sur des approches à base de connaissances. En particulier, pour la conception et/ou l'évaluation automatique des interfaces dédiés au contrôle des procédés industriels, l'outil Ergo-Conceptor [Moussa, 92], déjà cité précédemment, est capable de décider des vues graphiques appropriées à associer à chaque sous-système (voir plus loin). Les TFWWG (Tools for Working With Guidelines ; cf aussi § I.3) entrent dans le cadre de cette classe d'outils en visant à assurer la meilleure qualité ergonomique possible. En effet, ces outils consistent à formaliser des règles ergonomiques et à les intégrer au niveau du système générateur de l'IHM. Plusieurs travaux ont été réalisés dans ce sens, citons principalement les travaux de Vanderdonckt [Vanderdonckt, 99]. Notre recherche se situe également dans cette voie pour développer une approche à base de modèles permettant la génération semi-automatique des interfaces de contrôle des procédés industriels complexes [Szekely, 96].

La phase descendante de conception du SHM est ainsi achevée. Il importe, par la suite, d'évaluer le système conçu et réalisé dans cette phase. C'est le rôle de la phase ascendante d'évaluation du SHM que nous résumons maintenant.

III.3.2- Phase ascendante d'évaluation d'un SHM

Dans cette phase (figure 1.20), l'objectif est d'évaluer le SHM réel ou simulé. D'après Millot, les critères d'évaluation tiennent compte des performances du SHM global, exprimées par exemple en terme d'écart entre la production réelle et les objectifs, et également de critères ergonomiques permettant d'évaluer les difficultés rencontrées par les opérateurs lors de l'exécution de leurs tâches. La non observabilité directe des paramètres significatifs des difficultés humaines complique considérablement cette tâche. Il importe donc d'en construire des observateurs à partir des paramètres directement observables (séquences d'actions, etc.), des paramètres estimés par des méthodes subjectives (charge de travail, fatigue, etc.) et des paramètres estimés à partir d'un modèle théorique de l'activité cognitive de l'opérateur humain. Tous ces paramètres doivent être évalués au moyen de méthodes d'analyse d'activité.

Plusieurs travaux sont menés dans ce cadre, proposant des démarches *d'évaluation a priori* (avant la création) ou *a posteriori* (après la création) de l'interface homme-machine. Nous reviendrons sur cette notion *d'évaluation a priori* par la suite lorsque nous parlerons de la validation en rapport avec la démarche de conception que nous proposons dans ce mémoire.

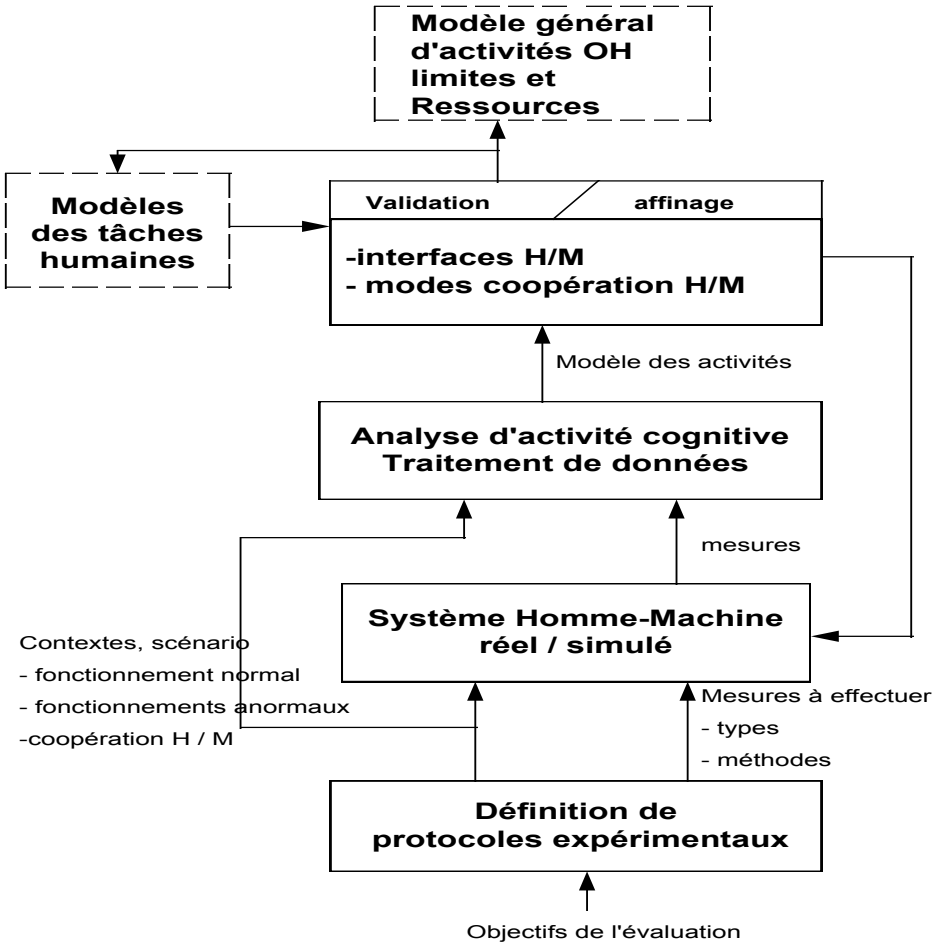


Figure 1.20 : Etape ascendante d'évaluation du système Homme-Machine (version de Millot [Millot, 90])

Le cadre général de cette démarche de conception et d'évaluation des SHM a le mérite de mettre l'accent explicitement sur une phase cruciale qui est la conception des IHM. Celle-ci doit être considérée comme une discipline à part entière ayant ses méthodes et ses outils.

A ce sujet, différentes approches et méthodologies sont proposées dans la littérature pour la conception des IHM. Nous proposons de synthétiser et discuter ci-dessous quelques unes de ces approches avant de passer plus loin à la présentation de notre contribution.

IV. Approches et méthodologies de conception des IHM proposées dans la littérature

Sans souci d'exhaustivité, nous présentons maintenant plusieurs travaux de recherche représentatifs dans le domaine des IHM, nous semblant, pertinents vis-à-vis de nos objectifs généraux.

IV.1- Travaux de Palanque et Bastide : le modèle ICO

Palanque et Bastide [Palanque, 97] [Palanque et al., 97] proposent de modéliser les interfaces des applications interactives en utilisant le formalisme ICO (Interactive Cooperative Objects). Celui-ci est basé sur une combinaison de techniques à objets et des réseaux de Petri ; il permet une spécification semi-formelle de l'interface et de son comportement, facilitant une analyse fonctionnelle de l'interface pour validation avant l'implémentation.

Ces auteurs proposent une architecture conceptuelle des applications interactives se basant sur trois types d'objets (figure 1.21) :

- Les objets passifs : ils n'ont pas d'activité spontanée, ni d'invocation de services des autres objets. Ils correspondent en général aux structures de données de l'application sans comportement, ni présentation.
- Les objets coopératifs (OC) : ce sont des objets non passifs mais aussi non interactifs, qui sont responsables des traitements et accèdent aux objets passifs. Ils ont un comportement à décrire mais pas de présentation puisqu'ils ne sont jamais activables par l'utilisateur.
- Les objets coopératifs interactifs (ICO) : ils sont en relation directe avec l'utilisateur et pour lesquels on doit décrire le comportement et la présentation. Ces objets modélisent en fait les fenêtres de l'interface de l'application interactive.

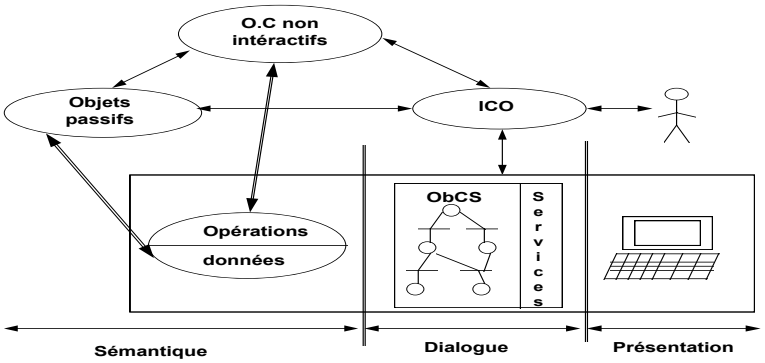


Figure 1.21 : Le modèle ICO

Un objet coopératif interactif, élément clé de cette approche, se définit par :
- la syntaxe de son interface : types, constantes, attributs et services et,
- la pragmatique qui définit sa structure de contrôle : l'ObCS (Object Control Structure) décrit par les Réseaux de Petri à Objets (RPO).

La démarche proposée pour la conception des applications interactives à l'aide du formalisme ICO consiste à :

1- Modéliser le noyau fonctionnel de l'application interactive, ce qui revient à concevoir, à partir de la spécification informelle de l'application, toutes les données du système en termes d'objets, et à identifier les différents types d'objets : passifs, coopératifs et coopératifs interactifs (ICO).
2- Modéliser la présentation en termes d'ensembles de fenêtres. Chacune d'elles correspondra à la présentation de l'un des ICO conçus dans l'étape précédente. A chaque fenêtre, on associe un ensemble de "widgets" assurant un service donné au niveau de l'interface.
3- Modéliser l'espace du dialogue en décrivant le comportement global de chaque fenêtre par un réseau de Petri à objets non contraint, précisant les relations entre les différents objets de la fenêtre.
4- Définir les relations présentation/dialogue en termes de fonctions d'activation des widgets par l'utilisateur.
5- Vérifier sur le modèle les principales propriétés de l'interface relatives à la disponibilité des services, l'absence de blocage, la réinitialisation, etc., avant de passer à l'implémentation concrète de l'interface.

La démarche proposée par ces chercheurs présente l'avantage d'utiliser des spécifications formelles au moyen de techniques telles que les RdP, permettant la validation de la spécification de l'interface avant son implémentation. Elle hérite des avantages des approches objets tels que la réutilisabilité et la modularité. Cependant, la validation de modèles basés sur les réseaux de Petri à objets n'est pas évidente. En effet, afin de pouvoir valider un tel modèle, il est absolument nécessaire de préciser les règles et les conditions permettant la transition d'un réseau de Petri à objets vers un réseau de Petri ordinaire sur lequel la vérification des propriétés pourra être menée. Ce qui n'apparaît pas clairement dans cette approche. De plus, la méthode n'explique pas le principe d'identification des objets de l'interface. Elle laisse cette tâche au bon sens du concepteur de l'interface et ne lui fournit aucun guide pratique à notre connaissance.

IV.2- Travaux de De Rosis : le formalisme XDM

Un formalisme nommé XDM (pour conteXt sensitive Dialogue Modelling) a été proposé par De Rosis et Pizzutilo pour la description formelle et l'évaluation des IHM [De Rosis et al., 98]. Le formalisme est basé sur une extension des réseaux de Petri et utilise la théorie des opérateurs KLM proposée par Card [Card et al., 83]. Le formalisme proposé permet : (1) la description des différents aspects statiques et dynamiques de l'interaction dans les différents contextes de fonctionnement du système, (2) une évaluation pré-empirique de l'interface.

La méthode a été développée dans le cadre d'un projet de prototypage itératif d'un système médical à base de connaissances. Le formalisme permet la translation des résultats de l'analyse de la tâche vers la conception de l'interface en décrivant un ensemble d'aspects essentiels de l'interaction, à savoir :
- Comment l'information relative à un état donné du système doit apparaître dans les différentes phases de l'interaction, ce qui revient à décrire les liens avec les objets d'interaction nécessaires ;
- Quelles actions l'utilisateur peut accomplir à un moment donné, ce qui revient à identifier les techniques d'interaction possibles pour chaque état ;

- Comment ces aspects physiques de l'interaction sont reliés aux objets de l'interface et aux tâches à accomplir.

Les étapes de leur démarche se présentent comme suit :
1- Décrire la dynamique du système interactif avec un RdP.
2- Eclater les transitions pour représenter la hiérarchie des tâches.
3- Décrire les aspects statiques de l'interaction (les états du système et les actions utilisateur) par association des projections logiques et physiques aux différentes places et transitions du RdP. Les projections logiques concernent l'association des besoins informationnels des opérateurs (BIO) aux places et des tâches aux transitions. Quant aux projections physiques, elles concernent la spécification des aspects de l'implémentation à savoir les objets et techniques d'interaction.
4- Ajouter les conditions nécessaires aux places et transitions et les couleurs aux marquages permettant de représenter l'adaptabilité du réseau à des différents contextes de fonctionnement.

Le formalisme XDM proposé permet donc la modélisation du dialogue homme-machine en précisant les propriétés statiques et dynamiques de l'interface et la simulation de son comportement dans les différents contextes de fonctionnement possibles. Et par la suite, il permet donc la vérification de quelques propriétés de l'interface telle que la complétude, l'absence d'ambiguïté, la consistance et la complexité de l'interface. Afin d'évaluer ce dernier critère, les auteurs proposent ici d'intégrer au formalisme la théorie KLM, permettant l'estimation du temps nécessaire pour l'accomplissement d'une action élémentaire.

Pour modéliser le dialogue, XDM constitue une méthode appropriée permettant la vérification *a priori* de quelques propriétés de l'interface. Cependant, la manière d'identifier les besoins informationnels des opérateurs et la répartition de ceux-ci au niveau des places du réseau n'apparaît pas clairement dans l'approche. Le choix des objets graphiques pour la constitution de l'interface est laissé au concepteur sans fournir aucun guide particulier. En outre, la modélisation du comportement intrinsèque des objets graphiques de l'interface n'est pas prise en considération. On se demande, donc, si la validation, à un haut niveau d'abstraction, de l'interface sans prise en compte du comportement des objets graphiques peut être considérée comme irréfutable.

IV.3- Travaux de Tabary et al. : la méthode TOOD

La méthode TOOD (pour Task Object Oriented Description) propose une approche de conception des IHM basée sur une décomposition hiérarchique du SHM en tâches modélisées par réseaux de Petri à objets [Mahfoudhi, 97] [Tabary et al., 98]] [Tabary, 01] [Abed, 01]. L'idée sur laquelle se sont basés les travaux de TOOD est que la spécification des IHM peut être définie à différents niveaux d'abstraction :
- Le niveau le plus haut : il est décrit à travers l'analyse de la tâche prescrite et conduit aux Besoins Informationnels de l'Opérateur (BIO) définis par le déroulement du dialogue,
- Le niveau le plus bas : il revient à décrire les aspects syntaxique et lexical. Il est préconisé ce niveau d'abstraction d'utiliser des "Kits de construction" tels que les boîtes à outils, les gestionnaires des fenêtres, les éditeurs d'interface, etc., qui offrent au concepteur le moyen de construire directement les interfaces graphiques

et de les tester rapidement à travers des éditeurs. Pour entamer cette construction, il suffit que le concepteur classe les BIO selon des critères de similarité en des classes d'objets appelés Objets Interactifs.

La méthode TOOD s'articule principalement autour de deux modèles : le modèle structurel et le modèle opérationnel. Le modèle structurel consiste à identifier et décrire les besoins et caractéristiques des tâches utilisateurs. Une tâche opérateur est définie comme étant un ensemble de traitements exécutés pour atteindre un but donné dans des conditions données. Les besoins des tâches sont définis selon quatre types :
- Les besoins relatifs aux déclenchements.
- Les besoins informationnels.
- Les contraintes de l'exécution.
- Les ressources humaines et matérielles.

Ces chercheurs proposent de profiter des avantages de l'approche orientée objet pour définir des objets tâches exploitant la notion d'événement. Ils ont utilisé pour cela un format extension de SADT pour représenter l'objet tâche intégrant une structure de contrôle de la tâche appelée TCS (Task Control Structure) modélisée par les Réseaux de Petri à Objets.

Les besoins informationnels de l'opérateur (BIO) sont déduits à partir des ressources relatives aux objets tâches élémentaires. A partir de ces BIO, on spécifie les objets nécessaires pour l'interface à réaliser. En effet, le deuxième modèle de la méthode TOOD, appelé modèle opérationnel, consiste à spécifier l'interface proprement dite. Ce modèle présente alors deux niveaux :
- un niveau local qui consiste à construire un modèle de l'utilisateur en termes de stratégies et plans d'actions et spécifier par la suite le modèle de l'interface an rapport avec les tâches élémentaires décrites dans le modèle structurel, et
- un niveau abstrait qui consiste à produire par agrégations des modèles locaux une spécification de l'interface à haut niveau d'abstraction en termes d'objets, d'états, d'actions et de comportements.

Une méthode de conception orientée objet de l'interface est aussi proposée par ces auteurs pour l'implémentation. Cependant, il s'agit de signaler que plusieurs problèmes pour la génération de l'interface restent encore non résolus : la méthode ne montre pas clairement comment procéder pour spécifier les vues graphiques ni comment elles sont générées. En outre, TOOD ne considère pas l'aspect des dysfonctionnements dans la partie analyse des SHM ce qui rend l'analyse de la tâche dissociée des différents contextes de fonctionnement du système. De même, similairement à ICO, la vérification des propriétés des réseaux de Petri à objets mérite de plus amples approfondissements.

IV.4- Travaux de Ait Ameur et al. : la méthode B

Ait-Ameur et al. visent à exploiter la méthode B pour la spécification des systèmes interactifs [Ait-Ameur et al., 98]. La méthode B est basée sur une description formelle du modèle tout comme les méthodes Z et VDM. Ces méthodes consistent en effet à définir un modèle du système utilisant des attributs de variables caractérisant le système, des invariants que le système devrait satisfaire et les différentes opérations altérant ces variables. La méthode B est basée sur la logique des prédicats et sur la

faible pré-condition de calcul. Elle permet de supporter des spécifications à travers des machines abstraites aussi bien que les raffinements et l'implémentation. Avec cette méthode, il est possible d'assurer tout le développement dans un langage commun avec la même sémantique et les mêmes preuves techniques. Elle fournit ainsi une approche uniforme pour la description des programmes de développements.

Cette approche consiste à fournir une méthode « indépendante du modèle » qui pourra par la suite être adaptée (ou personnalisée) aux modèles spécifiques d'IHM. Le point fort de la méthode B est sa capacité de raffinement. En effet, la possibilité qu'offre cette méthode pour dériver des programmes concrets à partir des spécifications abstraites est très importante. Cependant, cette méthode n'explique pas la manière à suivre pour constituer les besoins informationnels de l'opérateur (BIO) ni comment décider des objets de l'interface. La génération effective de l'interface nécessite aussi plus de développement.

Notons aussi que la principale critique apportée aux techniques telles que la méthode B, Lustre ou Lotos concerne la difficulté qu'on rencontre dans la lecture et la manipulation de leurs notations.

IV.5- Travaux de Rodriguez : le projet ALACIE

F.Gamboa-Rodriguez propose dans ses travaux de recherche un Atelier Logiciel d'Aide à la Conception d'Interfaces Ergonomiques (ALACIE) s'appuyant sur une méthodologie de conception cyclique fondée sur la modélisation de la tâche et le principe de prototypage des interfaces [Rodriguez, 98]. Le projet ALACIE se base sur la collaboration des spécialistes de disciplines différentes (principalement de l'analyse de la tâche et de l'ergonomie des interfaces) et par suite sur deux modèles complémentaires :

- Un modèle MAD* (une extension du modèle MAD, présenté au § I.2.1 intégrant la notion de temps et la gestion des interruptions) permettant la modélisation de la tâche utilisateur ainsi que sa simulation,
- Un modèle SSI (Spécification Sémantique de l'Interface) permettant la spécification sémantique abstraite de l'interface et ce, en traduisant les éléments des tâches principales en éléments sémantiques de l'interface et en assignant une représentation physique à chacun de ces éléments.

L'approche proposée se base donc sur ces deux modèles et est subdivisée en cinq étapes :

1- observation de l'activité de l'utilisateur,
2- modélisation de la tâche utilisateur,
3- révision et validation de la tâche modélisée,
4- extraction des caractéristiques importantes,
5- génération d'un ou de plusieurs prototypes de la future interface à partir des informations recueillies, des règles et principes ergonomiques et de la propre expérience de l'ergonome travaillant sur le projet.

F.Gamboa-Rodriguez a cherché principalement dans ses travaux à apporter des solutions à la problématique de l'intégration de l'analyse de la tâche dans les cycles de conception des interfaces. L'apport des ergonomes dans l'approche ALACIE

représente une ouverture intéressante, quoique empirique, à la fluidité du passage de l'Analyse de la Tâche à la spécification des IHM. Néanmoins l'étape délicate de l'extraction des caractéristiques importantes et des BIO à partir des résultats de l'analyse de la tâche reste non formalisée.

IV.6- Travaux de Bodart et al. : le projet TRIDENT

Le projet TRIDENT (Tools foR Interactive Development environmENT) a été défini dans le cadre de la conception d'une méthodologie de développement d'applications interactives de gestion [Bodart et al., 95]. Le but est de fournir une structure générale couvrant différents styles d'interaction (menus de sélection, langage de commande, manipulation directe, etc.) et différentes structures de dialogue (synchrone/ asynchrone, etc.). Cette structure générale sera, ensuite, spécialisée en structures affinées selon les classes de problèmes rencontrés.

Ces auteurs précisent que la spécification de l'interface utilisateur doit se baser sur l'analyse de la tâche, à partir de laquelle on identifie les besoins informationnels et fonctionnels ainsi que le graphe d'enchaînement des activités pour pouvoir définir par la suite les objets de l'interface (figure 1.22). Cependant, ils ne fournissent pas de méthode par laquelle on peut arriver à identifier les BIO à partir de l'analyse de la tâche, ni le processus permettant l'identification des objets de l'interface à partir de ces BIO. Ce passage délicat de l'analyse de la tâche à la spécification de la présentation reste flou, aucune indication ou règle n'est donnée à ce sujet.

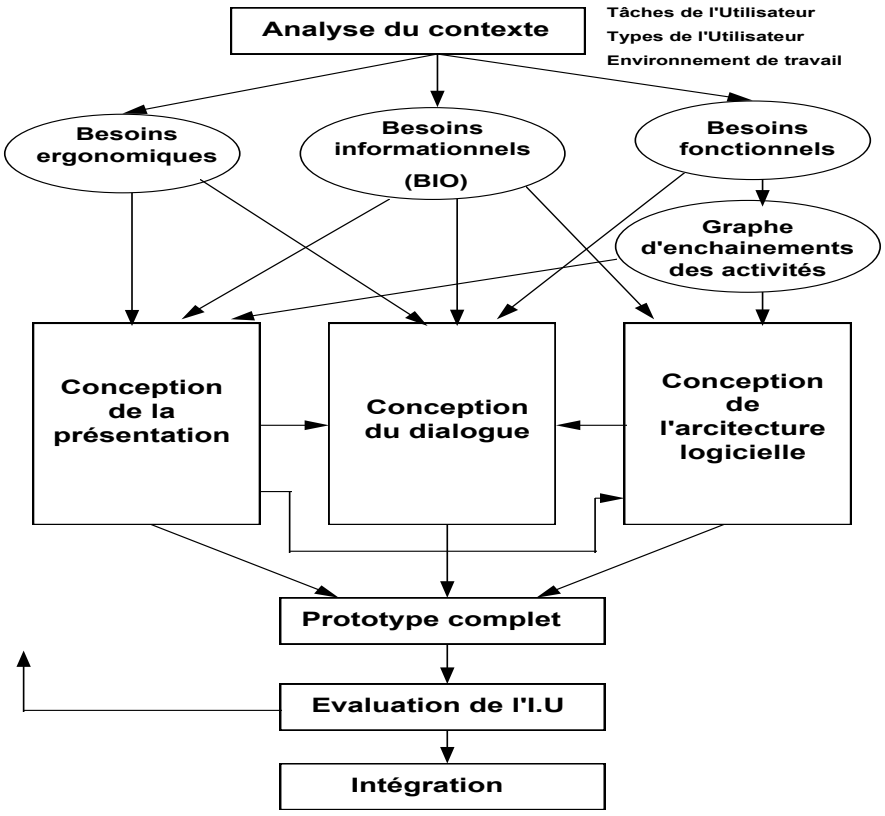


Figure 1.22 : Structure Méthodologique de TRIDENT

La structure méthodologique donnée par le projet TRIDENT est assez générale. Elle ne précise pas le formalisme à adopter pour la spécification du dialogue. Ce qui met les concepteurs d'interfaces, et particulièrement les novices, dans l'embarras du choix du formalisme approprié.

IV.7- Travaux de Moussa : le système Ergo-Conceptor

Concernant le domaine particulier des applications industrielles, le système Ergo-conceptor présente une approche ergonomique de conception des IHM [Moussa, 92]. La démarche suivie par Ergo-conceptor a pour objectif de fournir des spécifications de l'imagerie incluant des facteurs ergonomiques, utiles à la génération des vues. Ces spécifications sont déduites à partir d'une part des données techniques du système complexe dont on veut construire les interfaces de supervision et de ses objectifs ; et d'autre part des résultats issus de l'analyse de la tâche, des spécifications de l'imagerie incluant des facteurs ergonomiques, utiles à la génération des vues. Trois étapes sont alors prévues dans ce système (figure 1.23).

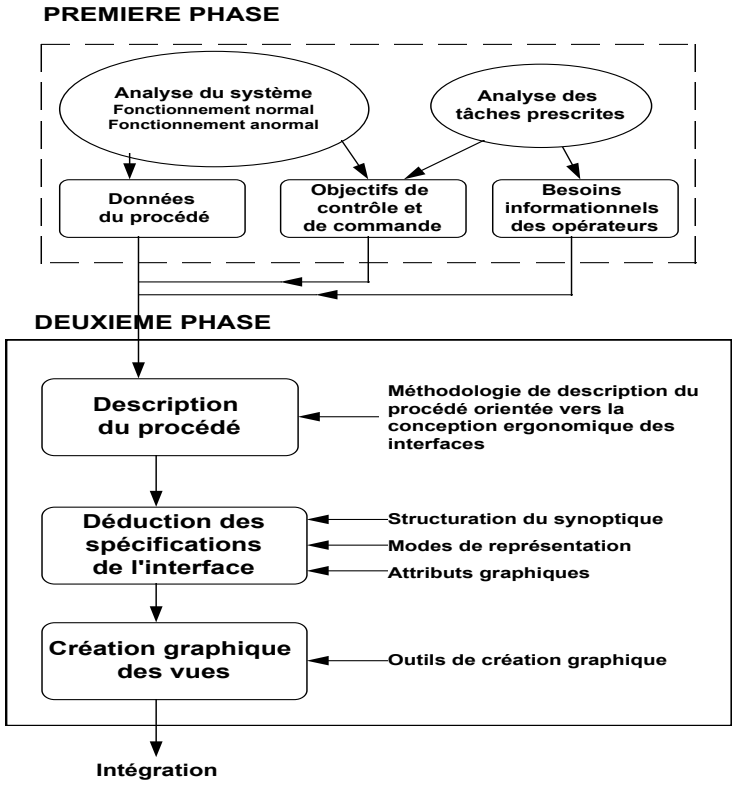


Figure 1.23 : Démarche d'Ergo-Conceptor

- *La première étape* consiste à décrire un procédé sur les deux plans : fonctionnel et structurel selon une méthodologie de description qui consiste à décrire un procédé, tout en visant à aboutir à un ensemble suffisant de données pour spécifier ensuite l'imagerie. La description peut s'effectuer selon trois axes complémentaires apportant chacun une description de nature différente : une description en terme de hiérarchie d'abstraction "moyens-objectifs", une description structurelle et une description fonctionnelle.
- *La deuxième étape* est consacrée à la spécification de l'interface. A cet effet, les besoins informationnels peuvent être répartis en vues décrites à l'aide d'une

base de connaissances ergonomiques. A la fin de cette étape, un cahier des charges informatisé des spécifications peut être généré.

- *La troisième et dernière étape* s'intéresse à la création interactive des vues pour laquelle le développeur se sert d'un éditeur graphique spécialisé intégrant le cahier des charges généré lors de l'étape 2, il peut alors éventuellement modifier les attributs graphiques des objets à sa guise.

Dans sa première version datant du début des années 90, le système Ergo-Conceptor part de l'idée de description des besoins informationnels des opérateurs ainsi que des objectifs de contrôle et de commande issus de l'étape de l'analyse du système homme-machine.

Bien que représentant une alternative assez originale dans le domaine de la conception des interfaces, ce système a aussi démontré certaines limites pour ce qui est de sa première version. En effet, Ergo-Conceptor ne génère que des vues statiques, en ce sens que ces vues ne gèrent pas le dialogue homme-machine mais plutôt l'impact des variables de contrôle et de commande du système par rapport à l'interface. En outre, la méthodologie utilisée pour la description du procédé ne permet pas de gérer certains aspects fonctionnels importants tels que le parallélisme ou encore la synchronisation et ne permet pas non plus de valider mathématiquement le modèle de l'interface décrit vu son aspect assez informel. De tels constats ont été pris en considération dans le cadre de nos recherches pour étendre la démarche suivie par Ergo-Conceptor.

A la lumière de ce panorama de méthodes de conception des IHM, une étude critique peut être faite. Cette étude est maintenant présentée.

V. Synthèse et identification des nouveaux besoins

V.1- Tableau récapitulatif

Les résultats de l'étude bibliographique sont présentés dans la table 1. Trois remarques importantes sont à souligner :

Tableau 1.1 : Evaluation de quelques approches proposées pour la conception de l'interface

	Palanque (ICO)	***Abed et ses collègues (TOOD)***	***De Rosis (XDM)***	***F.G.Rodriguez***	***Ait-Ameur and al.***	***Moussa (Ergo-Conceptor)***
Analyse de la tâche	*Non*	*Oui*	*Non*	*Oui*	*Non*	*Non*
Analyse des dysfonctionnements	*Non*	*Non*	*Non*	*Non*	*Non*	*Non*
Expliquer comment identifier les BIO	*Non*	*Oui*	*Non*	*Non*	*Non*	*Non*
Considérer les BIO dans la conception de l'interface	*Non précisé*	*Oui*	*Oui*	*Oui*	*Non*	*Oui*
Considérer les critères ergonomiques	*Non précisé*	*Non précisé*	*Non précisé*	*Oui*	*Non précisé*	*Oui*
Utiliser une technique formelle	*Oui (réseaux de Petri)*	*Oui (Réseaux de Petri)*	*Oui (Réseaux de Petri)*	*Oui (MAD*)*	*Oui (B)*	*Non*
Fournir un outil informatique	*En progrès*	*En progrès*	*Oui*	*Oui*	*Oui*	*Oui*
Valider les spécifications	*Oui (Partiellement)*	*Oui (Partiellement)*	*Oui*	*Oui (prototypage)*	*Oui*	*Non*
Générer automatiquement l'interface	*Dialogue*	*Oui (partiellement)*	*Non*	*Non*	*Non*	*Vues graphiques*
Fournir une approche complète et intégrée	*Non*	*Perspective*	*Non*	*Non*	*No*	*Non*

1- Aucune de ces approches ne considère l'aspect crucial de l'analyse des dysfonctionnements du système. De ce fait, les interfaces produites ne seront adaptées qu'aux situations de fonctionnement normal. Un dysfonctionnement pourrait, dans ce cas, ramener le système à une situation critique imprévisible.
2- La plupart des approches discutées ici, considèrent le concept de l'analyse de la tâche. Néanmoins, peu d'entre elles expliquent comment exploiter et intégrer les résultats de cette analyse de la tâche systématiquement dans leurs modèles.
3- Mis à part l'outil Ergo-Conceptor, aucune des approches proposées ne vise à automatiser l'utilisation des recommandations ergonomiques dans la spécification de l'interface.

Notre objectif est donc de surmonter les limites de la plupart de ces systèmes et de proposer une approche globale de conception et génération semi-automatique des IHM pour le contrôle des procédés industriels.

V.2- Guide méthodologique

Une "approche globale" de conception et génération automatique des IHM consisterait à satisfaire le cahier des charges :

- *considérer l'analyse de la tâche. En effet, les BIO varient en fonction du contexte de fonctionnement et du degré de difficulté de la tâche.*
- *Identifier les BIO à partir de l'analyse de la tâche et des états du système. Ainsi on présentera instantanément à l'opérateur l'information nécessaire adéquate.*
- *Déduire les objets de l'interface. Ceci peut être automatiquement réalisé en fonction des BIO.*
- *Utiliser des outils informatisés basés sur les recommandations ergonomiques pour spécifier graphiquement les objets déduits de l'interface. Des critères spécifiques définis au niveau ces recommandations devraient garantir une ergonomie de présentation et de dialogue.*
- *Utiliser des techniques formelles pour spécifier l'interface, ses objets graphiques et leurs comportements.*
- *Assister le concepteur de l'interface tout au long des différentes étapes de l'approche avec des outils informatiques.*
- *Valider théoriquement les spécifications avant de procéder à la génération effective de l'interface.*

Dans l'approche proposée, nous nous intéressons particulièrement à l'identification des BIO à partir de l'analyse du SHM et au processus de conception de l'interface tenant compte de ces BIO. Cette approche est composée de sept étapes (figure 1.24) :

1. Une première analyse préliminaire du processus et de son système de commande est nécessaire. Cette analyse fournit un document contenant les données du procédé et ses différentes contraintes techniques et fonctionnelles.
2. La seconde étape consiste à analyser le SHM en termes de processus, son système de commande et les tâches opérateur. Différentes techniques et méthodes sont proposées à cet effet.

3. La troisième étape consiste en la modélisation du comportement de l'opérateur. Cette modélisation doit exprimer l'interaction entre l'opérateur et l'interface.
4. La quatrième étape assure l'identification des BIO en termes d'objets d'interface.
5. Une fois les objets de l'interface déduits, cette étape consiste à spécifier l'interface en termes de vues et objets graphiques.
6. Les spécifications générées sont vérifiées, au niveau de cette étape, tirant bénéfice de la technique formelle utilisée pour la spécification de l'interface.
7. La dernière étape de cette approche est dédiée à la génération semi-automatique de l'interface.

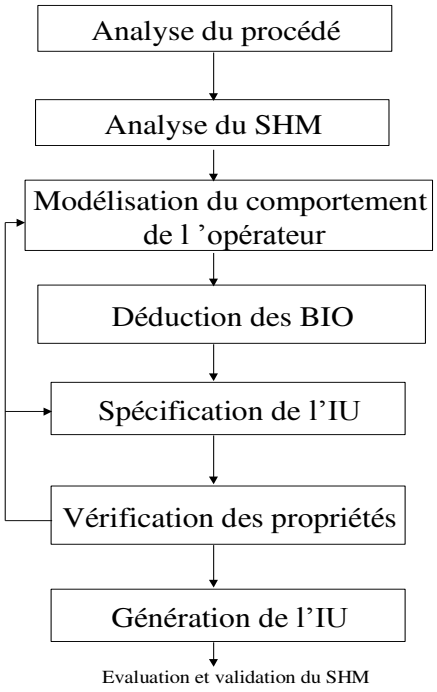


Figure 1.24 : Squelette méthodologique de l'approche proposée

Cette démarche est développée et détaillée au niveau du chapitre 3.

Conclusion

Nous avons introduit, au niveau de ce premier chapitre, le cadre général du domaine de notre recherche et donné un aperçu sur les aspects généraux de l'interaction homme-machine : à savoir les approches de modélisation de l'opérateur humain, d'analyse et modélisation de la tâche humaine, tout en insistant sur l'apport de connaissances provenant de l'ergonomie cognitive dans le domaine de l'interaction homme-machine.

Nous avons développé par la suite les aspects de génie logiciel pour les IHM en discutant des principaux modèles de cycle de vie classiques et enrichis pour l'IHM. La présentation du cadre général de conception et d'évaluation des SHM, proposé par Millot et Abed et adopté dans le cadre de nos travaux de recherche, a fait l'objet de la troisième partie de ce chapitre.

Notre intérêt porte sur l'une des étapes de cette démarche, à savoir la conception des IHM. Une étude bibliographique, portant sur les principales approches de conception des IHM, a été présentée dans ce sens. Ces approches ont été synthétisées et discutées à la fin de ce chapitre et un cahier des charges d'une "approche globale" a été proposé.

La concrétisation de cette approche représente l'objet de ce travail de recherche. Ainsi, dans ce qui suit, un état de l'art des outils et modèles de conception des IHM est proposé. Grâce à cette étude bibliographique, les choix permettant de mettre en œuvre notre approche de conception des IHM seront établis.

Chapitre 2

Méthodes, outils, modèles et formalismes de base venant en support des approches méthodologiques

Plan du chapitre

Introduction
I. Méthodes et techniques d'analyse des systèmes en mode de fonctionnement normal
    I.1- Approches fonctionnelles (cartésiennes)
    I.2- Approches systémiques
    I.3- Approches objets
II. Méthodes et techniques d'analyse des systèmes en mode de fonctionnement anormal
    II.1- Méthode d'analyse préliminaire des dangers (APD)
    II.2- Méthodes d'analyse et regroupement des pannes
    II.3- Méthodes d'analyse détaillée causes/effets d'une panne
III. Outils formels de modélisation utilisés dans les SHM
    III.1- Apports et limites des modèles formels
    III.2- Outils formels proposés dans les expériences menées dans le domaine des spécification des IHM
IV. Synthèse et choix des méthodes et outils
Conclusion

Chaque approche méthodologique de conception des IHM devrait être supportée par un ensemble de méthodes, outils et modèles permettant de l'appliquer. La phase d'analyse du SHM représente, en effet, un pré-requis nécessaire à la conception des IHM. Plusieurs méthodes peuvent être appliquées à ce sujet. Certaines sont dédiées à l'analyse en mode de fonctionnement normal. D'autres sont destinées à l'analyse en mode de fonctionnement anormal. Nous commençons, dans un premier temps, par synthétiser l'ensemble de ces méthodes.

Nous présentons, par la suite, un panorama d'outils formels possibles pour la modélisation des systèmes. Nous proposons, à la fin, une synthèse de ces différents outils et modèles en précisant les choix adoptés pour supporter l'approche de conception des IHM proposée dans le cadre de cette recherche.

I. Méthodes et techniques d'analyse des systèmes en mode de fonctionnement normal

La complexité croissante des procédés fait qu'une analyse fonctionnelle et/ou structurelle d'une installation s'avère souvent fastidieuse, voire impossible, sans l'adoption d'une ou de plusieurs méthodes. De nombreuses méthodes d'analyse et de modélisation du système et des tâches sont disponibles [Larvet, 94][Calvez, 93]. Trois principales classes de méthodes peuvent être distinguées :

1- Une première classe procédant à une découpe cartésienne du système : ces méthodes consistent à un découpage fonctionnel et hiérarchique du système telles que SA, SADT, etc.
2- Une deuxième classe relevant de l'approche systémique : cette classe de méthodes procède à deux modélisations ; l'une pour les données et l'autre pour les traitements telles que MERISE, AXIAL, etc.
3- Une troisième classe de méthodes orientées objet : elles procèdent à une structuration du système en termes d'objets (encapsulant données et traitements) et relations entre ces objets telles que Booch, OOSE, OMT, etc. UML est actuellement la méthode la plus complète et représentative de cette classe.

Quelques unes de ces méthodes, les plus significatives pour notre cas d'étude, sont ici présentées.

I.1- Approches fonctionnelles *(cartésiennes)*

Les approches fonctionnelles procèdent à un découpage fonctionnel du système permettant ainsi sa factorisation. Elles aboutissent à des logiciels modulaires et clairs ; néanmoins la tâche de maintenance est délicate précisément si on opte pour un cas d'évolution majeure qui pourrait toucher la globalité du système.

Nous avons choisi de présenter ici principalement les méthodes SA, SADT, SART et les graphes de fluence.

I.1.1- La méthode SA (Structured Analysis)

La méthode SA (Structured Analysis) a été développée par Demarco, à la base du modèle diagramme de flot de données (DFD) (figure 2.1). Son objectif est d'identifier et représenter les fonctions d'un système par niveaux de détails successifs et par niveaux d'abstraction successifs.

A tout niveau, chaque fonction peut être décomposée en sous-fonctions ou fonctions filles de niveau inférieur représentant le détail des traitements de la fonction mère. Les fonctions de SA sont appelées processus et ont pour but de transformer des données portées par des flots de données.

L'apport de la méthode SA est qu'elle représente les fonctions d'un système par niveaux de détails successifs et par niveaux d'abstraction successives. Sa limite est qu'elle n'exprime ni le contrôle ni le séquencement temporel des actions.

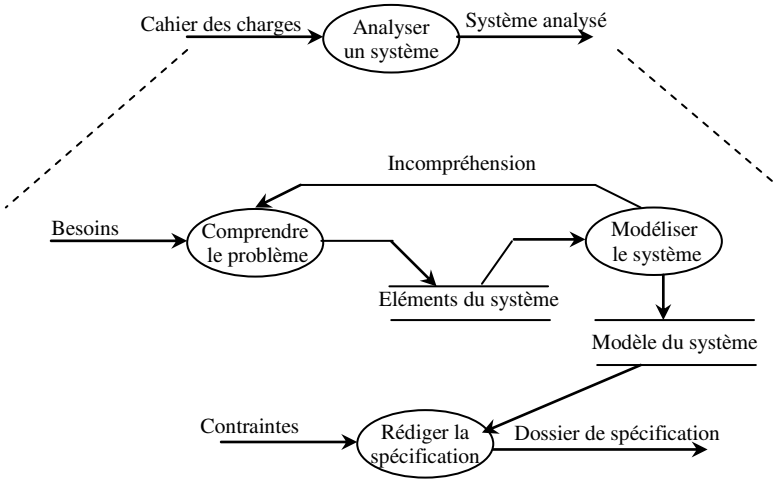


Figure 2.1 : Exemple de DFD pour une fonction analysée par SA

I.1.2- La méthode SADT (Structured Analysis and Design Technique)

La méthode SADT est une application du principe de la méthode "Diagramme Bloc" qui consiste à décrire un système complexe existant sur les deux plans : *le plan fonctionnel* (la mission du système, les différentes fonctions, leurs niveaux et leurs interactions) et *le plan structurel* (les frontières du système, les entrées-sorties, les principaux sous-systèmes et leurs liaisons) [IGL, 89]. L'analyse d'un système par la méthode SADT suit une démarche descendante, modulaire, hiérarchique et structurée (figure 2.2).

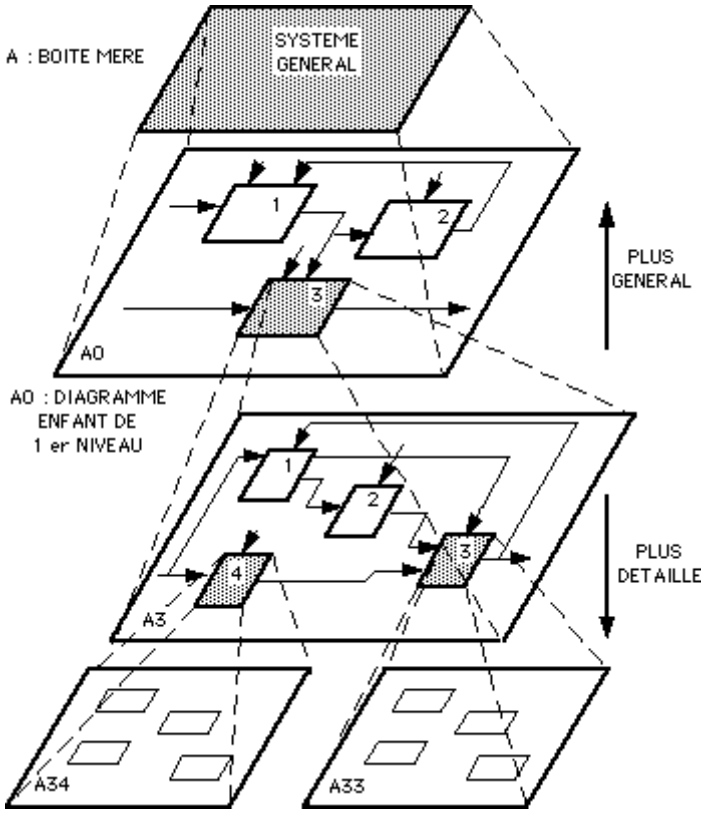


Figure 2.2 : Structure hiérarchique d'une analyse par SADT

SADT permet de formaliser clairement l'état fonctionnel du système et d'en faire ressortir les points critiques. Néanmoins, pour un système complexe mal décomposé, elle aboutit à des schémas difficiles à exploiter. Par ailleurs, cette méthode ne fournit pas d'éléments de réponse concernant l'aspect dynamique. De même, la formalisation des notions de contraintes et moyens de réalisation des fonctions reste assez limitée.

I.1.3- La méthode SART (Structured Analysis for Real Time Systems)

La méthode SART a été proposée par Ward et Millor en 85 [Ward et al., 85], puis étendue en 87 aux aspects architecture système par Hatley et Pirbhai. Elle se présente comme une extension sous l'angle du temps réel de la méthode fonctionnelle SA avec une bonne couverture de l'axe dynamique par la prise en compte des informations de contrôle (événements, exceptions, activations, synchronisation, etc.). La modélisation du comportement du système se fait dans un modèle dynamique séparé mais couplé au modèle fonctionnel.

Un troisième modèle est proposé par Hatley et Pirbhai : c'est le modèle d'architecture. Il est proposé dans le but de couvrir le troisième axe d'analyse à savoir l'axe de l'aspect structurel. Il est très proche du modèle de la décomposition SADT (actigrammes). Néanmoins, il reste moins convaincant et moins approfondi.

I.1.4- La méthode des graphes de fluence

Les graphes de fluence est une technique d'analyse fonctionnelle qui a été adaptée de la technique "analyse des schémas électriques", par Sinclair en 65 [Sinclair et al., 65]. Cette technique consiste à décrire le fonctionnement du système à partir des variables caractéristiques et de leurs interactions.

L'analyse par graphe de fluence consiste à établir, dans un premier lieu, un tableau déterminant les relations fonctionnelles entre les différentes variables du système. Puis, un diagramme causal sera déduit à partir de ce tableau (figure 2.3).

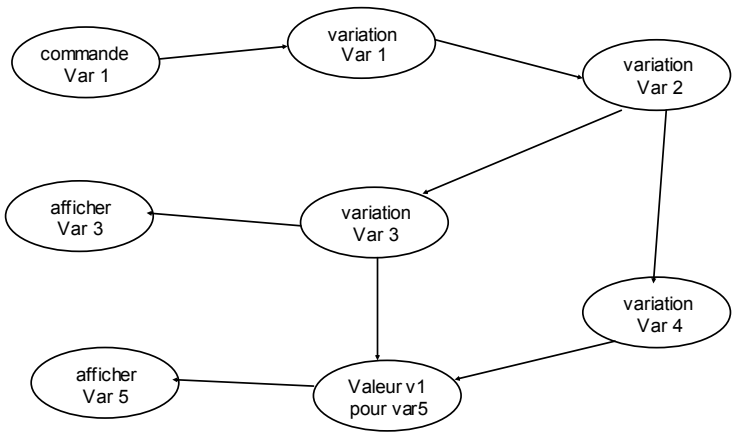


Figure 2.3 : exemple de graphe de fluence

Les graphes de fluence permettent, ainsi, de modéliser la représentation qu'a l'opérateur du système. Ce qui permettra d'identifier les interfaces homme-machine. Mais d'un autre coté, vu le nombre important de variables manipulées dans le cas des applications industrielles, cette méthode peut aboutir à des schémas très complexes, difficiles à lire et à exploiter.

I.2- Approches systémiques

L'approche systémique se base sur le concept qu'un système peut se décomposer en trois sous-systèmes [Tardieu et al., 00]:

- le sous-système opérant qui gère un flux d'événements et assure les fonctions (telles que : facturer clients, gérer stock,…) ;
- le sous-système de pilotage qui décide des actions à conduire en fonction des objectifs et des politiques de l'entreprise ; et
- le sous-système d'information qui s'intercale entre les deux premiers et dont l'objectif est de collecter, mémoriser, traiter et distribuer l'information.

Nous avons choisi de présenter ici la méthode la plus répandue de cette approche : MERISE (Méthode d'Etudes et de Réalisation Informatique des Systèmes Evolués).

La méthode MERISE permet de décrire l'ensemble du système d'information par un formalisme basé sur le concept de graphe structuré normalisé comprenant le traitement des données. La description est faite conformément à trois niveaux : le niveau conceptuel, le niveau logique et le niveau physique.

Au niveau conceptuel qui nous intéresse plus particulièrement, MERISE se base principalement sur le MCD (Modèle Conceptuel des Données) qui décrit le résultat de l'analyse conceptuelle des données (figure 2.4) et le MCT (Modèle Conceptuel des Traitements) qui décrit l'ensemble des traitements à apporter sur les données. Le MCD peut être transformé en un MPD (Modèle Physique des Données) c'est-à-dire un modèle directement exploitable par la base de données à utiliser.

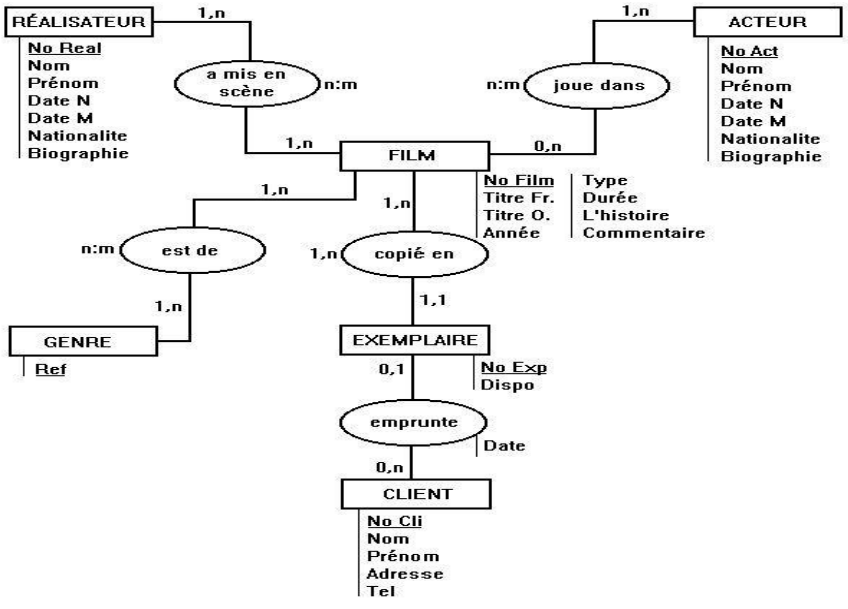


Figure 2.4 : exemple d'un MCD pour une agence de location de films vidéo

MERISE permet de supporter les développements des applications suivant les étapes successives du cycle de vie des systèmes d'information. Mais elle est surtout orientée vers les applications relevant du domaine de la gestion.

I.3- Approches objets

L'approche orientée objet essaye de couvrir à la fois les trois aspects d'analyse des systèmes de manière satisfaisante :

- l'aspect structurel : par la description des objets, de leurs propriétés et de leurs relations;
- l'aspect dynamique : par la description du comportement des objets, de leurs états et de leur réactivité; et
- l'aspect fonctionnel : par la description des fonctions et des traitements alloués aux objets et des conditions d'activation de ces traitements.

Le principe de la conception orientée objet est basé sur le concept de l'encapsulation des données ainsi que les procédures de manipulation de celles-ci.

L'approche objet permet une vérification et une validation des modèles par la simulation, ce qui garantit la faisabilité de l'analyse. Elle autorise plusieurs niveaux de réutilisation : les objets eux-mêmes, mais aussi des groupes d'objets cohérents, sous-systèmes, modules ou catégories de classes. Elle offre une facilité de transition vers la phase de conception, soit par enrichissement progressif des modèles, soit par application des règles de passage tenant compte de l'architecture de la réalisation retenue.

La critique adressée généralement à cette approche est qu'elle ne permet pas de montrer clairement les fonctions globales du système. De même, l'approche objets reste beaucoup moins intuitive que l'approche fonctionnelle et l'application des concepts objets nécessite une grande rigueur pour éviter les ambiguïtés.

Plus qu'une cinquantaine de méthodes orientées objets sont apparues durant les années 90 ; trois d'entre elles ont véritablement émergé : celle de Booch, OMT et OOSE [Rumbaugh et al., 91][Jacobson et al., 00]. Ces trois méthodes ont été à la base d'un projet d'unification qui a donné UML.

UML est donc né de la fusion des méthodes objet dominantes principalement (OMT, Booch et OOSE) et a été normalisé par l'OMG en 1997. Il est rapidement devenu un standard incontournable. Il donne une définition plus rigoureuse des concepts objet et apporte progressivement la dimension méthodologique qui faisait défaut à l'approche objet.

UML facilite la représentation et la compréhension de solutions objet :

- Sa notation graphique permet d'exprimer visuellement une solution objet, ce qui facilite la comparaison et l'évaluation de solutions.
- L'aspect semi-formel de sa notation, limite les ambiguïtés et les incompréhensions.
- Son indépendance par rapport aux langages de programmation, aux domaines d'application et aux processus, en font un langage universel.

Une autre caractéristique importante d'UML, est qu'il cadre l'analyse. UML permet de représenter un système selon différentes vues complémentaires : les diagrammes. Chaque type de diagramme possède une structure (les types des éléments de modélisation qui le composent sont prédéfinis) et véhicule une sémantique précise (il offre toujours la même vue d'un système). Combinés, les différents types de diagrammes UML offrent une vue complète des aspects statiques et dynamiques d'un système (figure 2.5).

UML favorise le prototypage, et c'est là une de ses forces. Il sous-entend une démarche d'analyse qui permet de concevoir une solution objet de manière itérative, grâce aux diagrammes qui supportent l'abstraction.

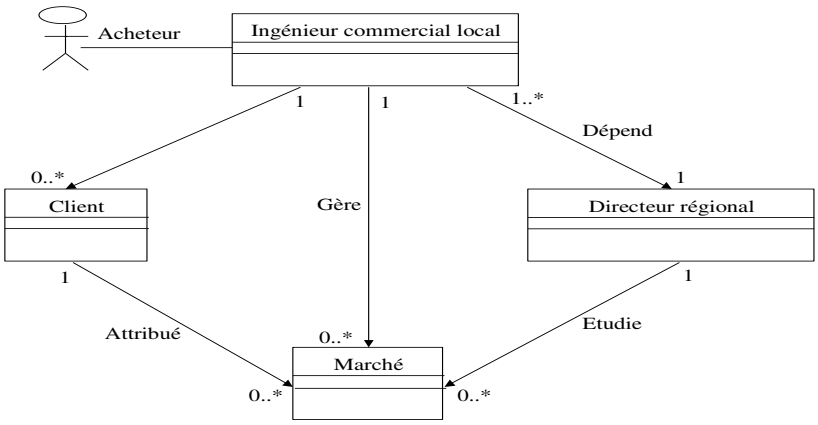


Figure 2.5 : Exemple de diagramme de classe des objets participant à une collaboration "ouvrir marché" modélisé par UML

Dans cette première partie, nous avons présenté certaines méthodes existantes utilisables pour l'analyse des systèmes en mode de fonctionnement normal. Une autre classe de méthodes existe. Elle regroupe les méthodes applicables en cas de nécessité de sûreté de fonctionnement des systèmes, pour l'analyse en mode de dysfonctionnement [Bergot et al., 95][Sahraoui, 96]. Cette classe de méthodes est maintenant présentée.

II. Méthodes et techniques d'analyse des systèmes en mode de fonctionnement anormal

Nous proposons ci-dessous un ensemble de méthodes et techniques d'analyse des systèmes en mode de fonctionnement anormal (voir aussi [Fadier, 90] et [Villemeur, 88] pour une revue quasi-exhaustive).

II.1- Méthode d'Analyse Préliminaire des Dangers (APD)

La méthode APD (Analyse Préliminaire des Dangers) a été utilisée la première fois vers les années 60 aux Etats Unis, dans le cadre de l'analyse de sécurité de missiles à propergols liquides [Ericson, 69]. Ensuite, elle a été formalisée par l'industrie aéronautique (Boeing), puis généralisée à de nombreuses industries : chimie,

nucléaire, aéronautique, etc. Il est généralement recommandé de l'appliquer dès les premières phases de conception du système à risques.

Son principe consiste à :
- Identifier les dangers d'une installation industrielle et ses causes (entités dangereuses, situations dangereuses, accidents potentiels). Cette identification se fait à l'aide de l'expérience et du jugement des ingénieurs et par l'utilisation des listes guides élaborées pour un domaine précis.
- Evaluer la gravité des conséquences (liées aux situations dangereuses et accidents potentiels). C'est une évaluation des risques (ce qui explique pourquoi la méthode est aussi appelée Analyse Préliminaire des Risques).
- Déduire tous les moyens, toutes les actions correctrices permettant d'éliminer ou de maîtriser les situations dangereuses et accidents potentiels mis en évidence.

II.2- Méthodes d'analyse et regroupement des pannes

II.2.1- La méthode AMDE (Analyse des Modes de Défaillances et de leurs Effets)

La méthode AMDE[2] fut employée à partir des années 60 dans le domaine de l'aéronautique pour l'analyse de la sécurité des avions [Recht, 66]. Par la suite, elle s'est généralisée dans de très nombreux domaines industriels : le domaine nucléaire aux Etats-Unis, le domaine aéronautique aux Etats-Unis et en France, le domaine spatial, le domaine chimique et celui de l'automobile [Suhner et al., 92].

C'est une méthode inductive. Elle permet de :
1. évaluer les effets de chaque mode de défaillance[3] des composants d'un système sur les différentes fonctions du système,
2. identifier les modes de défaillances ayant d'importants effets sur la disponibilité, la fiabilité, la maintenabilité ou la sécurité du système.

Il existe quatre étapes principales pour réaliser une AMDE :
1- Définir le système, ses fonctions et ses composants, et choisir les états de fonctionnement du système particulièrement importants en vue d'une analyse AMDE. Ce choix peut être effectué à l'aide d'une APD.
2- Etablir la liste des modes de défaillances et leurs causes pour les différents composants du système. Pour identifier les causes il faudra utiliser d'autres méthodes telles que : AdD ou MCPR (voir plus loin).
3- Etudier les effets des modes de défaillances établis. La considération des variables importantes du système et de leurs variations est essentielle au niveau de cette étape.
4- Fournir des conclusions et des recommandations.

L'AMDEC (Analyse des Modes de Défaillances, de leurs Effets et de leurs Criticité) est une extension de l'AMDE où l'on considère la probabilité d'occurrence de chaque mode de défaillance et la classe de gravité des effets à laquelle il appartient.

---

[2] *Une AMDE est réalisée dans un état donné de fonctionnement du système.*

[3] *Un mode de défaillance est l'effet par lequel une défaillance du comportement est observée.*

L'AMDE constitue une analyse préliminaire qui doit généralement être complétée par l'utilisation d'autres méthodes pour l'identification des combinaisons de défaillances pertinentes.

La principale critique faite pour cette approche est sa lourdeur qui vient du principe de son analyse faite pour tous les composants du système quelle que soit leur importance.

II.2.2- La méthode MCPR (Méthode de Combinaisons des Pannes Résumées)

Elle vient compléter l'AMDE, qui ne permet généralement d'étudier que les simples défaillances, par l'étude de combinaisons de ces défaillances [Villemeur, 81]. Elle peut être appliquée simultanément à plusieurs systèmes élémentaires.

Quatre étapes sont nécessaires pour son application :
1- Effectuer une AMDE pour chaque système élémentaire.
2- Elaborer les "Pannes Résumées Internes" (PRI) : regrouper dans un ensemble de pannes, les modes de défaillances dues à des causes internes au système et ayant les mêmes effets sur le système ou les autres systèmes élémentaires.
3- Elaborer les "Pannes Résumées Externes" (PRE) : les pannes résumées externes d'un système S1 est l'ensemble des pannes résumées internes (et/ou leurs combinaisons) relatives à d'autres systèmes externes pouvant affecter le fonctionnement du système S1 de la même manière.
4- Elaborer les "Pannes Résumées Globales" (PRG) : regrouper les PRI et les PRE et leurs combinaisons possibles ayant les mêmes effets sur le système élémentaire étudié et sur les autres systèmes élémentaires.

La MCPR est une extension et une généralisation de l'AMDE. Elle se prête bien à des analyses systématiques, pratiquées sur des ensembles de systèmes élémentaires complexes en interaction. La notion d'"effets" permet d'établir le lien entre les différentes pannes et les différents fonctionnements anormaux. En revanche, sa possible lourdeur invite à ne l'employer qu'à bon escient selon Villemeur [Villemeur, 88].

II.2.3- La méthode MTV (Méthode de la Table de Vérité)

Elle consiste à recenser toutes les combinaisons d'états (états de fonctionnement et états de pannes) des composants, les unes après les autres et à en étudier leurs effets [Villemeur, 88]. Les premières étapes sont analogues à celles de l'AMDE. Chaque composant sera caractérisé par un état de fonctionnement (0) et un état de panne (1). Ceci permet la construction des vecteurs des états des différents composants puis l'étude des effets de tous les vecteurs des états.

Son principe est fort simple mais il se révèle irréalisable pour l'analyse manuelle de grands systèmes, en raison d'un très grand nombre de combinaisons à considérer.

II.3- Méthodes d'analyse détaillée causes/effets d'une panne

II.3.1- La méthode MDS (Méthode de Diagrammes de Succès)

Le but essentiel de cette méthode est la représentation de la fonction du système par des diagrammes de blocs [Dhillon et al., 81]. Un bloc représente un composant, un sous-système ou une fonction. Par la suite, des calculs de fiabilité seront menés à partir de ce diagramme de succès. Elle ne considère aucun ordre de priorité dans le fonctionnement ou les défaillances.

Elle était dénommée au départ, Méthode du Diagramme de Fiabilité. Ses limites ont été rapidement apparues, au début des années 60. La recherche de voies nouvelles et plus fines d'analyse a conduit aux méthodes AMDE (voir § II.2.1) et AdD (voir ci-après).

II.3.2- La méthode AdD (Arbre des Défaillances)

Cette méthode est aussi appelée Méthode de l'Arbre des Causes (MAC). Elle est née en 61-62 dans les bureaux de la société de Bell-Telephone, développée par Watson pour évaluer et améliorer la fiabilité. La société Boeing et Hassl contribuèrent à développer et formaliser la méthode en 65 [Hass et al., 65][Lambert, 73][Lambert et al., 97]. Et depuis, l'emploi de la méthode est devenu fréquent dans de nombreux domaines industriels (aéronautique, nucléaire, chimique). C'est la méthode d'analyse de la fiabilité, de la disponibilité et de la sécurité des systèmes la plus largement utilisée.

Le but de cette méthode est de déterminer les diverses combinaisons possibles d'événements qui entraînent la réalisation d'un événement indésirable unique, puis la représentation graphique de ces combinaisons au moyen d'une structure arborescente.

La principale limite de cette méthode, réside dans le fait que l'événement indésirable à analyser doit être défini d'une manière précise. S'il est beaucoup trop général (ex : chute d'avion, fusion d'un cœur de réacteur nucléaire,…), l'analyse devient inextricable et impossible à réaliser. De même, si l'événement est trop spécifique, l'analyse risque de ne pas mettre en évidence les éléments importants du système. Il est recommandé de commencer par une APD ou une AMDE pour identifier les différents événements indésirables par rapport à différentes phases du système.

Cette méthode prend difficilement en compte les événements séquentiels, c'est-à-dire les événements dont la réalisation dépend de l'ordre d'apparition des différentes causes possibles. D'autres méthodes plus appropriées sont alors utilisées soit en remplacement, soit plus souvent en complément telles que : la méthode de l'Arbre des Conséquences, celle du diagramme Causes-Conséquences ou celle de l'Espace des Etats (présentées ci-après).

Il existe une correspondance entre Diagramme de Succès et Arbre des Défaillances. En effet, un diagramme série correspond à une "porte ET" et un diagramme parallèle correspond à une "porte OU". Mais MDS dérive d'une analyse fonctionnelle du système et l'AdD est déductive des causes des événements à partir de l'événement indésirable. Au fait, l'AdD a été élaborée pour dépasser les insuffisances de la MDS.

Un élément favorisant l'utilisation de l'AdD est qu'il existe de nombreux et performants programmes informatiques de calcul qui lui sont associés. Mais son emploi s'avère difficile voire impossible pour l'étude des systèmes élémentaires en interaction et fortement dépendants du temps. Elle demeure un outil complément à d'autres méthodes plus adaptées.

II.3.3- La méthode MACQ (Méthode de l'Arbre des Conséquences)

Elle fut employée la première fois entre 72 et 75 dans l'évaluation du risque lié aux centrales nucléaires aux Etats-Unis. Elle est très recommandée pour l'analyse des risques dans ce contexte [PRA, 82]. Dérivée de la méthode des arbres de décision, elle fut ensuite appelée "méthode des arbres d'événements" (Event Tree Method). C'est l'outil le plus fréquemment utilisé pour la caractérisation et la détermination des accidents potentiels. Elle permet de :

- identifier les différentes séquences d'événements possibles (acceptables et inacceptables),
- les représenter, et
- les évaluer de manière quantitative et qualitative.

Une séquence d'événements consiste en un premier événement appelé événement initiateur avec une succession ou une combinaison de défaillances. Si cette succession d'événements conduit à des conséquences jugées inacceptables, elle sera dénommée "séquence inacceptable", sinon, elle sera dénommée "séquence acceptable".

Cette méthode est généralement employée en liaison avec la méthode de l'arbre des défaillances. Elle peut être élaborée à partir d'une méthode inductive ou déductive. La démarche déductive la plus utilisée consiste en :

1. la définition des fonctions de sûreté (telles que : contrôle de réactivité, refroidissement du cœur,…),
2. la définition des événements initiateurs (par une évaluation technologique ou par construction d'un arbre des défaillances à partir des événements indésirables),
3. l'élaboration de l'arbre des conséquences « fonctions » (en identifiant les événements génériques de défaillances à partir des fonctions de sûreté),
4. l'élaboration de l'arbre des conséquences « systèmes » (en remplaçant dans l'arbre des conséquences « fonctions », les événements génériques (les fonctions) par les systèmes correspondants),
5. l'étude des événements indésirables au niveau des systèmes élémentaires et l'élaboration d'arbre de conséquences réduit.

La démarche inductive, quant à elle, conçoit trois étapes :
1. l'élaboration des PRG (pannes résumées globales) utilisant la MCPR,
2. la sélection des événements initiateurs (les définir parmi la liste des PRG qui entraînent une évolution des paramètres importants du système principal qui conduisent à un événement indésirable),
3. l'élaboration des arbres de conséquences, les événements considérés ici sont les PRG autres que les événements initiateurs (trois facteurs aident à la construction

de ces arbres de conséquences : le temps, les interactions fonctionnelles et les interactions entre systèmes).

II.3.4- la méthode MDCC (Méthode du diagramme Causes-Conséquences)

Elle a été initialement élaborée par le laboratoire RISO au Danemark, au début des années 70. Elle a été utilisée comme aide à l'analyse de fiabilité et de risque des centrales nucléaires dans les pays scandinaves [Nielsen, 71]. Elle combine les principes utilisés par la méthode déductive des arbres des défaillances (AdD) et celle inductive des arbres des conséquences (MACQ) [Hedin et al., 81]. Son champ d'application reste relativement limité.

Dans un diagramme Causes-Conséquences, on trouve deux parties :
- une partie 'Causes' qui représentent les causes d'un ou plusieurs événements "sommets" (ce sont des défaillances conduisant à des conséquences indésirables ou inacceptables)
- une partie 'Conséquences' qui est l'étendue des conséquences envisageables lorsque se réalisent les événements sommets.

Le principe d'élaboration du diagramme causes-conséquences consiste à :
1- faire la sélection d'un événement initiateur (APD, AMDE, MCPR),
2- rechercher les causes de l'événement initiateur (AdD),
3- rechercher les conséquences de l'événement initiateur (MACQ).

C'est une méthode intéressante pour l'analyse des systèmes où l'ordre dans lequel surviennent les défaillances est important. Cependant, elle apparaît difficile à utiliser pour l'analyse des systèmes trop complexes.

Une fois le système analysé, un document de spécification fonctionnelle peut être établi. Ce document précisera la structuration du système, l'ensemble de ses variables "pertinentes", le principe de son fonctionnement normal ainsi que ses différents dysfonctionnements possibles. Pour chacun de ces dysfonctionnements, un diagnostic sera élaboré précisant ses symptômes, ses effets et les procédures de recouvrement possibles.

L'étape qui suit immédiatement l'analyse d'un système consiste en sa modélisation préparant l'étape de sa réalisation. Les outils formels de modélisation présentent à cet effet des aides considérables. Ces outils sont maintenant présentés.

III. Outils formels de modélisation utilisés dans les systèmes homme-machine

III.1- Apports et limites des modèles formels

Les spécifications conventionnelles des systèmes sont généralement faites en langage naturel ou par des diagrammes. Elles sont donc difficiles à analyser et parfois même ambiguës. Vu que les erreurs ou omissions dans les systèmes informatiques coûtent très cher et s'avèrent parfois lourdes de conséquence, les spécifications formelles paraissent les plus prometteuses [Traore, 98].

En effet, seules ces techniques permettent aux concepteurs de décrire le comportement externe du système indépendamment des contraintes techniques et d'implémentation, et de mener par la suite des validations sur les spécifications avant de procéder à la réalisation proprement dite. Ces techniques sont spécialement recommandées et justifiées pour les systèmes suivants [Turner, 93] :

- les systèmes complexes : plusieurs systèmes existants appartiennent à cette catégorie ; notons que la tendance est de produire des systèmes de plus en plus complexes.
- les systèmes concurrents : on y distingue les systèmes distribués, les systèmes temps réel, et les systèmes issus de la programmation parallèle.
- les systèmes critiques pour la qualité : leur arrêt n'est pas dangereux mais leur efficacité et fiabilité sont très importantes ; citons les systèmes d'exploitation, les systèmes de télécommunication et les applications financières.
- les systèmes critiques pour la sûreté : on recense par exemple les systèmes de contrôle pour la défense, la médecine, l'industrie nucléaire, la télécommunication et le trafic aérien.
- les systèmes critiques pour la sécurité : les informations qu'ils regroupent sont confidentielles et nécessitent des préventions des utilisations illégales ; tels que les systèmes de la sécurité nationale.
- les systèmes standardisés : qui présentent une structure générale et qui sont largement utilisés.

Les techniques de spécification formelle permettent de garantir des spécifications concises (claires, non ambiguës), complètes, consistantes et extensibles. Elles permettent, ainsi, de découvrir les erreurs, les ambiguïtés et les inconsistances des idées tôt dans le cycle de développement. Ces techniques nécessitent, par la suite, une spécification exacte et claire des besoins et imposent une bonne structuration du domaine d'étude, ce qui facilite l'identification des problèmes possibles de sur- ou sous-spécification dans les phases préliminaires de développement des systèmes [Sahraoui et al., 91][Turner, 93].

Leur principale limite, en revanche, est qu'elles paraissent souvent d'accès délicat. Le premier obstacle auquel on se heurte est souvent une difficulté de lecture. En fait qu'il s'agisse de spécification ou de programme, concevoir un système judicieux de notation est une tâche non triviale, trop souvent négligée ou sous-estimée. Au delà des notations, les approches formelles ralentissent les phases initiales du développement (spécification, conception) au risque de produire une impression négative [Monin, 96].

Parmi les autres critiques formulées à ce sujet, on retrouve le fait qu'une méthode formelle se bâtit à partir de deux éléments principaux : un langage de spécification et un système de vérification. Ces deux composantes sont malheureusement très inégalement développées selon les approches. C'est ce qui rend la critique plus forte envers ces techniques.

Dans la littérature, il existe une multitude de techniques formelles de spécification basées sur différentes théories : l'algèbre universelle, la logique typée, la théorie des ensembles (Z, VDM, B) ou alors la théorie des automates (diagramme états-transition,

réseaux de Petri, etc.). Chacune de ces techniques a un ou plusieurs domaines de prédilection en terme d'application (traitement des données, applications temps réel, gestion des protocoles, etc.), une communauté de chercheurs et d'utilisateurs, et un certain nombre de variantes ou d'écoles. L'essentiel est de savoir choisir l'outil formel le plus adéquat vis-à-vis des besoins et de se placer au bon niveau d'abstraction pour bien spécifier, ni trop ni trop peu [Monin, 96].

En particulier, pour le domaine de l'interaction homme-machine, différents chercheurs ont proposé des méthodes formelles variées [Brun et al., 97][D'Ausbourg et al., 97][Paterno, 01][Gomes et al., 01]. Nous présentons ci-dessous, une étude bibliographique de quelques-unes de ces méthodes.

III.2- Outils formels proposés pour la spécification des IHM

III.2.1- La logique temporelle

Des modèles d'interaction abstraits basés sur la logique du premier ordre ont été proposés pour fournir une structure de spécification de l'Interface pour les systèmes complexes. Dans ces modèles, l'interaction est décrite par les quatre ensembles suivants [Johnson et al., 92] :

P : qui exprime l'ensemble des séquences d'entrée,
C : qui définit l'ensemble des commandes possibles,
S : qui définit l'ensemble des états possibles du système et
D : qui décrit l'ensemble des écrans possibles.

Quelques exemples de spécification utilisant cette logique sont donnés ci-dessous :

* *Exemple 1 :* $\forall p \in P,\ \forall s \in S,\ \exists! c \in C:$ *interpret (p, s, c)*
Ce qui veut exprimer que pour toute entrée *p*, il existe une unique commande *c*, telle que l'interprétation de *p* dans l'état *s* est la commande *c*.

* *Exemple 2 :* $\forall c \in C,\ \forall s \in S,\ \exists! s' \in S:$ *effect (c, s, s')*
Cette phrase veut dire que l'effet de toutes les commandes *c* est de transformer l'état du système vers l'unique état *s'*.

* *Exemple 3 :* $\forall s \in S,\ \exists d \in D:$ *view (s, d)*
Cela veut dire que tous les états *s* peuvent être visualisés à travers un écran *d*.

A l'aide de cette logique abstraite et déclarative, le concepteur n'aura pas besoin de spécifier les périphériques à utiliser dès le départ. Les détails d'entrées, de commandes et de comportement pourront être ajoutés au fur et à mesure que la conception progresse. Mais la principale limite de ce formalisme est qu'il ignore l'information temporelle et donc le séquencement des commandes ne peut pas être modélisé. Comme solution à ce problème, Johnson et Harrison ont proposé la logique temporelle qui est une extension de la puissance déclarative statique de la logique du premier ordre. Le but de cette logique était d'introduire l'information temporelle dans la spécification des systèmes interactifs. Il a été alors nécessaire de concevoir de nouveaux opérateurs tels que [Johnson et al., 92] :

✧ : pour exprimer l'éventualité,
O : pour exprimer la notion de suivant,
□ : pour exprimer la notion de toujours,
U : pour exprimer la notion jusqu'à (Until).

Bien que ces extensions ont réussi à dépasser quelques limites de la logique du premier ordre, elles ne sont pas arrivées à résoudre complètement tous les problèmes de la spécification. On peut remarquer par exemple que la propriété d'éventualité n'est pas toujours appropriée. Elle peut risquer des erreurs graves et causer par la suite l'échec du système. D'autant plus que cette logique temporelle manque encore de moyens pour intégrer les facteurs humains et incorporer la notion du temps réel.

ACTL (Action Based Temporal Logic) [De Nicola, 91] est une extension de la logique temporelle permettant de décrire les propriétés spécifiques d'une application interactive [Duce et Paterno, 93].

III.2.2- Les langages ensemblistes

Les représentants les plus connus de cette famille de langages sont VDM, apparu dès les années 60, et Z apparu avec les années 70. Une autre variante de ces langages a été proposée par la suite, avec la méthode B.

Ces techniques se différencient suivant la manière d'utiliser les notations ensemblistes, la logique employée ou la facilité avec laquelle elles permettent de passer de la spécification à la programmation.

Avec Z, la notion d'ensemble est utilisée comme moyen d'expression universel, ce qui a l'avantage de l'uniformité : l'espace des états du système modélisé est un ensemble, les types sont des ensembles, même les opérations sont des ensembles. Elles sont modélisées par des relations, c'est-à-dire des sous-ensembles du carré cartésien de l'ensemble des états.

Le système de notations possède des symboles pour les différentes sortes de relation (fonctions, injections, etc.) et différents opérateurs permettant de construire des relations.

Les différentes structures et opérations sont déclarées au moyen de tables appelées schémas. Un schéma comporte deux parties :

- une partie déclarative définissant les variables (un peu dans le style du langage Pascal), à partir des types construits au moyen d'ensembles prédéfinis et des opérateurs ensemblistes usuels (union, produit cartésien,…), et
- une partie prédicat précisant les contraintes sur les valeurs au moyen d'assertions logiques.

La méthode B est née avec l'idée d'un processus conduisant d'une spécification au code exécutable. Elle utilise la théorie des ensembles dans le même esprit que Z et en reprend les notations. Elle diffère pour les mécanismes de structuration et la manière de spécifier les changements d'états. Elle regroupe :

- un langage de spécification (les machines abstraites),

- une technique de raffinage et de codage,
- des obligations de preuve associée à chaque étape,
- des mécanismes de structuration visant à décomposer les machines abstraites,
- des outils permettant d'assister et de contrôler les tâches.

Le langage Z a été proposé pour la modélisation de l'interaction homme-machine dès les débuts des années 90 [Harrison et al., 90]. Récemment, la méthode B a été adoptée pour la spécification des IHM. Cette première expérience est présentée dans les travaux de Ait-Ameur et al. (voir chapitre 1).

III.2.3- Les automates d'état fini

Les automates d'état offrent l'avantage de visualiser les enchaînements d'actions autorisées. Les automates sont spécifiés statiquement par l'ensemble des états d'interaction et dynamiquement par un ensemble d'arcs étiquetés par des jetons d'entrée représentant les actions de l'utilisateur, caractérisés par un nom et un ensemble de données (figure 2.6).

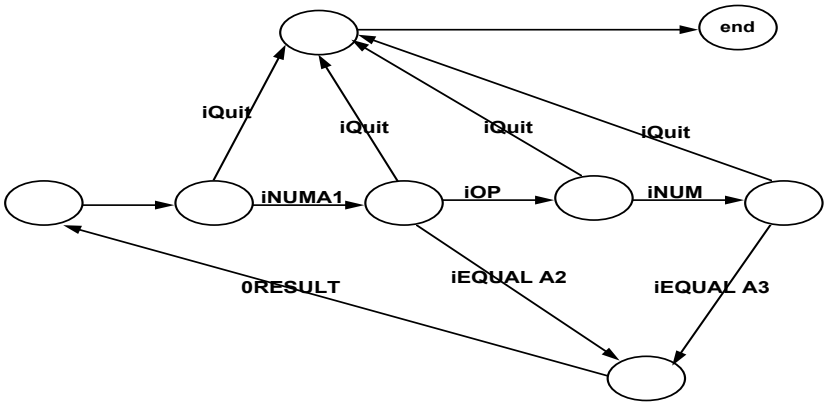


Figure 2.6 : Automate à états finis décrivant une opération de calcul dans une calculatrice

A l'aide de cette spécification, on devient capable de décrire les actions du système et leurs séquences vis-à-vis des entrées [Harel et al., 87][Vallee, 92]. Mais en contrepartie, l'automate devient complexe et difficile à comprendre à cause de sa taille qui grandit rapidement en fonction des états possibles. Plusieurs extensions ont été proposées :

* *Les RTN* (Recursive Transition Network) : les arcs y sont étiquetés par des appels récursifs à d'autres automates afin de limiter le problème majeur de la taille des diagrammes [Jacob, 85].

* *Les ATN* (Augmented Transition Network) : proposés par Woods, ils introduisent des extensions au schéma de base du diagramme de transition récursif en associant des variables simples, des conditions et des expressions de sorties textuelles aux noeuds et/ou arcs de l'automate.

Les diagrammes états-transitions ont été proposés dès les années 60 pour la modélisation de l'enchaînement des écrans graphiques [Parnas, 69]. Jacob a proposé par la suite l'utilisation de ces diagrammes pour les interfaces à manipulation directe

[Jacob, 85]. Actuellement, ils font l'objet d'extensions dédiées à la modélisation de la navigation dans les applications web (StateWebCharts) [Winckler et Palanque, 03].

III.2.4- Le langage Lotos

Lotos constitue l'une des techniques de description formelle basée sur le concept des systèmes à états et transitions pour la spécification du comportement. Cette technique permet la modélisation du comportement séquentiel, du choix et du non déterminisme d'une manière non ambiguë. Elle assure la modélisation de communication synchrone et asynchrone. C'est une technique proposée pour des types de données abstraits (ADT : Abstract Data Types) [Turner, 93].

Avec Lotos, on définit quatre parties :
- la syntaxe,
- la sémantique statique,
- la sémantique algébrique des types de données, et
- la sémantique dynamique du comportement.

Lotos se base sur trois concepts de base :
- les processus (le système et ses composants) : un processus est représenté par une boite noire avec des points de connexion pour représenter ses attachements physiques ou logiques avec le reste du système,
- les événements, et
- les arbres pour exprimer le comportement (figure 2.7)

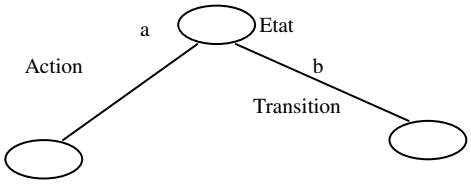


Figure 2.7 : Arbre en Lotos

Cette technique a servi de base pour la description formelle du concept d'interacteur dans la spécification des IHM. Nous pouvons citer à ce sujet principalement les travaux réalisés à Pise : FullLotos [Paterno et Faconti., 92] et TLIM (Task Lotos Interactors modelling Model) [Paterno et Mancini, 99].

III.2.5- Le langage Lustre

Le langage Lustre est un langage synchrone fondé sur le modèle flot de données. Un programme peut être considéré comme un réseau d'opérateurs, et l'hypothèse de synchronisme consiste à supposer que chaque opérateur de ce réseau répond instantanément à ses entrées [Halbwachs et al., 91]. L'idée de base est d'exprimer, avec Lustre, les propriétés requises d'un programme, profitant de sa nature déclarative et d'utiliser, par la suite, l'automate produit par le compilateur pour vérifier ses propriétés. Lustre s'appuie sur trois concepts fondamentaux :
- le flot de données : c'est un couple formé d'une suite, éventuellement infinie, de valeurs d'un type donné, et d'une horloge représentant une suite d'instants.
- Un opérateur de retard «pre»
- Un opérateur d'initialisation (->)

Un programme lustre est composé d'un système d'équations et d'assertions. Les assertions sont une généralisation des équations. Elles consistent en des expressions booléennes qui sont supposées toujours vraies, pour optimiser le code.
Exemple : assert not(x and y) : pour exprimer le fait que deux événements en entrée représentés par les flots booléens x et y, ne surviennent jamais simultanément.

Ce langage a été proposé avec les travaux de D'Ausbourg et al. [D'Ausbourg et al., 96] [D'Ausbourg et al., 97], pour la vérification automatique des propriétés des IHM.

III.2.6- Les Réseaux de Petri

Les réseaux de Petri (RdP) [4] ont été proposés par Carl Adam Petri en 1962. Ils représentent un formalisme mathématiquement fondé dédié à la modélisation du parallélisme et du synchronisme dans les systèmes discrets, prenant en considération les aspects événementiels qui ne peuvent pas être traités par les automates [Petri, 62] [David et al., 92] [Berthomieu et al., 01] [Girault et Valk, 02].

Un RdP est un graphe biparti fondé sur deux types de noeuds : " ○ " pour représenter les places et " —" pour représenter les transitions avec un ensemble d'arcs reliants les différentes places aux différents transitions (figure 2.8). Il est défini par un quadruplet R = <P, T, Pré, Post > avec :

- **P** : l'ensemble des places du réseau.
- **T** : l'ensemble des transitions du réseau.
- **Pré : P x T → |N**, une fonction d'incidence avant, définissant le nombre de jetons nécessaires dans une place Pi entrée d'une transition Tj permettant la franchissabilité de cette transition.
- **Post : P x T → |N**, une fonction d'incidence arrière, définissant le nombre de jetons à déposer dans une place Pi sortie d'une transition Tj après son franchissement.

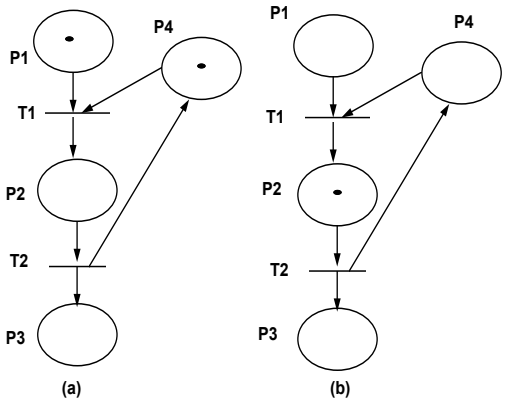


Figure 2.8 : Exemple de l'état d'un RdP avant et après le franchissement d'une transition

Les RdP sont en expansion continue et représentent un outil adéquat pour la modélisation de l'Interface Homme-Machine. Au début, ils n'ont été utilisés qu'à des fins de description des tâches à informatiser. Mais par la suite, et surtout avec l'apparition des Réseaux de Petri de Haut Niveau, on a commencé à profiter de leur puissance pour modéliser le dialogue Homme-Machine. Nous citons, principalement,

[4] *Voir sites :* http://www.ec-lille.fr/~rdp/, et http://www.daimi.au.dk/PetriNets/

les travaux de Willem et Biljon qui ont été les premiers à avoir pensé à adapter les RdP's à la modélisation du dialogue Homme-Machine [Willem et al., 88]; ceux de Palanque et al. avec le formalisme ICO; ceux de Tabary et al. avec TOOD (faisant suite aux travaux de [Abed, 90][Abed et Ezzedine, 98]) et ceux de De Rosis et al. avec XDM (voir chapitre 1).

IV. Synthèse et choix des méthodes et outils

Nos travaux de recherche visent à élaborer une démarche de conception et de spécification des IHM pour le contrôle des procédés industriels. Les premières étapes de la démarche consistent à analyser et modéliser le système homme-machine dans l'objectif suivant :

1- Décomposer le système hiérarchiquement selon les besoins de surveillance afin de mettre en évidence les sous-systèmes élémentaires faciles à analyser et à modéliser.

2- Identifier les différents états de fonctionnement de chaque sous-système et déduire les variables pertinentes et significatives pour chacun de ces états.

3- Analyser la tâche opérateur pour chacun des états de fonctionnement et déduire l'ensemble des actions appropriées.

4- Modéliser l'interaction homme-machine en fonction des différents états de fonctionnements du système et des actions opérateur, afin de déduire par la suite, l'ensemble des BIO nécessaires pour la spécification des interfaces de supervision.

La problématique est donc de savoir choisir les meilleures méthodes d'analyse et outils de modélisation permettant de satisfaire ces besoins. Pour cela, nous avons établi un certain nombre de critères d'évaluation des différentes méthodes et outils existants, permettant de sélectionner les plus adéquats pour notre cas d'étude. Trois groupes de critères sont définis :

- Le premier groupe est proposé pour l'évaluation des méthodes d'analyse des systèmes, il comprend :

- le degré du recouvrement de l'aspect structurel de l'analyse,
- le degré du recouvrement de l'aspect fonctionnel de l'analyse,
- le degré du recouvrement de l'aspect dynamique de l'analyse,
- le taux d'utilisation de la méthode dans la pratique,
- la capacité du découpage hiérarchique du système,
- le degré de visibilité de la circulation du flux d'information,

- Le deuxième groupe de critères est élaboré pour évaluer les outils formels de modélisation, il englobe :

- la puissance de l'outil en terme de vérifiabilité des propriétés,
- le degré de la prise en considération de l'aspect événementiel,
- le degré de la prise en considération de l'aspect synchronisme,
- le degré de la prise en considération de l'aspect parallélisme,
- l'aptitude à l'implémentation,
- l'orientation de l'outil vers la conception des nouveaux systèmes,
- l'orientation de l'outil vers la vérification des propriétés des systèmes existants.

Tableau 2.1 : Evaluation informelle des méthodes d'analyse et des outils de modélisation des systèmes

	Méthodes d'analyse des systèmes						Outils formels de modélisation					
Méthodes et Outils Critères d'évaluation	SA	SADT	SART	Graphe de fluence	MERISE	UML	Logique temporelle	Langages ensemblistes	Automates à états finis	Lotos	Lustre	RdP
Recouvrement de l'aspect structurel	0	2	1	0	1	1						
Recouvrement de l'aspect fonctionnel	2	2	2	2	2	2						
Recouvrement de l'aspect dynamique	0	0	2	1	1	2						
Taux d'utilisation en pratique	1	2	2	1	2	2						
Découpage hiérarchique du sys.	1	2	1	0	1	0						
Visibilité de la circulation du flux d'info.	2	2	2	1	2	1						
Simplicité	2	2	1	2	2	1	1	0	1	0	0	1
Aspect graphique	2	2	2	2	2	2	0	0	2	1	0	2
Domaine d'application industrielle	2	2	2	2	0	1	0	1	1	2	1	2
Outil informatique	2	2	2	0	2	2	0	1	1	2	0	2
Aspect vérifiabilité							1	2	1	2	2	2
Aspect événementiel							0	1	2	1	1	2
Aspect synchronisme							0	0	1	2	2	2
Aspect parallélisme							0	0	2	2	1	2
Aptitude à l'implémentation							1	1	2	2	2	2
Orientation conception							1	2	2	2	0	2
Orientation vérification							1	2	1	2	2	2
TOTAL	**14**	**18**	**17**	**11**	**15**	**14**	**5**	**10**	**16**	**18**	**11**	**21**

- Le dernier groupe consiste en un ensemble de critères généraux appliqués à la fois pour évaluer aussi bien les méthodes d'analyse que les outils de modélisation, il regroupe :

- le degré de simplicité impliquant la facilité d'apprentissage de la méthode ou de l'outil,
- l'aspect graphique de la méthode ou de l'outil de modélisation,
- le degré d'applicabilité pour le domaine concerné par notre étude à savoir les applications industrielles,
- l'existence d'un outil informatique supportant la méthode ou le langage de modélisation.

Le tableau 2.1 récapitule les résultats de l'évaluation des méthodes et outils selon cet ensemble de critères. Nous avons choisi d'assigner pour chaque critère une note entre 0 et 2 montrant le degré de satisfiabilité de la méthode ou de l'outil à ce critère :

- 0 : ne satisfait pas le critère
- 1 : satisfait moyennement le critère
- 2 : satisfait bien le critère

Selon nos objectifs d'analyse, cette évaluation informelle montre que la méthode SADT est la plus adéquate. En effet, la méthode SADT, bien que développée en fin des années 70, reste parmi les méthodes d'analyse fonctionnelle les plus utilisées pour les systèmes industriels. Elle est particulièrement adaptée à l'analyse et la conception des systèmes. Elle est proposée, ici, spécialement pour avoir une vue structurée sur le système global, qui est généralement complexe. Cette décomposition SADT nous permettra par la suite d'étudier les sous-systèmes considérés comme « élémentaires », selon notre approche de décomposition et y appliquer les méthodes d'analyse de dysfonctionnements. Notons que d'autres auteurs l'utilisent à cet effet, tels [Abed, 01] et [Ezzedine, 02].

Pour l'analyse des systèmes en cas de dysfonctionnement, un schéma récapitulatif des principales méthodes développées pour la sûreté de fonctionnement des systèmes industriels précisant les différents liens possibles existants entre les différentes méthodes est proposé (figure 2.9). D'après ce schéma, une solution pertinente serait donc d'utiliser une combinaison des méthodes complémentaires suivantes :

1- commencer par la méthode APD pour identifier l'ensemble des états de fonctionnement du système et les dangers possibles (étape qui peut être ignorée) ;
2- choisir un état de fonctionnement et appliquer la méthode AMDEC pour identifier les différents modes de défaillances ;
3- analyser plus finement ces défaillances par la méthode AdD ;
4- analyser les événements conséquences possibles des défaillances par la MACQ ;
5- ayant l'ensemble des pannes et leurs effets, appliquer, éventuellement, la méthode MCPR pour déduire un ensemble de pannes regroupées selon leurs effets ;

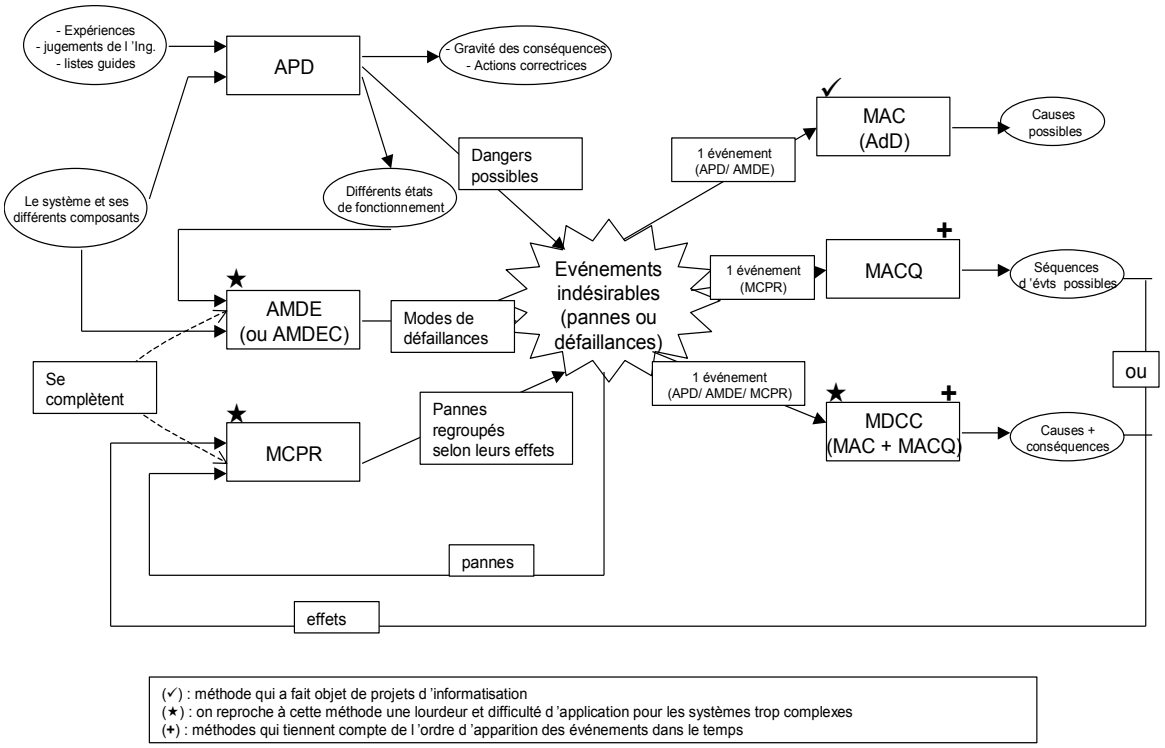


Figure 2.9 : Schéma récapitulatif des principales méthodes développées pour la sûreté de fonctionnement des systèmes industriels.

Rappelons ici que notre objectif de mener une analyse des dysfonctionnements des systèmes, doit déboucher principalement sur le fait de pouvoir dégager l'ensemble des variables pertinentes et significatives pour chaque cas de fonctionnement du système ainsi que l'ensemble des causes possibles d'un dysfonctionnement donné, afin de pouvoir aider l'opérateur à localiser la cause de la panne et à rétablir une situation de fonctionnement normal.

L'AMDEC, en effet, permet d'établir la liste des dysfonctionnements possibles d'un système donné et l'AdD la complète en analysant finement ces défaillances et précisant leurs causes possibles. Une combinaison de ces méthodes paraît donc la plus adéquate pour notre cas d'étude. Nous l'exploiterons dans le chapitre suivant.

Du moment que l'interface à spécifier concerne les procédés industriels (donc complexes et critiques pour la sûreté), le choix d'une technique de spécification formelle pour ce type d'interface est donc totalement justifié.

Parmi les outils proposés pour cette fin et présentés ici, le tableau d'évaluation montre bien que les Réseaux de Petri sont les plus adéquats pour la modélisation de l'interaction homme-machine. En effet, les RdP offrent un grand pouvoir d'expression des aspects événementiels, du synchronisme et du parallélisme, autant de critères pertinents pour les contraintes de spécification des interfaces graphiques interactives.

Ils ont d'ailleurs toujours été efficaces dans le domaine de l'automatique et ils deviennent de plus en plus utilisés dans celui de la conception d'interfaces homme-machine [Abed, 01].

Conclusion

Nous avons présenté au niveau de ce chapitre les différentes méthodes applicables pour l'analyse des SHM, phase nécessaire pour toute approche de conception des IHM. Deux classes de méthodes existent à cet effet, les méthodes d'analyse en mode de fonctionnement normal et les méthodes d'analyse en mode de dysfonctionnement. Une présentation succincte de ces méthodes a fait l'objet de la première partie du chapitre. Nous avons exposé, par la suite, une étude bibliographique des différents outils formels de modélisation et quelques expériences menées dans le domaine des spécifications des IHM.

Une synthèse de ces outils et modèles a été présentée à la fin du chapitre précisant les choix effectués dans le cadre de nos travaux de recherche. Nous passons maintenant à la présentation détaillée de l'approche adoptée.

Chapitre 3

Approche proposée pour la spécification, la vérification et la génération semi-automatique d'IHM dans les systèmes industriels complexes

Plan du chapitre

Introduction
I. Présentation générale de l'approche proposée
II. Présentation détaillée de l'approche
II.1- Analyse du SHM
II.2- Modélisation de l'interaction homme-machine
II.3- Déduction des BIO
II.4- Spécification de l'interface
II.5- Vérification des propriétés de l'interface
II.6- Génération de l'interface
II.7- Intégration et évaluation de l'IHM intégrée
Conclusion

L'objectif de ce travail de recherche est l'élaboration d'une approche globale de conception des IHM qui s'appuie sur l'utilisation d'outils formels pour couvrir de façon progressive et cohérente l'ensemble des étapes, depuis l'analyse du SHM et la modélisation de l'interaction homme-machine jusqu'à la génération semi-automatique des différentes vues graphiques.

Nous commençons par rappeler ici le squelette méthodologique de l'approche préconisée répondant au cahier des charges de l'approche globale de conception des IHM, élaboré à la fin du chapitre 1. Nous présentons, par la suite, les détails des différentes étapes de cette approche, développant l'ensemble des méthodes et outils choisis pour supporter chacune d'elle.

I. Présentation générale de l'approche proposée

L'approche proposée identifie sept étapes (figure 3.1) [Riahi, 97][Riahi et al., 98a][Riahi et al., 98b].

1. Une première étape d'analyse préliminaire du processus et de son système de commande est nécessaire. Cette analyse fournit un document rassemblant les données du procédé et ses différentes contraintes techniques et fonctionnelles.
2. La seconde étape consiste à analyser le SHM en termes de processus, son système de commande et les tâches opérateur. Différentes techniques et méthodes sont proposées à cet effet.
3. La troisième étape consiste en la modélisation du comportement de l'opérateur. Cette modélisation doit exprimer l'interaction entre l'opérateur et l'interface.
4. La quatrième étape assure l'identification des BIO en termes d'objets de l'interface.
5. Une fois les objets de l'interface déduits, cette étape consiste à spécifier l'interface en termes de vues et objets graphiques.
6. Nous tirons bénéfice de la technique formelle utilisée dans la spécification de l'interface pour vérifier les spécifications générées, au niveau de la 6ème étape.
7. La dernière étape de cette approche est dédiée à la génération automatique de l'interface.

La première étape est réalisée avec le responsable de l'installation industrielle. C'est une étape en amont de l'approche. L'étape 2, quant à elle, consiste à conduire une analyse du procédé en vue de spécifier le cahier des charges de l'IHM. Cette étape est réalisée en coopération avec le concepteur de l'installation industrielle et de l'opérateur humain, futur utilisateur de l'interface en salle de contrôle.

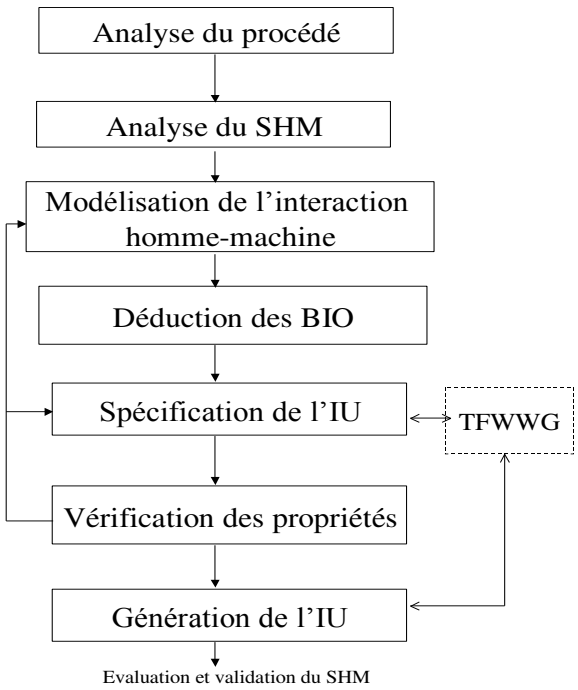


Figure 3.1 : Squelette méthodologique de l'approche proposée

L'étape suivante consiste à identifier et décrire les tâches opérateur en précisant l'enchaînement des actions pour l'accomplissement de ces tâches et en déduisant des

différentes analyses menées auparavant l'ensemble des Besoins Informationnels des Opérateurs (BIO), informations pertinentes pour l'exécution de chaque action.

Une modélisation formelle de l'interaction homme-machine, moyennant les réseaux de Petri Interprétés, permet, par la suite, de situer en chaque point d'interaction, l'ensemble des BIO nécessaires. La spécification de l'interface proprement dite peut alors commencer. Elle consiste à déduire pour l'ensemble des BIO identifiés, les objets graphiques correspondants pour l'affichage.

La prise en compte des critères ergonomiques de présentation sera assurée, à ce niveau avec l'intégration d'un outil à base de connaissances, du type TFWWG (Tools For Working With Guidelines). Ce type d'outil permettra de décider du choix des objets adéquats de présentation et de la qualité ergonomique des différentes vues graphiques à générer (voir chapitre1, §I.3).

L'utilisation des RdP pour la modélisation vise à préparer le terrain à des vérifications théoriques formelles et une validation *a priori* des interfaces au niveau de l'étape 6 ; ce qui permet en principe de gagner un temps considérable dans le cycle de développement des interfaces. Cette approche est considérée comme étant une approche basée sur un modèle (Model based Approach [Szekely, 96], voir chapitre 1) ; ses différentes étapes sont expliquées, détaillées et illustrées dans ce qui suit.

II. Présentation détaillée de l'approche

II.1- Analyse du SHM

L'analyse du SHM consiste à identifier les sous-systèmes significatifs et importants à contrôler et à analyser leurs comportements. L'objectif de cette analyse est d'identifier les différentes vues graphiques appropriées pour le contrôle et la commande de ces sous-systèmes. Le découpage du système à contrôler est donc fortement influencé par les vues graphiques qui vont le supporter. L'objectif est de fournir à tout instant t, une vue graphique par sous-système, reflétant son état de fonctionnement et regroupant l'ensemble des besoins informationnels de l'opérateur à cet instant là.

Au cas où un sous-système après étude nécessite, pour faciliter son contrôle et sa supervision, plus d'une vue graphique, ce sous-système sera dupliqué en deux ou plusieurs sous-systèmes supportés chacun par une vue graphique propre à lui. A titre d'exemple, si l'étude poussée d'un sous-système donné montre que dans un contexte particulier de fonctionnement, deux vues graphiques simultanées sont nécessaires pour mener à bien son contrôle/commande, ce sous-système sera dupliqué en deux sous-systèmes (figure 3.2).

Ainsi, trois étapes se révèlent nécessaires pour cette analyse.

Etape 1 : Procéder à une décomposition hiérarchique du SHM

Les systèmes à contrôler, dans notre contexte de travail, sont des systèmes industriels complexes. L'étude de tels systèmes constitue une tâche indispensable pour toute approche basée sur un modèle. Nous préconisons donc de commencer l'analyse par une décomposition hiérarchique du SHM identifiant des sous-systèmes élémentaires plus claires, faciles à étudier et à modéliser. La méthode bien connue SADT est ici proposée.

L'analyse d'un système par la méthode SADT suit, en effet, une démarche descendante, modulaire, hiérarchique et structurée. Elle permet de modéliser les sous-systèmes élémentaires appropriés pour l'étude et précise l'ensemble des variables pertinentes d'entrée et de sorties de chaque sous-système ainsi que l'ensemble des contraintes et moyens adéquats.

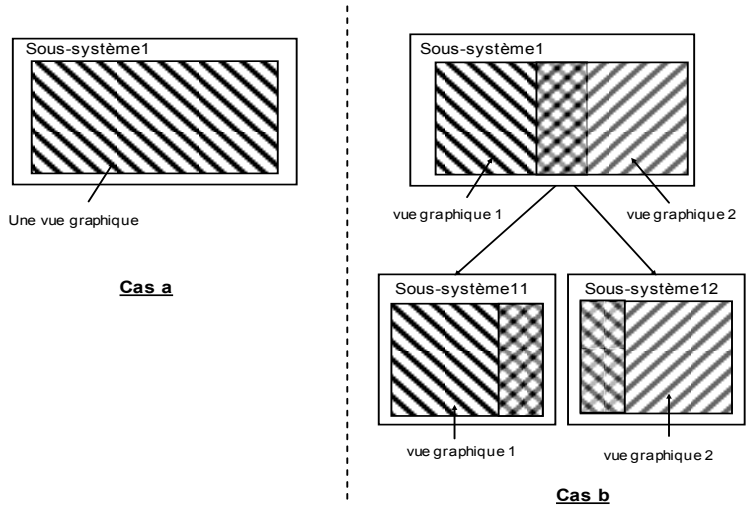


Figure 3.2 cas (a) d'un sous-système sans nécessité de duplication, et (b) d'un sous-système dupliqué

Etape 2 : Analyser les dysfonctionnements de chaque sous-système

Une fois les sous-systèmes élémentaires identifiés, l'analyse des dysfonctionnements devient possible. Pour cette fin, nous proposons l'utilisation d'une combinaison des méthodes AMDE et AdD (voir chapitre 2, §II.2). Rappelons que la méthode inductive AMDE consiste à étudier pour chaque sous-système l'ensemble de ses composants matériels et recenser pour chacun de ces composants : (1) les différents modes de défaillances envisageables, (2) les causes possibles, (3) les effets sur le sous-système en question et (4) les effets sur l'ensemble des autres sous-systèmes. Le résultat de cette analyse est résumé dans un tableau présentant l'ensemble des évènements indésirables de chaque sous-système.

La méthode déductive AdD permet, par la suite, de mener une analyse plus fine de chacun de ces événements indésirables, le but étant de déterminer les diverses combinaisons possibles d'événements qui entraînent l'occurrence de chacun de ces événements indésirables. Le résultat de cette analyse sera représenté graphiquement au moyen d'une structure arborescente.

Etape 3 : Identifier pour chaque sous-système et pour chaque contexte de fonctionnement, les tâches nécessaires de l'opérateur

L'analyse des différents contextes de fonctionnement du système étudié étant faite, l'étape suivante consiste à identifier pour chacun des contextes préalablement identifiés (contexte de fonctionnement normal et contextes de fonctionnement anormaux) l'ensemble des tâches opérateur nécessaires pour le suivi et le contrôle.

Soulignons à ce propos que l'analyse des tâches humaines nécessite un haut niveau de connaissance et de savoir faire et une collaboration étroite et pluridisciplinaire entre les différents intervenants du processus de développement des applications interactives. Plusieurs méthodes d'analyse et de modélisation de la tâche sont proposées dans la littérature (voir chapitre 1, §I.2). Nous supposons à cet effet, que l'analyse de la tâche est accomplie en adoptant une méthode appropriée parmi celles-ci.

Le résultat de cette analyse permet de définir les actions relatives aux tâches de l'opérateur. Les paramètres de ces actions constitueront avec l'ensemble des variables d'entrée/sorties et les variables de contrôle (identifiées au moyen de SADT, AMDE et AdD) l'ensemble des besoins informationnels de l'opérateur (BIO).

Ainsi, cette analyse du SHM avait pour principal objectif d'identifier les différents contextes de fonctionnement du système ainsi que les interventions de l'opérateur humain relatives à ces situations. L'étape qui suit consiste à modéliser le processus de l'interaction homme-machine à travers cet ensemble de vues. Les réseaux de Petri sont proposés à cet effet.

II.2- Modélisation de l'interaction homme-machine

L'objectif, ici, est de proposer une approche de modélisation de l'interaction homme-machine lors de l'accomplissement de l'opérateur de ses tâches de supervision et de commande [Riahi et al. 00][Moussa et al., 02]. Cette approche doit fournir un modèle réseau de Petri vérifiable sur lequel on pourra valider certaines propriétés importantes de l'interface. Le principe de construction du modèle, proposé ici, garantit ce postulat. De plus, afin d'être facilement approuvée et adoptée par les concepteurs, cette approche devrait être pédagogique et facile à utiliser. Ceci pourrait être, en effet, assuré en favorisant la simplicité et la caractéristique graphique de la modélisation.

Nous avons, donc, choisi la technique formelle des réseaux de Petri pour la modélisation du comportement de l'opérateur. En effet, on considère dans la littérature que cette technique est assez efficace pour l'expression des aspects de synchronisme, de concurrence et de parallélisme. Les RdP proposés ici sont les RdPI (Réseaux de Petri interprétés) [Moalla, 85]. Ce type de réseaux introduit les notions d'événements et de conditions ainsi que la notion d'actions. En effet, une condition de passage (Cj), un événement de déclenchement (Evj) et une action éventuelle (Aj) sont associés à chaque transition Tj d'un RdPI (figure 3.3).

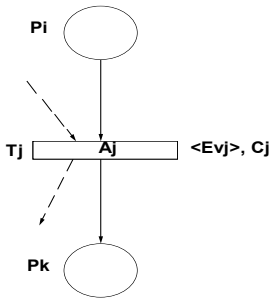


Figure 3.3 : Réseau de Petri Interprété (RdPI)

Pour modéliser l'interaction homme-machine moyennant les RdPI, nous avons convenu d'utiliser les places pour représenter *l'état du comportement* de l'opérateur vis-à-vis de l'évolution du système [Khelil, 01].

Nous nous référons dans ce sens, à la théorie décisionnelle de l'opérateur humain dans la résolution d'un problème. Cette théorie stipule que dans sa résolution d'un problème, l'opérateur humain passe par quatre phases, comme exposé par Rasmussen [Rasmussen, 86] : (i) la détection, (ii) l'évaluation de la situation, (iii) la décision et (vi) l'action. Nous modélisons les évolutions entre ces différents états par les transitions (voir chapitre1, §I.1.2).

Notons ici que chaque changement au niveau de l'état du système doit absolument affecter l'interface de contrôle afin d'informer l'opérateur de la situation et lui faciliter sa tâche de commande. Cela implique un changement au niveau des vues graphiques affectant les paramètres des objets graphiques (couleurs, formes, etc.) voire même un changement au niveau du contenu de la vue (apparition et/ou disparition de certains objets, etc.).

Cette technique de modélisation nous permet donc de déduire les BIO en fonction des changements de contexte de fonctionnement du système industriel.

Nous considérons qu'une tâche opérateur est composée d'un ensemble organisé d'actions élémentaires. La structure modélisant une action élémentaire de l'opérateur (par exemple une action de régulation de température) est donnée ci-dessous (figure 3.4).

La validation de la condition i (transition T1) modélise le fait que l'opérateur va commencer l'exécution de l'action relative à cette condition. L'apparition, par la suite, de l'événement "*fin action*" (transition T2) exprime le fait que l'action de l'opérateur a été exécutée et a pris fin. La place P2 exprime un état d'attente de la fin de l'exécution de l'action, alors que les places P1 et P3 modélisent l'état de l'opérateur avant et après l'exécution de son action. Par exemple, P1 modélise l'intention mentale de l'opérateur pour agir. La place P3 exprime son état à la fin de l'accomplissement de l'action.

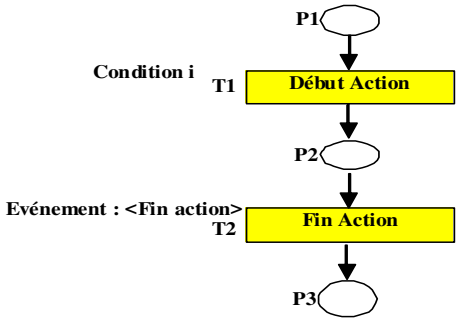


Figure 3.4 : Structure modélisant l'état d'un opérateur face à une action élémentaire

Toutes les actions de l'opérateur (élémentaires ou non) sont ordonnées selon des compositions typiques : séquentielle, parallèle, alternative, de choix, itérative ou de fermeture [Khelil, 01]. Nous présentons ci-dessous le principe de chacune de ces compositions avant de donner la structure globale du modèle de l'interaction homme-machine ainsi construit.

II.2.1- Composition séquentielle

La composition séquentielle de n actions traduit le séquencement de leur exécution.

Règle de composition :
La composition séquentielle de n actions est assurée en fusionnant la place *fin* du réseau modélisant l'action i, avec la place *début* de l'action i+1 (figure 3.5).

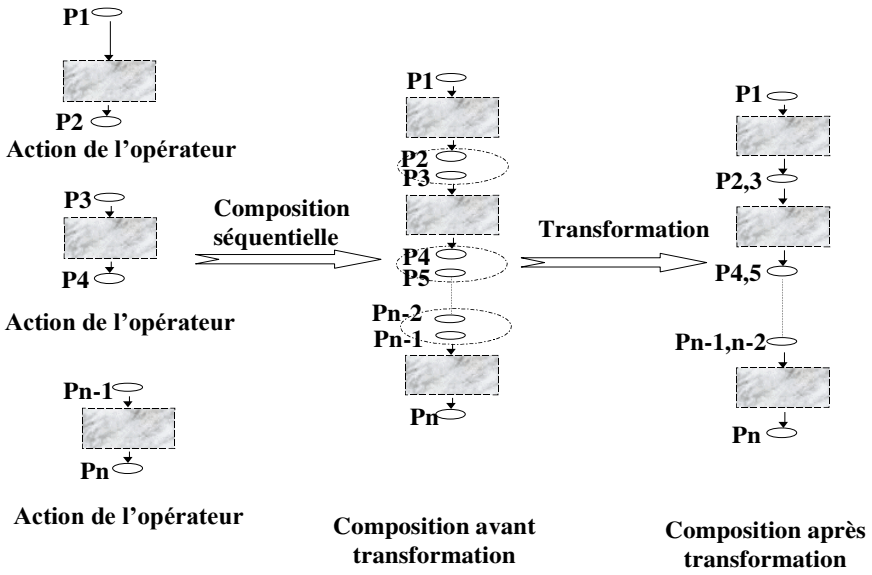


Figure 3.5 : Composition séquentielle

Précaution à prendre :
Dans le contexte considéré de l'exécution de la tâche de l'opérateur humain, le séquencement des actions décrites doit être possible.

Comme exemple de composition séquentielle d'actions de l'opérateur, nous pouvons imaginer une situation pour laquelle l'analyse de la tâche a révélé nécessaire que l'opérateur humain accomplisse trois actions l'une à la suite de l'autre : commencer par fermer un robinet, puis commander l'ouverture d'une première vanne de sécurité V1 et commander toute de suite après une deuxième vanne de sécurité V2. La modélisation d'une telle interaction sera modélisée par une composition séquentielle de trois structures élémentaires (modélisant les trois actions élémentaires) comme précisé dans la figure 3.5.

II.2.2- Composition parallèle

La composition parallèle exprime la possibilité de leur exécution simultanée. Le parallélisme est assuré par le fait qu'une place de synchronisation d'entrée, active en même temps toutes les places d'initialisation des actions parallèles à exécuter. Cependant, le parallélisme effectif ne peut s'effectuer que si les actions à exécuter ne sollicitent pas les mêmes ressources. Sinon, un séquencement partiel ou total serait nécessaire.

Règle de composition :
La composition parallèle de n réseaux relatifs à n actions fait intervenir deux structures de composition PAR1 et PAR2.

- La structure PAR1 modélise une synchronisation de départ de n réseaux (figure 3.6).
- La structure PAR2 modélise une synchronisation à l'arrivée des n réseaux (figure 3.7).

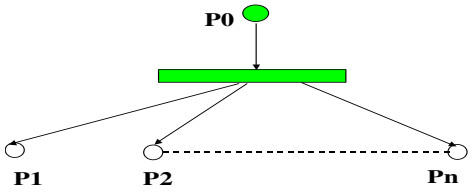


Figure 3.6 : Structure PAR1

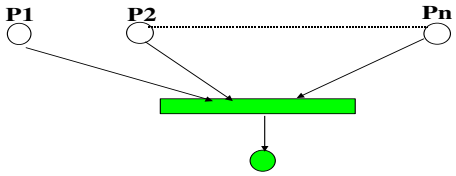


Figure 3.7 : Structure PAR2

Evidemment, le nombre de places Pn dans les deux structures PAR1 et PAR2 devrait être égal au nombre des actions parallèles Ai. Ainsi, pour assurer la composition parallèle des actions, il est nécessaire de synchroniser les places d'entrée et celles de sortie de ces actions (figure 3.8).

Précaution à prendre :
Dans le cas où une ou plusieurs ressources peuvent être sollicitées par l'exécution des branches (actions) parallèles, une analyse fine doit juger de l'opportunité de la parallélisation. A signaler que, d'une façon générale, la gestion des ressources partagées s'effectue d'une façon tout à fait indépendante des tâches.

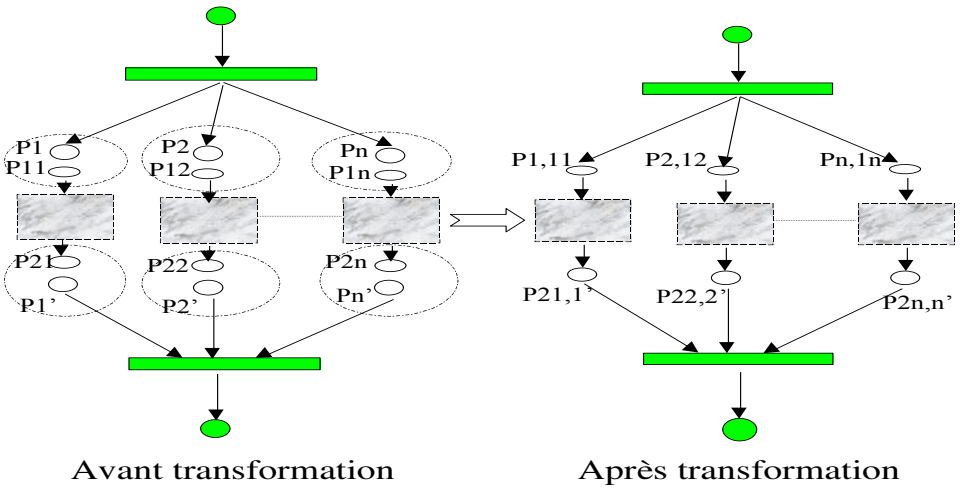


Figure 3.8 : Composition parallèle

Considérons, par exemple, une situation dans laquelle il est demandé à l'opérateur d'ajuster les températures de deux produits en parallèle. La modélisation de cette interaction pourrait

être assurée par la composition en parallèle de deux branches relatives aux actions de régulation des températures.

II.2.3- Composition alternative

La composition alternative de n actions traduit une exécution toujours exclusive de ces actions. Afin d'éviter un conflit effectif, des conditions sont associées aux transitions pour déterminer sans ambiguïté laquelle des actions doit être exécutée.

Règle de composition :
La composition alternative de n réseaux est réalisée en les composant en séquentiel avec une structure ALT et en fusionnant toutes les places *fin* de ces réseaux. La structure ALT permet la validation d'une seule condition à la fois (figure 3.9).

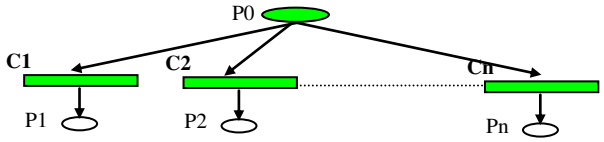
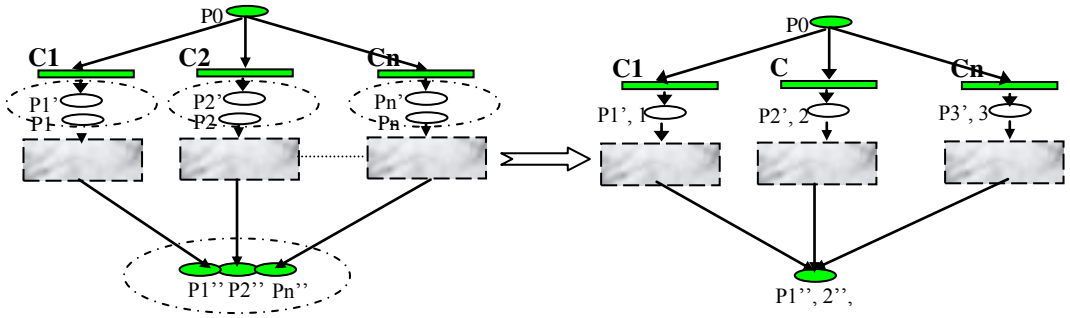


Figure 3.9 : Structure ALT

La structure ALT comporte un ensemble de transitions égal au nombre des réseaux à composer en alternatif. Ces transitions sont issues d'une même place d'entrée P0. Elles permettent, grâce aux conditions qui leur sont associées, d'initialiser sans ambiguïté un réseau unique parmi les n modélisés, ce qui garantit l'absence de conflit effectif (figure 3.10).

Les conditions Ci, i variant de 1 à n, dépendent de l'état courant interne au système.



(a) Avant transformations

(b) Après transformations

Figure 3.10 : Composition Alternative

Précaution à prendre :
Dans le contexte considéré de l'exécution de la tâche, on doit pouvoir faire correspondre à chaque condition Ci au moins un état éventuel du système qui la valide. Le dit contexte doit permettre l'exécution effective de l'action.

Comme illustration simple de cet aspect, nous pouvons considérer le cas d'un sous-système pour lequel l'analyse a mis en évidence deux situations de dysfonctionnement D1 et D2 qui ne puissent jamais survenir en même temps. Pour chacune de ces situations, une suite d'actions de l'opérateur est nécessaire. La modélisation d'une telle interaction se fait par une

composition alternative de deux réseaux modélisant chacun les actions relatives à une situation de dysfonctionnement. Les conditions associées C1 et C2 relatives à la structure ALT de composition vont exprimer respectivement l'occurrence des dysfonctionnements D1 et D2.

II.2.4- Composition choix opérateur

Il y a deux compositions choix opérateur possibles : le choix exclusif et le choix inclusif.

- *Composition choix exclusif :* Pour la composition choix exclusif, elle rappelle la composition alternative, à la différence que la décision de l'action à exécuter n'est pas déterminée par l'état courant interne du système mais par un choix de l'opérateur : l'opérateur intervient à travers l'exécution d'une action élémentaire pour décider laquelle des actions parmi les n est à exécuter.

  Règle de composition :
  La composition de n réseaux permettant un choix exclusif par l'opérateur (figure 3.12) peut être faite en composant en séquentiel une structure élémentaire de décision du choix opérateur (figure 3.11) avec la structure obtenue par composition alternative des n réseaux.

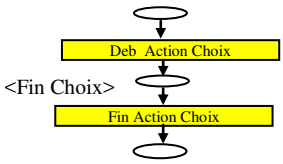


Figure 3.11 : structure élémentaire de décision du choix opérateur

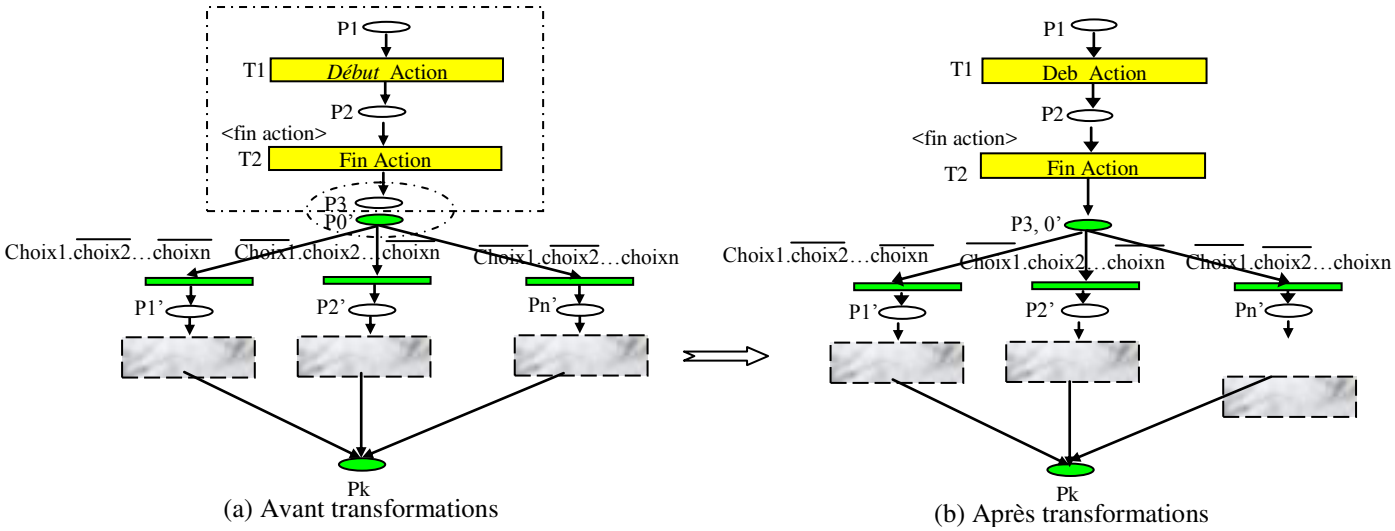


Figure 3.12 : Composition de choix opérateur exclusif

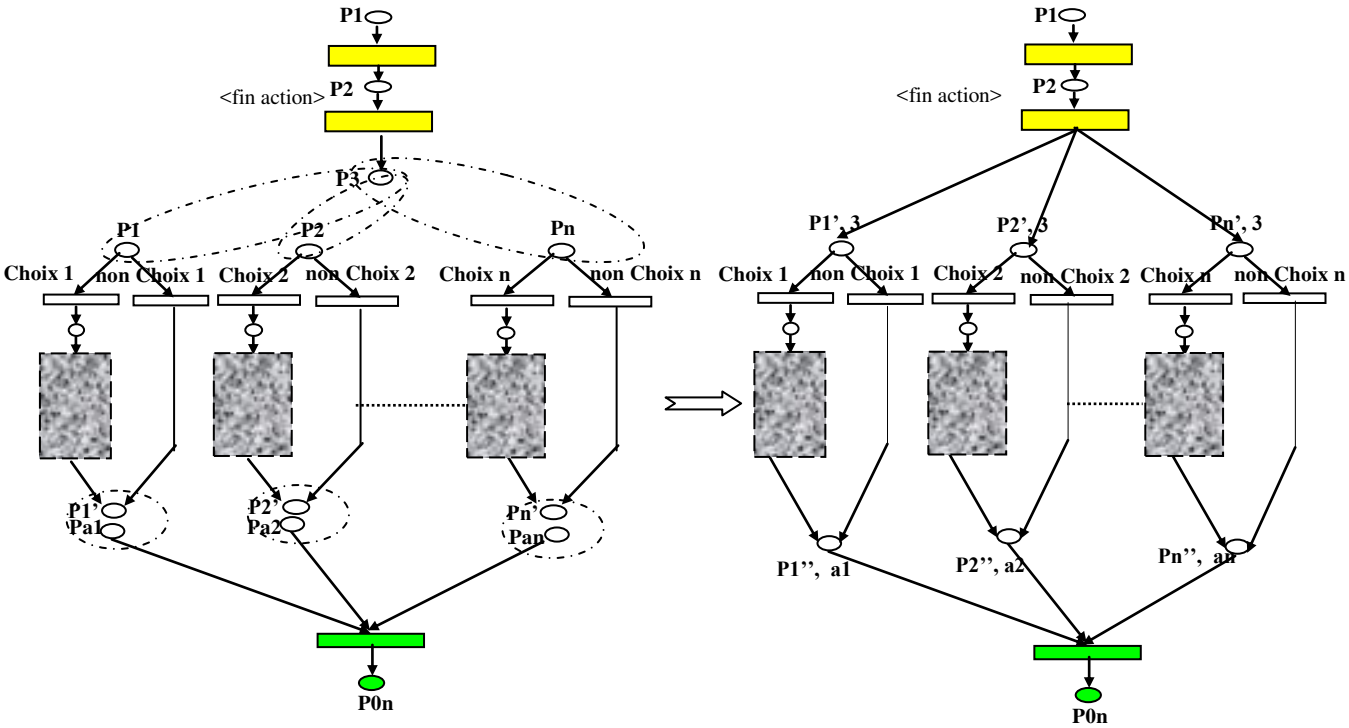
Précautions à prendre :
L'action de choix doit déterminer une et une seule évolution possible ce qui garantit l'absence de conflit effectif. Le contexte de ce choix doit permettre l'exécution effective de l'action. De plus, à chaque expression de choix, on doit pouvoir faire correspondre au moins un état éventuel du système qui la valide.

Comme cas simple d'illustration, nous pouvons imaginer une situation pour laquelle l'opérateur se trouve contraint de choisir exclusivement entre deux actions possibles : ouvrir un interrupteur I1 pour commander un certain circuit électronique ou alors ouvrir un deuxième interrupteur I2 permettant de commander un autre circuit.

- *Composition choix inclusif :* A la différence de la composition choix exclusif pour laquelle l'opérateur décide du choix d'une seule action à exécuter parmi n, dans la composition choix inclusif, l'opérateur peut décider du choix d'un sous-ensemble d'actions à exécuter parmi les n (0 ou n actions). Ainsi, pour un choix inclusif, l'opérateur intervient à travers l'exécution d'une action élémentaire pour décider lesquelles parmi les n actions sont à exécuter, les autres seront « court-circuitées ».

Règle de composition :
La composition de n réseaux permettant un choix inclusif par l'opérateur peut être faite (figure 3.13) :

a. en incluant chacun des n réseaux dans une structure alternative,
b. en composant en parallèle les structures alternatives obtenues,
c. en composant en séquentiel une structure élémentaire de décision des choix opérateur avec le réseau des structures alternatives obtenue en b,
d. en opérant la simplification.



a. Avant transformations

b. Après transformations

Figure 3.13 : Composition Choix opérateur inclusif

Précautions à prendre :
L'action de choix doit déterminer une ou plusieurs évolutions possibles et le contexte de ce choix doit permettre l'exécution effective des actions correspondantes. De plus,

à chaque expression de choix, on doit pouvoir faire correspondre au moins un état éventuel du système qui le valide.

Supposons, par exemple, que l'opérateur se retrouve dans une situation pour laquelle il devrait choisir d'ajuster les températures d'un ou de plusieurs produits en entrée pour réguler la température d'une solution mélange. Cette situation peut être modélisée par une composition choix inclusif de plusieurs réseaux modélisant chacun l'action d'ajustement de l'un de ces produits au moyen de son organe de commande respectif.

II.2.5- Composition itérative

La composition itérative de n actions exprime leur exécution successive avec itération possible. L'itération est soumise à une condition « Itération » calculée au cours de l'exécution séquentielle des n actions.

Règle de composition :
La composition itérative d'un réseau est réalisée en englobant le réseau dans une structure **I** d'itération (figure 3.15). La structure d'itération comporte deux transitions Ti et Tni. Ces transitions sont issues d'une même place d'entrée. Elles permettent, grâce aux conditions qui leur sont associées, d'être franchies sans ambiguïté ce qui garantit l'absence de conflit effectif (figure 3.14).

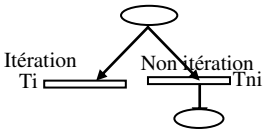
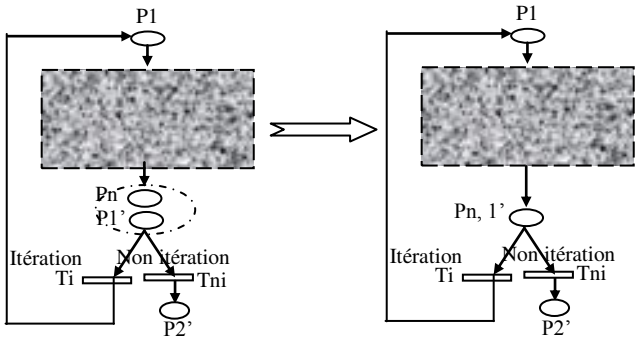


Figure 3.14 : Réseau I



(a) Avant transformations (b) Après transformations

Figure 3.15 : Composition itérative

Précautions à prendre :
Le contexte qui valide la condition d'entrée ou la condition d'itération doit permettre l'exécution effective de l'action globale du réseau. De plus, un tel contexte doit être possible.

Comme illustration simple de cet aspect, nous pouvons imaginer une situation de dysfonctionnement pour laquelle l'opérateur devrait agir sur un produit en entrée en augmentant son débit petit à petit par des doses de 5 unités. Cette tâche de correction sera

modélisée par une composition itérative d'une action élémentaire relative à l'ajout de 5 unités à chaque fois au débit du produit en entrée.

II.2.6- Composition de fermeture

La composition de fermeture d'un réseau traduit la mise en boucle de ce réseau.

Règle de composition :
La composition de fermeture d'un réseau est réalisée en englobant le réseau dans une structure F de fermeture (figures 3.16 et 3.17).

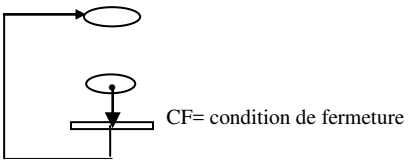
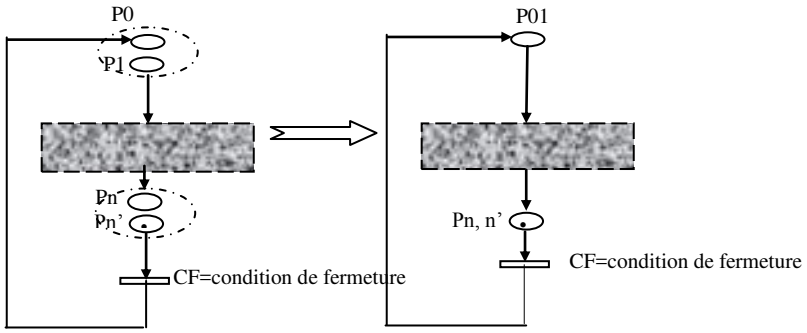


Figure 3.16 : Structure de fermeture

Précautions à prendre :
La condition de fermeture peut être utilisée comme déclencheur de la tâche globale. Tout contexte qui la valide doit, en principe, garantir l'évolution jusqu'au bout de l'exécution de la tâche.



(a) Avant composition (b) Après composition

Figure 3.17 : Composition de fermeture

La composition de fermeture sera utilisée pour construire le modèle global de l'interaction homme-machine et d'exprimer la boucle de surveillance et de contrôle permettant de revenir à la situation initiale de surveillance d'un système donnée.

II.2.7- Principe de construction du modèle global de l'interaction homme-machine

La construction du modèle global d'une tâche utilisateur s'effectue à partir des structures de base modélisant les différentes actions élémentaires de l'opérateur. Elle se base sur l'application des différentes opérations de composition explicitées ci-dessus et son principe réside dans le fait que la construction du modèle de la tâche n'utilise que des structures et des

règles de composition définies. Cela est important pour assurer par avance de bonnes propriétés au modèle obtenu.

Remarquons ici, que normalement, toute tâche doit être « réinitialisable » pour que l'on puisse la réexécuter de nouveau. La construction du modèle de la tâche doit donc se terminer obligatoirement par une composition de fermeture.

Par ailleurs, l'application des règles de composition peut conduire à des réseaux dans lesquels apparaissent des simplifications évidentes qui peuvent être effectuées pour réduire la taille du modèle tout en conservant ses bonnes propriétés. Ces simplifications concernent, en fait, des états instables. C'est le cas notamment de (1) la composition séquentielle de tâches élémentaires non soumises à des conditions explicites de déclenchement ou alors (2) la dernière étape de la composition choix opérateur inclusif.

Illustration :
Dans la figure 3.18, nous illustrons le processus de modélisation de l'interaction homme-machine moyennant les RdPI et se basant sur ce principe de composition. Nous supposons, dans cet exemple, que la tâche de l'opérateur revient à accomplir une première action A1, qui consiste pour ce cas à ouvrir une vanne de sécurité, en séquence avec deux autres actions en parallèle A2 et A3, permettant de régler le débit d'une solution mélange en agissant simultanément sur les débits d1 et d2 des deux produits en entrée.

Chacune de ces actions est modélisée par la structure élémentaire présentée ci-dessus (figure 3.4), et le modèle global est construit par une composition de ces trois structures élémentaires. Les places P2, P5 et P8 modélisent des états d'attente de l'exécution des actions de l'opérateur, respectivement A1, A2 et A3.

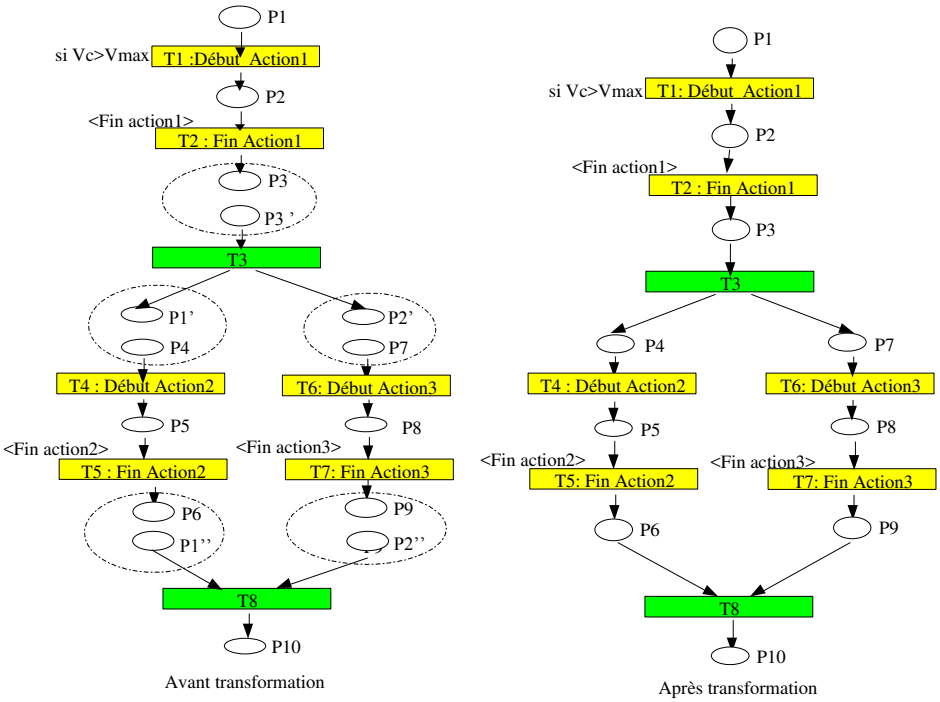


Figure 3.18 : Exemple d'un modèle d'interaction homme-machine

Le RdPI proposé pour la modélisation de l'interaction homme-machine est alors défini par l'ensemble : < P, T, E, OB, Pre, Post, μ, Precond, Action, Info-Transition > avec :

- P : ensemble des places = {P1, P2,..., Pn},
- T : ensemble des transitions = {T1, T2,..., Tm},
- E : ensemble des événements externes incluant l'événement "toujours présent" <e>,
- OB : ensemble des objets graphiques de l'interface,
- Pre : P x T $\rightarrow$ N définit les poids des arcs reliant les places pi de P aux transitions tk de T,
- Post : P x T $\rightarrow$ N définit les poids des arcs reliant les transitions tk de T aux places pi de P,
- μ : T $\rightarrow$ E associe à chaque transition l'événement de déclenchement approprié,
- Precond : T $\rightarrow$ Expression Booléenne définit la condition nécessaire de passage de chaque transition,
- Action : T $\rightarrow$ A définit l'éventuelle procédure d'action appropriée associée à chaque transition,
- Info-Transition : T $\rightarrow$ OB associe à chaque transition l'ensemble des objets graphiques appropriés. Cette association est expliquée ci-dessous.

Une fois le comportement de l'opérateur modélisé, les BIO peuvent être déduits.

II.3- Déduction des Besoins Informationnels des Opérateurs (BIO)

L'opérateur a besoin d'identifier dans la mesure du possible instantanément l'état de fonctionnement du procédé qu'il contrôle. Cette information lui est transmise à travers différents objets au niveau de l'interface (messages, valeurs, graphiques, alarmes, etc.). Ces objets sont, en fait, reliés à des variables d'état du système. Il suffit donc d'identifier l'ensemble des variables d'information appropriées à chaque état du système. Ces variables d'information découlent, en fait, de l'analyse du SHM menée auparavant (voir §II.1).

De plus, et dans le but d'accomplir ses tâches, l'opérateur aura à intervenir et commander quelques variables lors de l'apparition des dysfonctionnements. L'interface doit donc présenter dans de telles situations, l'ensemble des objets de commande qui permettent à l'opérateur d'exécuter ses actions et commander le processus.

L'ensemble de ces variables d'information et de commande constitue les BIO. Nous expliquons ici la manière de déduire ces BIO à travers les précédentes analyses et le modèle de l'interaction homme-machine établi au niveau de l'étape précédente de la démarche.

Nous associons aux transitions "*début action*" les variables d'information et de commande reflétant l'état du système et respectant les besoins des opérateurs en matière d'information (ses BIO) (figure 3.19).

Par exemple, si nous considérons l'action A1 du modèle présenté dans la figure 3.18, cette action consiste à vider un silo de stockage d'un produit C. Au niveau de l'état modélisé par la place P2, l'opérateur doit disposer des BIO nécessaires pour bien accomplir sa tâche. La variable d'information VC informe l'opérateur du niveau courant du produit C dans le silo de stockage et la variable de commande CVC lui permet d'agir sur ce niveau à travers la vanne de sécurité. Ces deux variables constituent donc les BIO relatifs à cet état.

Pour les actions A2 et A3 qui modélisent des actions de l'opérateur sur les débits des produits en entrée A et B, l'opérateur aura besoin de connaître pour chacun de ces produits le volume courant (VC) du mélange ainsi que le débit courant (DA respectivement DB). Pour agir, il a

besoin d'une variable de commande (CDA respectivement CDB). Cet ensemble de variables constitue les BIO relatifs à cette situation.

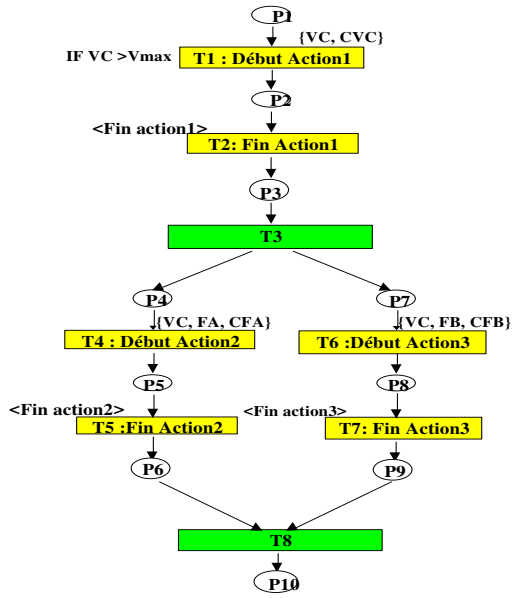


Figure 3.19 : Exemple d'un modèle de l'interaction homme-machine avec les BIO

Une fois les BIO identifiés (c'est-à-dire l'ensemble des variables d'information et de commande formé), l'étape suivante consiste à spécifier les objets graphiques relatifs à ces BIO :

- à chaque variable d'information, on associe un objet graphique d'information,
- à chaque variable de commande, on associe un objet graphique de commande.

Le principe d'association des objets graphiques et de leur gestion dynamique au niveau des différentes vues graphiques est expliqué dans la partie qui suit.

II.4- Spécification de l'Interface

Cette étape consiste à spécifier les deux principales composantes de l'interface (en plus de la composante relative à l'application) conformément au modèle de Seeheim [Pfaff, 85] ou au modèle Arch [Bass et al., 91] ; à savoir : la composante présentation et la composante dialogue. Le principe de la définition de chaque composante est maintenant présenté.

II.4.1- Spécification de la présentation assistée par un système à base de connaissances

La principale problématique au niveau de la spécification de la présentation est la prise en compte des critères et recommandations ergonomiques disponibles à ce sujet (voir chapitre 1 §I.3).

En effet, de nombreuses règles ergonomiques sont proposées dans la littérature pour la spécification des interfaces. Quelques tentatives d'inventaire des règles ergonomiques pour la conception et l'évaluation des applications interactives de contrôle de procédés industriels sont à signaler :

- *Gilmore et al.* [Gimore et al., 89] ont beaucoup contribué à ce sujet et leurs travaux ont abouti aux premières versions du standard NUREG (dédié aux applications nucléaires). Ils ont proposé un guide ergonomique pour la conception des interfaces de contrôle des procédés industriels dans lequel une centaine de recommandations ergonomiques, adaptées à ce domaine particulier sont listées. Les règles présentées dans ce manuel sont organisées selon principalement quatre catégories : (1) les vues graphiques, (2) les périphériques de contrôle et d'entrée/sortie, (3) l'intégration vues/contrôle, (4) la disposition de l'espace de travail et des facteurs de l'environnement. Chaque recommandation est présentée selon un format structuré.

- *O'Hara et al.* [O'Hara et al., 96][O'Hara et al., 99] ont contribué aux dernières versions du standard NUREG et ont proposé la première version d'un système *hypermédia* décrivant les recommandations ergonomiques applicables au domaine de contrôle des procédés. Bien que leurs premiers résultats soient prometteurs, les auteurs soulignent quelques problèmes méthodologiques, conceptuels et pratiques liés aux spécificités de ce domaine d'application particulier.

Il s'agit de souligner que les recommandations ergonomiques ne sont pas bien connues ni utilisées dans la pratique, bien que les besoins industriels et les projets potentiels soient nombreux dans le domaine qui nous intéresse dans ce mémoire. Elles sont en effet, plus fréquentes et plus faciles à formaliser et automatiser pour un certain niveau d'abstraction des objets de l'interface et un peu moins évidentes pour un haut niveau d'abstraction de l'interface [Moussa, 92][Moussa et al., 00a].

Ainsi, le fait de composer une interface à partir d'objets graphiques élémentaires intégrant des critères ergonomiques (bouton, vanne, cadran, etc.), ne garantit pas pour autant une interface ergonomique. Ceci est dû à l'absence de règles ergonomiques de haut niveau d'abstraction.

Pour la présentation des interfaces, dans notre démarche, nous proposons de bénéficier des travaux menés à ce sujet depuis les années 80 par un certain nombre de chercheurs [Alty et al., 85][Johannsen, 95][Moussa et al., 90][Kolski et al., 91][Kolski et al., 96] concernant des approches à bases de connaissances pour la conception et/ou l'évaluation automatiques des interfaces dédiés au contrôle des procédés industriels. En particulier, la première maquette de l'outil Ergo-Conceptor proposé dans les recherches antérieures de notre équipe vise à être capable de décider globalement des vues graphiques appropriées à associer à chaque sous-système à contrôler. Il considère, à cet effet, d'une part les caractéristiques de chaque sous-système à contrôler (l'ensemble de ses différents états de fonctionnement, les BIO associés à chaque état, etc.) et d'autre part, un premier niveau de recommandations ergonomiques spécifiques formalisées et stockées dans sa base de connaissances. Nous nous attacherons à utiliser cette approche de spécification assistée par un système à base de connaissances dans notre démarche.

Les étapes précédentes de notre approche, comme nous l'avons déjà expliqué, aboutissent à l'identification de plusieurs sous-systèmes dont chacun dispose en principe des variables informationnelles et/ou de commande adéquates (figure 3.20) [Moussa et al. 99][Moussa et al., 00b].

La question qui se pose ici est comment décider :

- du nombre de vues graphiques à associer à chaque sous-système : en fonction du volume des BIO,

- des types de vues graphiques : selon le niveau d'abstraction du sous-système et son rôle dans la description du SHM (ex : informationnelle, ou de commande), et
- des types de représentation : selon le type de la vue associée (ex : supervision ou historique pour une vue informationnelle).

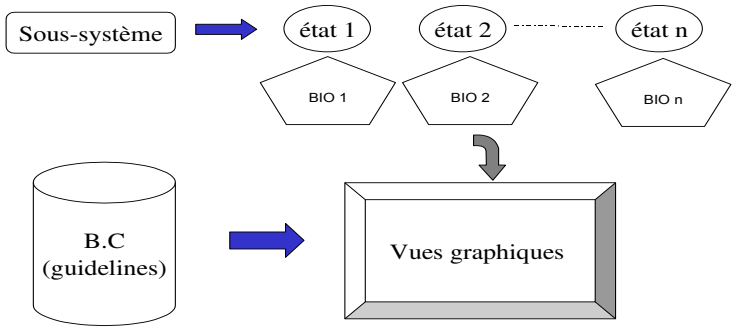


Figure 3.20 : Processus de spécification de la présentation de l'interface

Afin d'assurer la meilleure qualité ergonomique possible, quelques recommandations ergonomiques sont formalisées sous forme clausale et stockées dans la base de connaissances du système Ergo-Conceptor+ qui sera décrit plus particulièrement dans le chapitre 5 (figures 3.21.a et 3.21.b). Ces règles constituent des exemples qui montrent comment on peut décider (1) des types des vues et des représentations associées selon le niveau d'abstraction du sous-système en question, et (2) du choix des objets graphiques à associer aux variables du système.

```
SI niveau_abstraction (sous-sys) =  "bas"
        ALORS générer_vue_commande (sous-sys)
        ET générer_vue_informationnelle (sous-sys)
SI niveau_abstraction (sous-sys) =  "haut"
        ALORS générer_vue_informationnelle (sous-sys)
SI quantite_BIO (sous-sys) >= MAX
        ALORS  générer_multiple_vues (sous-sys)
  NB: (générer_multiple: décompose le sous-système en N vues)
```

Figure 3.21.a : Exemples de recommandations ergonomiques pour la génération des vues graphiques

```
SI type_variable (V) = "informationnelle"
        ET type_vue (sous-sys) = "informationnel"
        ET type_representation (sous-sys) = "historique"
        ALORS  select_courbe (V)
SI type_variable (V) = "informationnelle"
        ET type_vue (sous-sys) = "informationnel"
        ET type_representation (sous-sys) = "supervision"
        ALORS  select_barregraphe (V)
SI type_variable (V) = "commande"
        ET type_vue (sous-sys) = "commande"
        ALORS  select_bouton_commande (V)
```

Figure 3.21.b : Exemples de recommandations ergonomiques pour la sélection des objets de l'interface

Ainsi, pour un niveau d'abstraction bas du sous-système à contrôler (figure 3.21.a), on préconise la génération de deux types de vues : une vue informationnelle de supervision et une vue de commande. Si par contre, le niveau d'abstraction est élevé, une seule vue peut être générée ; c'est la vue informationnelle de supervision. En effet, à un haut niveau d'abstraction on ne peut pas autoriser une commande du système. Il faudrait passer à un niveau plus bas mettant en évidence les différentes variables de commande du sous-système en question.

Par ailleurs, si pour un sous-système et pour une situation de fonctionnement donné, on se retrouve avec une quantité de BIO qui dépasse un nombre maximal recommandé (figure 3.21.b), il faudrait penser à générer plusieurs vues. Chaque vue propose un niveau informationnel raisonnable et gérable par l'opérateur.

Aussi, pour le choix de la représentation graphique d'un objet donné, certaines règles ergonomiques sont proposées. Par exemple, pour le cas de la représentation graphique d'une variable d'information dans une vue informationnelle avec un type de représentation « historique », on recommande l'utilisation d'une courbe. Si par contre, le type de représentation associé à cette vue est « supervision », on conseille d'utiliser les barregraphes. Pour le cas d'une vue de commande, s'il s'agit d'une variable de commande, des modes de représentation « bouton de commande » sont prévus.

Une fois les modes graphiques de représentation de l'interface décidés, la deuxième étape de la spécification de l'interface consiste à préciser et définir le dialogue approprié. Le dialogue de chaque objet graphique est modélisé par les RdPI. Ceci est expliqué ci-dessous.

II.4.2- Spécification du dialogue

Afin de modéliser le dialogue homme-machine, il est maintenant nécessaire de définir le comportement des différents objets graphiques de l'interface. Ce comportement dépend des caractéristiques graphiques des objets, des services qu'ils offrent, du domaine de l'application, de l'évolution du système, etc.

A chaque objet graphique, nous associons une structure de contrôle modélisée par un RdPI. Considérant les deux cas particuliers des objets graphiques identifiés (objet informationnel et objet de commande), nous proposons deux structures génériques de contrôle.

La première structure concerne les objets informationnels (figure 3.22.a). Ces objets passent principalement par deux états : affiché et masqué. La structure de contrôle d'un tel objet doit donc posséder deux transitions particulières dont les actions associées assurent l'affichage et le masquage de l'objet. Le second type de structure concerne les objets de commande (figure 3.22.b). Un objet de commande passe principalement par trois états : masqué, affiché en état désactivé (affiché/désactivé) et affiché en état activé (affiché/activé).

Le dialogue global de l'interface est ainsi décrit par l'ensemble des RdPI proposés ici, autrement dit, celui modélisant l'interaction homme-machine et ceux correspondants aux structures de contrôle des différents objets graphiques.

Se pose, alors, la question de la communication entre ces différents réseaux de Petri. La communication entre ces différents RdPI est assurée moyennant des variables logiques. Les valeurs de ces variables sont positionnées au niveau des transitions particulières "*début action*" et "*fin action*".

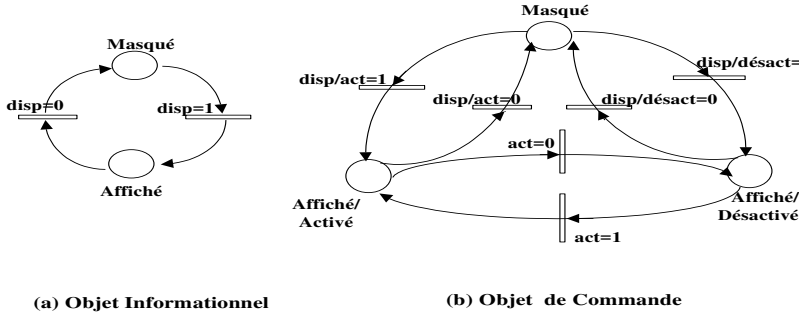


Figure 3.22 : Structures de contrôle des objets de l'interface

La figure 3.23 illustre ce processus de communication permettant d'assurer la gestion dynamique du dialogue homme-machine, avec le cas de la variable d'information VC et la variable de commande CVC identifié dans la figure 3.19 (voir partie identification des BIO §II.3). Elle montre une partie du RdPI modélisant l'interaction homme-machine pour ce cas, ainsi que les structures de contrôle de deux objets graphiques : un objet informationnel associé à la variable d'information VC (par exemple une courbe) et un objet de commande associé à la variable de commande CVC (par exemple un bouton de commande).

Considérons, par exemple, la transition T1 de ce modèle. Supposons que cette transition modélise le début d'une action élémentaire qui consiste à vider le silo de stockage d'un produit C suite à la détection d'un dysfonctionnement D1. L'action associée à cette transition consiste donc à afficher l'objet informationnel VC et activer l'objet de commande CVC. Cette action est modélisée au niveau de la transition par : A = {VC.disp:=1, CVC.disp/act:=1}.

En effet, quatre types de variables logiques sont utilisés pour assurer la communication entre ces différents réseaux de Petri :

- ***disp*** (pour disponible) : c'est une variable logique utilisée pour exprimer le besoin d'afficher ou masquer un objet graphique informationnel sur l'écran. Si disp est positionnée à 1, cela veut dire qu'on demande d'afficher l'objet informationnel. Si par contre, disp est positionnée à la valeur 0, c'est qu'on demande de masquer l'objet informationnel en question.

- ***disp/act*** (pour disponible et activé) : c'est une variable logique permettant d'exprimer les transitions d'un objet de commande entre les états « masqué » et « affiché en état activé ». Si elle est positionnée à 0, c'est qu'on demande de le masquer. Si on la positionne à 1, c'est qu'on demande de rendre disponible cet objet de commande, c'est-à-dire de l'afficher en état activé.

- ***disp /désact*** (pour disponible et désactivé) : c'est une variable logique, positionnée à 1, elle exprime le fait qu'on demande d'afficher un objet de commande en état désactivé. Si elle est positionnée à 0, c'est qu'on souhaite masquer l'objet.

- ***act*** (pour activé) : c'est une variable logique permettant d'exprimer les transitions d'un objet graphique de commande entre les états « affiché/ désactivé » et « affiché/activé ».

Positionnée à 1, elle exprime le fait d'activer l'objet, et positionnée à 0, elle traduit une demande de désactivation.

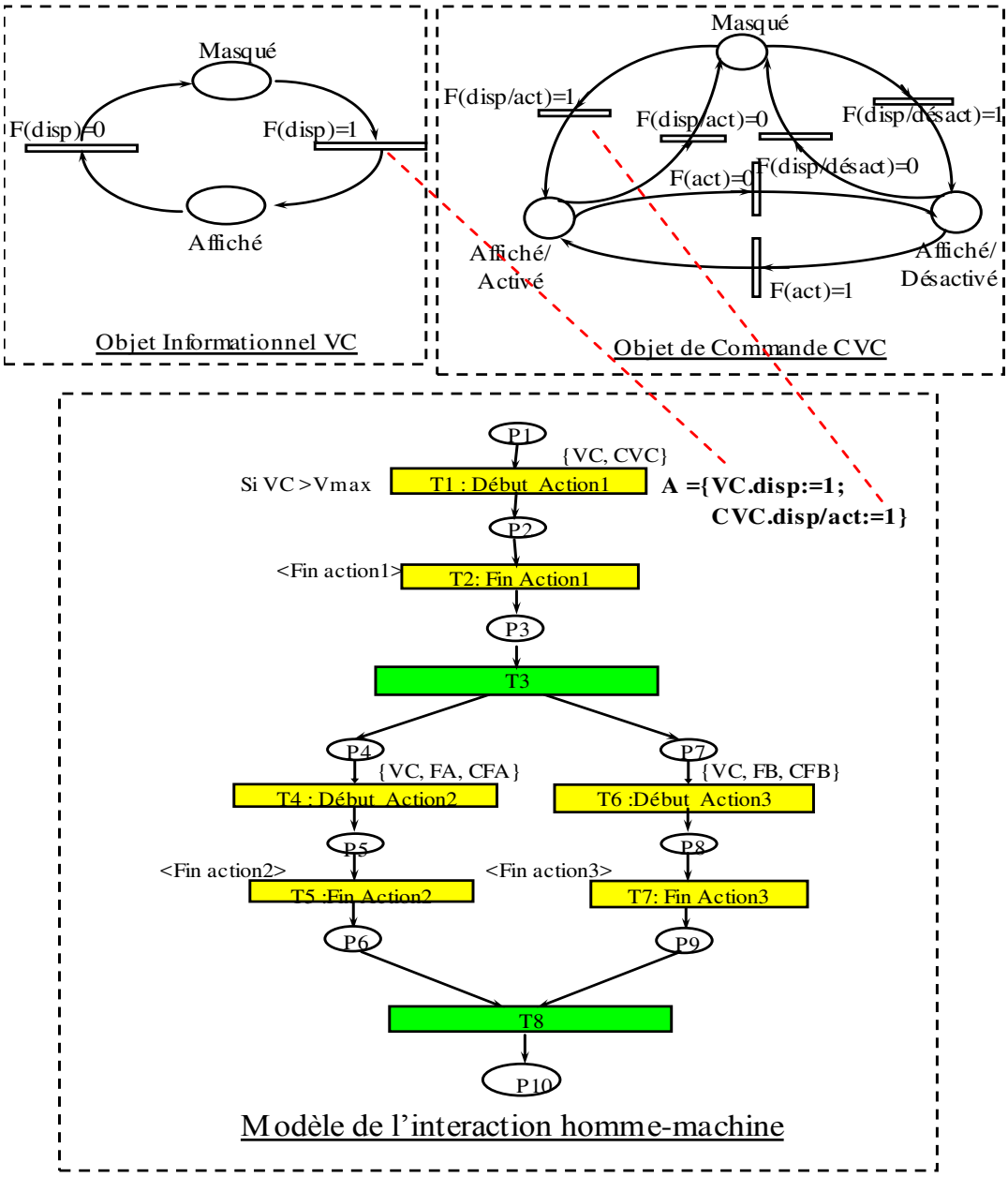


Figure 3.23 : Processus de communication entre les différents RdP

Ainsi, pour le cas de la transition T1, on demande d'afficher l'objet informationnel VC (VC.disp :=1) et d'afficher en état activé l'objet de commande CVC (CVC.disp/act :=1).

Les variables logiques sont évaluées au niveau des conditions associées aux transitions des structures de contrôle des différents objets graphiques en termes de Formule logique (F) (ex : F(disp=0), F(disp/act)=1, etc., (figure 3.23)). Ces formules logiques manipulent donc des variables logiques (attributs propres aux différents objets graphiques) positionnées au niveau

du modèle de l'interaction homme-machine en fonction de l'évolution de l'état du système contrôlé.

D'une manière concrète, moyennant ce mécanisme et selon le contexte de fonctionnement, sur une vue graphique, les objets seront affichés, masqués, affichés en état actif ou affichés en état inactif (grisé par exemple).

II.5- Vérification des propriétés de l'Interface

La validation d'un modèle basé sur les RdP dépend beaucoup de ses caractéristiques. En effet, les extensions apportées en général aux RdP causent des difficultés pour la vérification de plusieurs propriétés. Les RdPI, utilisés ici, sont non autonomes. En dépit de leur perte partielle de quelques résultats théoriques (pour la vérification des propriétés), ils permettent d'exprimer les concepts des systèmes temps réel et de modéliser de manière concise des applications complexes [Brams, 83]. Afin de pouvoir valider un tel modèle, il est donc nécessaire d'associer certaines contraintes et axiomes pour sa construction.

Comme nous l'avons expliqué auparavant (voir §II.2), nous considérons, principalement, que les événements relatifs à des actions de l'opérateur sont toujours présents et qu'une action de l'opérateur doit toujours arriver à sa fin dans n'importe quel cas (grâce à l'intervention de l'opérateur humain ou à l'écoulement d'un certain temps d'attente fixé par un chien de garde associé à ces transitions). Ainsi, nous pouvons garantir que les événements externes ne causent jamais d'interblocage (deadlock) du modèle RdP et nous pouvons donc transiter vers un RdP ordinaire en supprimant tous les événements externes associés au modèle (car ils sont tous relatifs à des actions de l'opérateur humain donc ils vont certainement se produire). De ce fait, la vérification des propriétés devient possible.

En outre, comme la construction du modèle global a été basée sur une technique de fusion des places de structures élémentaires, la composition va donc conserver les propriétés vérifiées sur les structures élémentaires. En d'autres termes, nous pouvons dire que, en composant des structures élémentaires validées, le modèle global est lui même validé.

La structure élémentaire sur laquelle est basée le modèle de l'interaction homme-machine vérifie un certain nombre de 'bonnes' propriétés telles que la bornitude, le non blocage, la vivacité et la persistance. Ces propriétés sont donc garanties, par le processus de construction suivi, pour le modèle global de l'interaction homme-machine [Khelil, 01].

Cette vérification de propriétés structurelles des RdP permet, de facto, de garantir certaines bonnes propriétés de l'interface spécifiée. Nous commençons par exposer, ci-après, quelles sont ces bonnes propriétés à vérifier pour assurer le bon fonctionnement d'une interface donnée. Nous expliquerons, par la suite, le processus suivi pour la vérification de ces propriétés sur le modèle préconisé de l'interaction homme-machine.

II.5.1- Les propriétés à vérifier

Le bon fonctionnement d'une interface homme-machine donnée peut être selon nous assuré principalement par les trois propriétés suivantes :

- *L'absence de blocage* : L'interface proposée ne doit jamais se bloquer sous l'effet de n'importe quelle action de l'utilisateur.

- *La disponibilité de services* : A un moment donné, l'interface doit garantir la disponibilité de certains services relatifs à la situation en cours.

- *L'absence de conflit et la stabilité* : L'interface ne doit jamais présenter de situation de conflit effectif et elle doit assurer une certaine stabilité des différentes vues graphiques qu'elle présente. Il faudrait donc assurer l'absence de tout état d'évolution imprévisible de l'interface et il ne faut pas qu'il y est un conflit de vues dans une situation donnée de fonctionnement.

Ces propriétés de bon fonctionnement sont garanties par des propriétés structurelles des RdP. En effet :

- La propriété de *bornitude* garantit un nombre fini d'états possibles du système ce qui se traduit par une stabilité des vues graphiques au niveau de l'interface pour ces différents états.

- La propriété de *vivacité* interprète la potentialité du système à atteindre tous les services possibles modélisés et garantit l'absence de blocage partiel ou total ; ce qui se traduit par l'absence de blocage de l'interface réalisée et l'accessibilité aux différents services qu'elle offre.

- la propriété de *persistance* garantit l'absence de conflit au niveau du réseau et par la suite la stabilité de l'interface et l'absence de confusion ou ambiguïté avec les différents états du système.

Nous présentons de suite, le principe de vérification de ces propriétés structurelles sur le modèle de l'interaction homme-machine adopté.

II.5.2- Principe de vérification

Le processus de construction par composition, étudié dans cette approche, a pour but d'établir les règles qui permettent de systématiser cette construction mais surtout de garantir, par avance, la vérification des propriétés caractéristiques de la bornitude, de la vivacité et de la persistance. Nous montrons ci-dessous de quelle manière la vérification des dites propriétés est effectivement garantie par ce processus de construction.

La propriété de persistance est vérifiée car les structures élémentaires et les compositions utilisées pour la construction du modèle de la tâche utilisateur n'admettent aucun conflit structurel qui peut conduire à un conflit effectif (voir §II.2).

Rappelons que le processus de construction par de structures élémentaires modélisant les actions opérateurs (SEAO) et construit, par compositions successives, des réseaux que nous allons intentionnellement désigner par « réseaux correctement construits (RCC) ». La dernière composition de fermeture transforme le dernier RCC obtenu en un modèle de description de tâche utilisateur.

Un RCC peut être défini comme suit :

RCC = SEAO \ RCC obtenu par application d'une composition unique à un ensemble de RCC.

Un RCC a toujours une seule place d'entrée *début* et une seule place de sortie *fin*. Il a les propriétés suivantes :

Pr1 : Toute marque déposée dans la place *début* du RCC finit, tôt ou tard, par évoluer vers la place de sortie *fin*.
Pr2 : Dans son évolution de la place *début* à la place *fin*, et pour toute transition choisie dans le RCC on peut toujours trouver une configuration de valeurs des variables (internes et/ou opérateur) telle que l'évolution de la marque conduit à franchir cette transition.
Pr3 : Une fois arrivée à la place *fin*, toutes les places du RCC depuis la place *début* à part la place *fin*, auront un marquage vide.

En conséquence, si tout RCC vérifie ces trois propriétés, la composition finale de fermeture de tout RCC conduit à un modèle de tâche initialisé et ayant la propriété de bornitude et de vivacité.

La vérification de ces trois propriétés est montrée ci-après pour la structure élémentaire SEAO et pour une composition séquentielle de deux ou de plusieurs structures élémentaires.

- **Vérification des propriétés pour une structure élémentaire :**

Rappelons qu'une structure élémentaire SEAO se compose d'une place *début* P1, d'une place *action* P2 et d'une place *fin* P3 (figure 3.3). Le déclenchement de la transition T1 est conditionné par la condition (condition i). Le franchissement de la transition T2 se fera systématiquement sur occurrence de l'événement de fin d'exécution de l'action associée à la place P2. Moyennant la précaution d'interprétation du §II.2, toute marque placée dans P1 évoluera tôt ou tard vers P2 par la vérification de la condition (condition i) et passera à la place P3 dès la fin de cette action.

Proposition 1 : Si un réseau R vérifie les propriétés Pr1, Pr2 et Pr3 alors la réduction de R par suppression d'une SEAO non initialisée (figure 3.24) préserve la réciprocité concernant la vérification des propriétés Pr1, Pr2 et Pr3.

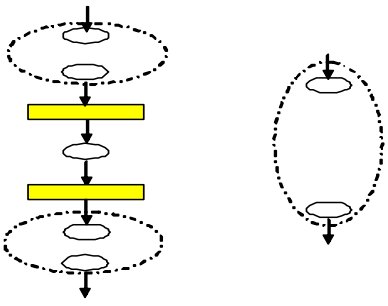


Figure 3.24 : Suppression d'une SEAO

Preuve : De par la structure d'une SEAO, du fait qu'elle vérifie les propriétés Pr1, pr2 et Pr3 et tenant compte des précautions d'interprétations, la proposition se justifie directement par les approches classiques d'analyse par réduction.

- **Vérification des propriétés pour une composition séquentielle :**

D'après la proposition 1, la vérification des propriétés Pr1, Pr2 et Pr3 pour une composition séquentielle de deux ou plusieurs SEAO est triviale. En effet, une telle composition est réalisée en fusionnant la place *début* d'une SEAO et la place *fin* d'une autre SEAO à composer en séquentiel. En appliquant des réductions successives de SEAO, on aboutit à une seule SEAO qui vérifie, par hypothèse les trois propriétés.

La vérification étudiée de ces trois propriétés pour les autres compositions est présentée dans [Khelil, 01].

Nous arrivons, à présent, à l'avant dernière étape de l'approche, à savoir la génération de l'interface.

II.6- Génération de l'interface

Comme expliqué au §II.4 la spécification de la présentation de l'interface se base sur une approche à base de connaissances permettant de décider des vues et objets graphiques adéquats en fonction des BIO identifiés et des recommandations ergonomiques proposées dans le domaine de contrôle des procédés industriels complexes. Cette approche de spécification et génération graphique est supportée dans nos travaux par l'outil informatique Ergo-Conceptor+. Cet outil intègre à cet effet :

(i) un système à base de connaissances pour la spécification et génération des vues et objets graphiques, et
(ii) un joueur de RdPI permettant de gérer l'évolution du dialogue homme-machine et la dynamique de l'affichage des objets graphiques.

Ces deux principaux modules de l'outil Ergo-Conceptor+ sont succinctement présentés ici. Ils sont détaillés au niveau du chapitre 5.

II.6.1- La génération graphique

Le système à base de connaissances intégré dans l'outil Ergo-Conceptor+ assure la prise en considération de recommandations ergonomiques dans le processus de spécification et de génération semi-automatique des vues graphiques.

En effet, ce système doit regrouper un ensemble de recommandations ergonomiques du domaine rassemblées, organisées et formalisées sous forme de règles de production (figure 3.25). Ces règles sont exploitées par la suite, moyennant un moteur d'inférence capable de décider des vues et objets graphiques appropriés à un instant donné, en fonction des BIO fournis et d'une base de bibliothèque d'objets graphiques prédéfinis.

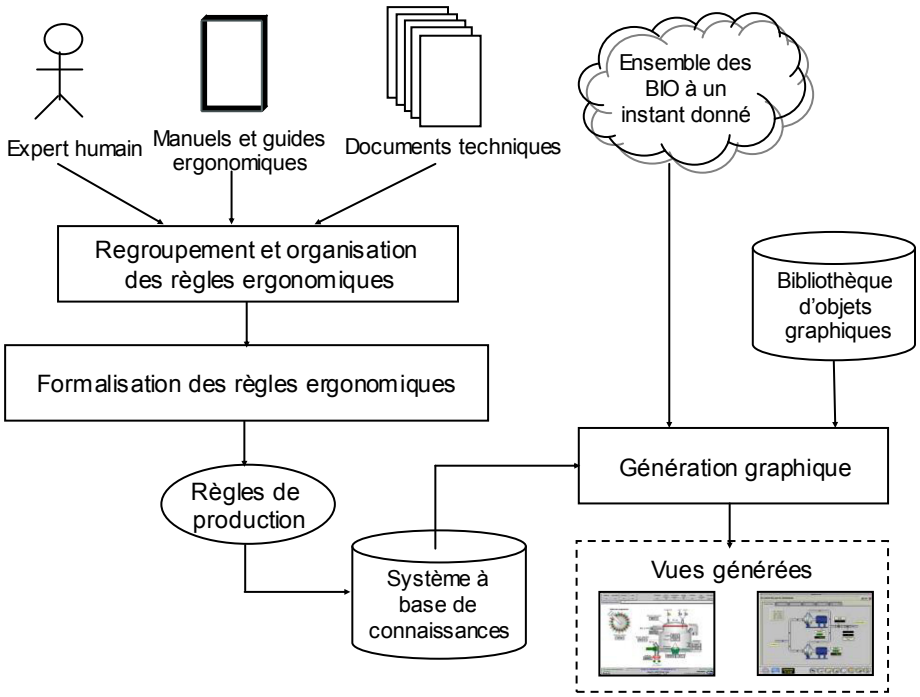


Figure 3.25 : Système à base de connaissances de l'outil Ergo-Conceptor+

II.6.2- La gestion du dialogue homme-machine

Une fois les vues générées, la gestion de leur dynamique est assurée par le module joueur des RdPI (appelé aussi interpréteur), du système Ergo-Conceptor+. Son principe consiste, en effet, à interpréter l'évolution des marquages des différents RdPI du système modélisant le dialogue homme-machine, en fonction de l'évolution de l'état du SHM et des différents événements qui se produisent sur le procédé.

Le joueur des RdP est basé principalement sur sept composants (figure 3.26) :

1- la base de données du procédé contenant les différentes variables du système,
2- la base de données des objets de l'Interface contenant l'ensemble des instances créées,
3- les matrices "Pre" des différents réseaux de Petri, indiquant les conditions de franchissement des transitions,
4- les matrices "Post" des différents réseaux de Petri, exprimant ce qu'on doit faire lors du franchissement des transitions,
5- l'ensemble des marquages courants des différents réseaux de Petri,
6- le code des différents programmes des traitements relatifs aux actions associées aux transitions des réseaux, et
7- une liste des événements produits pour simuler l'évolution de l'état du système et par suite l'évolution des différents réseaux de Petri.

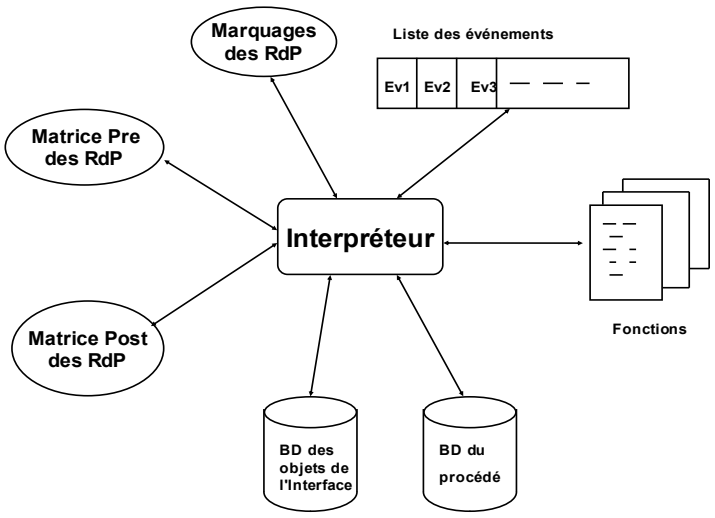


Figure 3.26 : Joueur ou interpréteur des RdPI

Afin de refléter les spécificités des RdPI employés dans notre approche de modélisation de l'interaction homme-machine, nous proposons de compléter la description des matrices « Pre » et « Post », permettant de définir les règles de franchissement des RdP, par trois vecteurs (figure 3.27) :

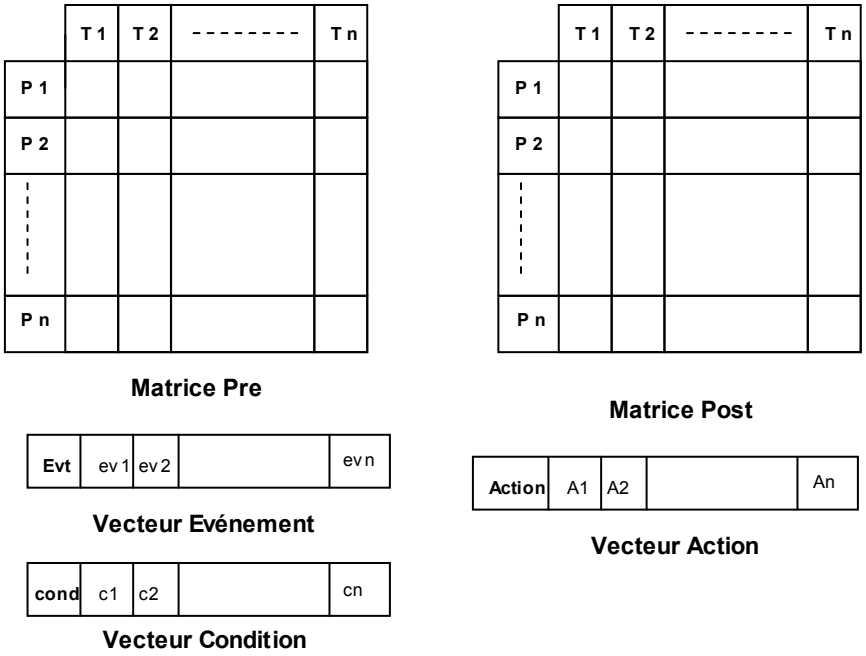


Figure 3.27 : Structures des matrices « Pre » et « Post »

- un vecteur *Evénement* associant pour chaque transition l'événement nécessaire pour la rendre franchissable,

- un vecteur *Condition* précisant les conditions nécessaires aux franchissement de chaque transition, et

- un vecteur *Action* associant les actions respectives à accomplir au franchissement d'une transition donnée.

Rappelons que l'outil Ergo-Conceptor+ et les détails de conception de ces deux modules sont présentés au niveau du chapitre 5.

II.7- Intégration et évaluation de l'IHM intégrée

Une fois l'interface générée, l'étape qui suit dans le cycle de vie des systèmes informatiques, consiste à l'intégrer dans le système global de contrôle du procédé en question et procéder à son évaluation.

L'évaluation d'un système homme-machine (SHM) concerne généralement deux types de critères [Grislin, 95][Grislin et al., 97][Kolski, 97][Nielsen, 93][Bastien et Scapin, 01] :
- les critères de performances du SHM, calculés en termes d'écart entre les objectifs de l'entreprise et les résultats obtenus, et
- les critères ergonomiques, liés à l'utilisation du système interactif vis-à-vis de la réalisation de ses tâches dans différents contextes de fonctionnement (normaux et anormaux).

Au niveau de cette étape, c'est d'évaluation *a posteriori* dont il s'agit ici. L'évaluation *a priori* est, en effet, assurée avec (1) l'intégration d'un outil du type TFWWG permettant la prise en considération des critères ergonomiques de présentation dès les premières étapes de la conception de l'interface, et (2) l'utilisation d'une technique formelle pour la modélisation de l'interaction homme-machine garantissant les bonnes propriétés de l'interface, qui sont d'ailleurs vérifiées avant la génération proprement dite de l'interface.

L'évaluation *a posteriori* se fait, dans une première étape, dans un contexte simulé de fonctionnement pour ne pas perturber le fonctionnement réel du procédé et par la suite sa production. Puis, une fois cette étape validée, on passe à une évaluation sur site réel. Différentes techniques et méthodes existent à cet effet. Leur choix dépend de leurs conditions d'application et de l'aménagement de la salle de contrôle [Taborin, 89]. Ce travail d'intégration et d'évaluation de l'IHM ne sera pas traité dans le cadre de ce mémoire.

Conclusion

Ce chapitre a été consacré à la présentation de l'approche méthodologique de spécification, de vérification et de génération semi-automatique d'interfaces homme-machine, élaborée dans le cadre de ce travail de recherche. Cette approche méthodologique vise à satisfaire le cahier des charges identifié au niveau du chapitre 1, et à couvrir toutes les étapes nécessaires depuis l'analyse du SHM jusqu'à la génération semi-automatique de l'interface, en passant par la modélisation de l'interaction homme-machine, la déduction des BIO et la spécification des différentes vues et objets graphiques.

Ces différentes étapes ont été détaillées tout le long de ce chapitre. Une attention particulière a

été accordée à l'outil formel préconisé pour la modélisation de l'interaction homme-machine, en l'occurrence les réseaux de Petri interprétés (RdPI), et à son apport pour la vérification des bonnes propriétés et l'évaluation *a priori* de l'interface.

La faisabilité de cette démarche fait l'objet du chapitre suivant à travers une étude de cas relative à une application industrielle simplifiée.

Chapitre 4

Application de la méthodologie sur une application industrielle : de l'analyse du SHM à la vérification des propriétés de l'IHM

Plan du chapitre

Introduction
I. Présentation de l'application industrielle
II. Application de la démarche
    II.1- Analyse du SHM
    II.2- Modélisation de l'interaction homme-machine par les RdP
    II.3- Déduction des BIO
    II.4- Spécification de l'interface
    II.5- Vérification des propriétés de l'interface
Conclusion

Dans le chapitre précédent, nous avons présenté et détaillé les différentes étapes de l'approche préconisée, dans le cadre de ce travail de recherche, pour la conception des IHM dédiées au contrôle des procédés industriels. Nous proposons de montrer dans ce chapitre, son applicabilité sur un cas d'application industrielle simplifiée.

Pour ce faire, nous commençons, dans une première partie, par la présentation de l'application industrielle en question et sa spécification fonctionnelle et technique. Nous passons, par la suite, à l'application de toutes les étapes de la démarche dans l'ordre et nous présentons les résultats obtenus.

I. Présentation de l'application industrielle

L'application choisie concerne le cas d'un procédé industriel de fabrication de godets métalliques remplis par une solution chimique préparée préalablement. Le processus simplifié de préparation comprend deux phases principales (figure 4.1) :

- une première phase qui consiste à façonner les godets par emboutissage de pièces métalliques prédécoupées ;
- une deuxième phase qui consiste à verser dans les godets une dose de solution chimique. La solution est obtenue à partir d'une préparation primaire P et d'un produit S.

L'étude a porté sur cette dernière phase qui met en œuvre deux modules : un module de préparation et un module de remplissage.

Le module de remplissage comprend principalement un réservoir de solution et un doseur de remplissage. Les godets façonnés sont présentés un à un pour remplissage. L'arrivée d'un godet à l'emplacement d'un remplissage est détectée par une cellule photoélectrique (signal C=1). La commande du doseur de remplissage est alors activée jusqu'à la réception d'un signal impulsionnel de fin de remplissage (FR). Une fois rempli, le godet est évacué automatiquement et un nouveau godet à remplir prend sa place. La présentation des godets vides et leur évacuation une fois remplis sont supposées être commandées par un module à part.

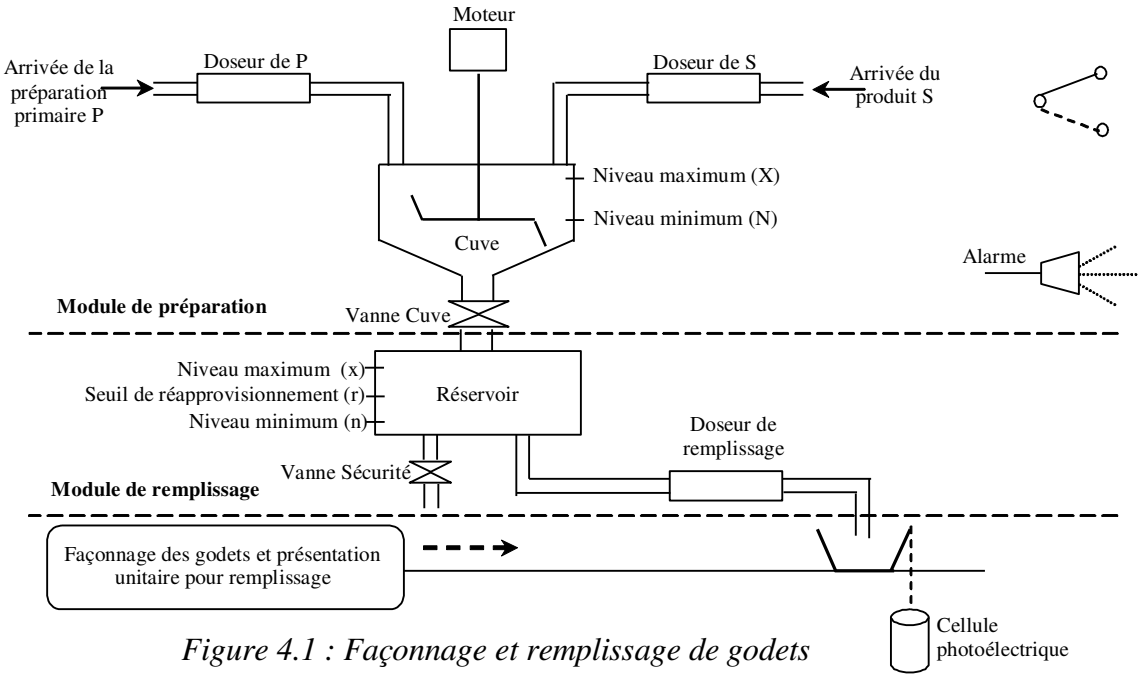


Figure 4.1 : Façonnage et remplissage de godets

Le réservoir de solution est équipé d'une électrovanne de sécurité pour la commande de vidange du réservoir et de trois détecteurs de niveau :

- un détecteur de niveau minimum : n (n =1 ssi le niveau minimum de la solution est atteint sinon n = 0),
- un détecteur de niveau maximum : x (x =1 ssi le niveau maximum de la solution est atteint sinon n = 0),

- un détecteur de niveau seuil réapprovisionnement : r (r =1 ssi le niveau seuil de réapprovisionnement existe).

Le module de préparation est équipé principalement de deux doseurs similaires à celui de remplissage pour le dosage des produits P et S, d'une cuve, d'un moteur pour agiter la solution et d'une électrovanne pour commander le remplissage de la solution dans le réservoir.

A chacun des deux doseurs de ce module sont associées deux signaux de commande pour Début et Fin de dosage (DP et FP pour le dosage du produit P ; DS et FS pour le dosage du produit S). Aux deux doseurs sont également associés deux dispositifs permettant de régler la quantités de dosage respectives et des dispositifs de chauffage et refroidissement permettant d'ajuster les températures des solutions.

La cuve est équipée de deux détecteurs de niveau :
- un détecteur de niveau minimum : N (N =1 ssi le niveau minimum est atteint),
- un détecteur de niveau maximum : X (X =1 ssi le niveau maximum est atteint),

Trois thermomètres sont prévus pour mesurer les températures de la solution primaire P, du produit S et de la solution de mélange.

Le module de préparation est commandé par un relais à deux positions (marche/arrêt). Dès la mise du relais en position marche, le module de préparation se met à développer deux fonctions parallèles :
- d'un côté, le moteur est commandé pour agiter la solution ;
- d'un autre côté, chaque fois que le niveau seuil de réapprovisionnement du réservoir disparaît (r =0), il y a déclenchement d'un cycle de préparation-réapprovisionnement du réservoir. A savoir, injection dans la cuve d'une dose du produit P, puis injection d'une dose du produit S, puis commande de l'électrovanne pour le remplissage du réservoir jusqu'au niveau maximum.

La remise du relais en position arrêt ramène le module à l'état de repos.

Six contraintes de fonctionnement s'imposent pour ce système :

C1 : La commande début remplissage du godet ne peut être engagée que si le niveau minimum est présent dans le réservoir.
C2 : Une fois engagée, l'opération de remplissage ne peut plus être interrompue jusqu'à la détection de l'impulsion de fin de remplissage du godet (le niveau minimum suffit pour permettre une opération de remplissage d'un godet)
C3 : La commande du moteur doit être suspendue si le niveau N n'est pas atteint dans la cuve.
C4 : Les commandes de dosage de la solution primaire P et du produit S ainsi que la commande de réapprovisionnement doivent également être suspendues si le remueur n'est pas actif.
C5 : L'exécution effective de l'arrêt du fonctionnement du module de préparation (par suite de la mise du relais en position arrêt) ne peut intervenir qu'en dehors du cycle de préparation-réapprovisionnement.
C6 : L'opération de remplissage n'est autorisée que si la température de la solution chimique mélangée ne dépasse pas les 35°C. Cette température dépend des températures

des composants P et S utilisés pour cette préparation. L'ajustement de la température du mélange peut donc être assuré par l'action sur les températures des produits en entrée P et S.

Le procédé pédagogique proposé ci-dessus est intéressant à plusieurs titres. En effet, en dépit d'une structure apparemment simple, cette application recouvre un large spectre des problématiques rencontrées dans le contrôle des procédés industriels.

Il est question de supervision, de détection d'alarme, de commande et de suivi. L'ensemble de ces tâches est affecté à un opérateur humain. C'est donc un cas typique permettant de vérifier l'ensemble des concepts avancés dans la démarche globale proposée.

II. Application de la démarche

La démarche préconisée, comme expliquée au chapitre précédent, commence par procéder à une analyse du SHM. Elle consiste par la suite, à modéliser l'interaction homme-machine moyennant les RdPI et à associer l'ensemble des BIO relatifs à chaque état de fonctionnement du système. La spécification de l'interface est assurée par l'intégration d'un système à base de connaissances intégrant un ensemble de recommandations ergonomiques du domaine. Toutes ces étapes sont illustrées ci-dessous sur le cas de l'application industrielle étudiée.

II.1 - Analyse du SHM

II.1.1- Décomposition fonctionnelle du système par SADT

La fonctionnalité principale de la partie étudiée de ce système consiste à préparer le mélange puis en remplir les godets. La première étape de notre démarche consiste à appliquer la méthode SADT pour décomposer le système et déceler les principaux sous-systèmes élémentaires pour les étudier.

L'actigramme A0 (figure 4.2) présente la fonctionnalité principale "préparer mélange et remplir godets".

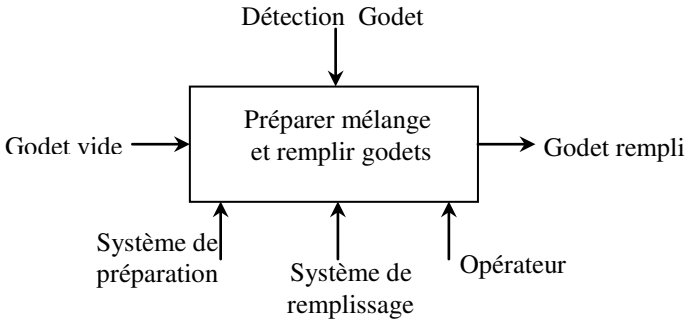


Figure 4.2 : Décomposition du système par SADT
(Actigramme A0)

Une décomposition de cette fonctionnalité peut révéler, dans l'actigramme A1, deux sous-fonctionnalités "préparer mélange" et "remplir godets" (figure 4.3).

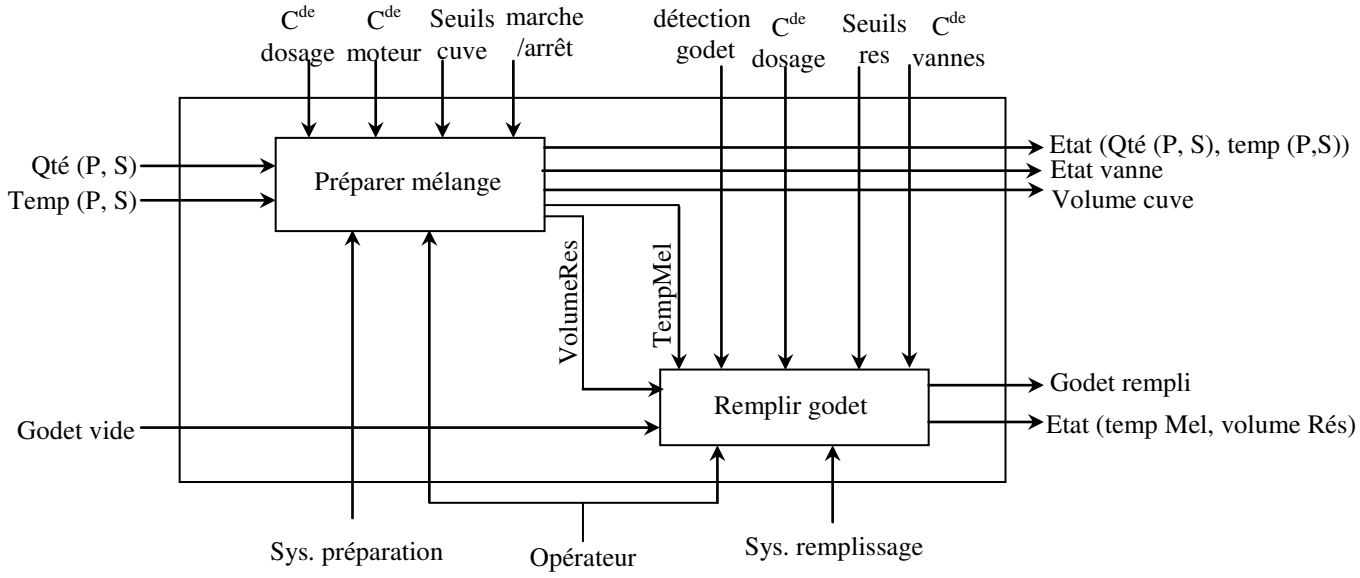


Figure 4.3 : Décomposition du système par SADT
(Actigramme A1)

Le système étudié étant simple, nous pouvons nous arrêter à ce premier niveau de décomposition et considérer les deux sous-systèmes élémentaires :

- sous-système de préparation (S1)
- sous-système de remplissage (S2)

L'étape suivante consiste à analyser ces deux sous-systèmes et étudier de près leurs différents composants et modes de fonctionnement normaux et anormaux. Avant de passer à cette étape d'analyse, nous listons ci-dessous la nomination de l'ensemble des variables du système.

Les variables du système :

On distingue principalement deux types de variables : les variables de commande et les variables d'information. Pour ces dernières, trois classes existent selon les types de données informationnelles. On trouve les données entrées par l'opérateur, les données de sortie (résultats des capteurs et signaux d'alarme) et les données informationnelles de type consignes.

a- Variables d'information de type entrées par l'opérateur (symbole utilisé : In)

In.QteP : donnée de l'opérateur fixant la quantité de dosage de la solution primaire P.
In.QteS : donnée de l'opérateur fixant la quantité de dosage du produit S.
In.TempP : donnée de l'opérateur fixant la température de la solution primaire P.
In.TempS : donnée de l'opérateur fixant la température du produit S.

b- Variables d'information de type sorties (symbole utilisé : Out)

Out.QteP : Quantité dosée de la solution primaire P mesurée par le capteur à la sortie du doseur P.
Out.QteS : Quantité dosée du produit S mesurée par le capteur à la sortie du doseur S.

Out.QteMel : Quantité dosée du mélange mesurée par le capteur à la sortie du doseur de remplissage des godets.
Out.TempP: Température de la solution primaire P mesurée par un thermomètre placé à la sortie du doseur P.
Out.TempS : Température du produit S mesurée par un thermomètre placé à la sortie du doseur S.
Out.TempMel : Température du mélange mesurée par un thermomètre placé à la sortie du doseur de remplissage des godets.
Out.VolCuve : Volume courant du mélange dans la cuve.
Out.VolRes : Volume courant du mélange dans le réservoir.
Out.FindosP : signal de fin de dosage de la solution primaire P.
Out.FindosS : signal de fin de dosage du produit S.
Out.FindosMel : signal de fin de dosage du mélange de remplissage.
Out.Moteur : information sur l'état du moteur (marche ou arrêt).
Out.VanCuve : information sur l'état de la vanne-cuve (ouverte ou fermée).
Out.VanSecurite : information sur l'état de la vanne-sécurité du réservoir (ouverte ou fermée).
Out.DetectGodet : signal indiquant la présence ou non d'un godet à l'emplacement prévu pour le remplissage.
Out.AlarmeMinCuve : signal d'alarme indiquant l'absence du niveau minimum dans la cuve.
Out.AlarmeMaxCuve : signal d'alarme indiquant l'atteinte du niveau maximum dans la cuve.
Out.AlarmeMinRes : signal d'alarme indiquant l'absence du niveau minimum dans le réservoir.
Out.AlarmeMaxRes : signal d'alarme indiquant l'atteinte du niveau maximum dans le réservoir.
Out.AlarmeTempMel : signal d'alarme indiquant le dépassement de la température de consigne du mélange de remplissage.

c- Variables d'information de type consignes (symbole utilisé : Cs)

Cs.MinCuve : consigne du volume minimal du mélange qui doit être présent dans la cuve.
Cs.MaxCuve : consigne du volume maximal du mélange qui peut exister dans la cuve.
Cs.MinRes : consigne du volume minimal du mélange qui doit être présent dans le réservoir.
Cs.MaxRes : consigne du volume maximal du mélange qui peut exister dans le réservoir.
Cs.ReaproRes : consigne de la quantité de réapprovisionnement du mélange dans le réservoir.
Cs.Godet : consigne de la quantité de remplissage des godets.

d- Variables de commande (symbole utilisé : Cd)

Cd.DebdosP : commande de début de dosage de la solution primaire P.
Cd.DebdosS : commande de début de dosage du produit S.
Cd.DebdosMel : commande de début de dosage du mélange pour remplissage du godet.
Cd.VanCuve : commande d'ouverture ou de fermeture de la vanne de la cuve.
Cd.VanSecurite : commande d'ouverture ou de fermeture de la vanne sécurité du réservoir.
Cd.Moteur : commande de (marche/arrêt) du moteur (remueur du mélangeur).
Cd.OnOff : commande ouvert/fermé relativement à la position d'un relais marche/arrêt du système de préparation.

II.1.2- Analyse des dysfonctionnements du système

II.1.2.1- Analyse par la méthode AMDE (Analyse des Modes de Défaillances et leurs Effets)

Rappelons ici que le principe de l'AMDE consiste à étudier pour chaque sous-système l'ensemble de ses composants matériels et recenser pour chacun de ces composants :

- les modes de défaillances envisageables
- les causes possibles de ces défaillances
- leurs effets sur le sous-système étudié
- leurs effets sur les autres systèmes

avec éventuellement :

- les moyens de détection
- les actions éventuelles de l'opérateur

L'élaboration d'une AMDE (voir chapitre 2) demande une connaissance approfondie du fonctionnement normal et/ou dégradé du système étudié. Le recensement des modes de défaillances et de leurs causes éventuelles s'appuie sur l'expérience d'exploitation acquise pour des matériels similaires.

Les principales difficultés surgissent au niveau :

- de la définition des effets des défaillances : variations des paramètres, actions automatiques ou manuelles,…
- de l'évaluation de ces effets.

Au cours de l'élaboration de l'AMDE, un contact permanent entre le spécialiste en sûreté de fonctionnement et les spécialistes des systèmes est nécessaire. Dans le cas de combinaisons des défaillances non prévues lors de la conception du système, ce dialogue permet d'aboutir à l'élaboration d'hypothèses aussi réalistes que possible [Villemeur, 88]. Pour notre cas d'application, l'AMDE va être élaborée pour les deux sous-systèmes mis en évidence avec la décomposition SADT :

Le premier sous-système (S1) est responsable de la fonctionnalité de préparation du mélange. Il est composé de deux doseurs (l'un pour la solution primaire P et l'autre pour le produit S), d'une cuve et d'un moteur pour remuer la solution, d'une vanne-cuve permettant l'écoulement de la solution vers le réservoir et de cinq capteurs (pour mesurer les températures des entrées P et S ainsi que le volume courant dans la cuve et contrôler l'atteinte des niveaux seuil minimum et seuil maximum).

Le deuxième sous-système (S2) est responsable de la fonctionnalité de remplissage des godets. Il est composé d'un réservoir, d'une vanne de sécurité permettant l'évacuation du mélange vers l'extérieur, d'un doseur du mélange pour le remplissage et d'un capteur de température de la solution de remplissage.

L'élaboration de l'AMDE consiste donc à considérer ces deux sous-systèmes et étudier pour chacun d'eux, les modes de défaillances de leurs composants, les causes possibles, ainsi que les effets de ces défaillances sur le système, les moyens de détection et les éventuelles actions de l'opérateur. Le résultat de cette analyse est résumé dans le tableau 4.1 ci-dessous.

Tableau 4.1 : Analyse du système par l'AMDE

Sous-système de préparation (S1)

Composant	Modes de défaillances	Causes possibles	Effets sur S1	Effets sur tout le système	Détection	Actions de l'Opérateur
Doseur P	Destruction de dosage (DD)	Usure du doseur défaillance accidentelle	Problème dosage P arrêt moteur arrêt système préparation	arrêt système de remplissage *arrêt système*	In.QteP <> Out.QteP	
	Défaillance à la sollicitation (DS)	Usure du doseur Entrée non valide par l'op. *(exemple : valeur de In.qteP impossible)*	Problème dosage P arrêt moteur arrêt système préparation	arrêt système de remplissage *arrêt système*	Out.QteP = 0	
	Signal intempestif de fin de dosage (SI)	Usure du doseur Erreur de l'op. dans le réglage de la valeur de la consigne	Absence seuil min dans la cuve Atteinte seuil max dans la cuve	arrêt système de remplissage *arrêt système*	Out.FindosP = 1 et Out.QteP <> In.QteP Ou Out.FindosP = 0 avec temps dépassé	

Tableau 4.1 : Analyse du système par l'AMDE (suite)

Doseur S	Destruction de dosage (DD)	Usure du doseur défaillance accidentelle	Problème dosage S arrêt moteur arrêt système préparation	arrêt système de remplissage *arrêt système*	In.QteS <> Out.QteS	
	Défaillance à la sollicitation (DS)	Usure du doseur Entrée non valide par l'op. *(exemple : valeur de In.qteP impossible)*	Problème dosage S arrêt moteur arrêt système préparation	arrêt système de remplissage *arrêt système*	Out.QteS = 0	
	Signal intempestif de fin de dosage (SI)	Usure du doseur Erreur de l'op. dans le réglage de la valeur de la consigne	Absence seuil min dans la cuve Atteinte seuil max dans la cuve	arrêt système de remplissage *arrêt système*	Out.FindosS = 1 et Out.QteS <> In.QteS Ou Out.FindosS = 0 avec temps dépassé	
Moteur	Défaillance moteur (ou refus de démarrage)	Défaillance mécanique (DM) Défaillance électrique (DE)	Problème dosage P Problème dosage S arrêt système préparation	Absence seuil min dans le réservoir Arrêt sys. remplissage	Out.moteur = 0	Fermeture vanne ouva

Tableau 4.1 : Analyse du système par l'AMDE (suite)

Cuve	Absence seuil min dans la cuve	Blocage vanne ouva ouverte Fuite conduite (P ou S) Bouchage conduite (P ou S) Fuite cuve Fuite conduite cuve-vanne ouva	arrêt moteur arrêt système préparation	Absence seuil min dans le réservoir Arrêt sys. remplissage	Out.AlarmeMinCuve = 0	Fermeture vanne ouva
	Atteinte seuil max dans la cuve	Blocage vanne ouva fermée Bouchage conduite cuve-vanne ouva	arrêt moteur arrêt système préparation	Arrêt sys. remplissage	Out.AlarmeMaxCuve = 1	Ouverture vanne ouva
Vanne Cuve	Refus Ouverture	Défaillance mécanique (DM) Défaillance électrique (DE) Erreur Opérateur	Arrêt moteur niveau max cuve atteint	Absence niveau min Rés Arrêt sys. Remplissage	Out.VanCuve <> Cd.VanCuve	
	Refus Fermeture	Défaillance mécanique (DM) Défaillance électrique (DE) Erreur Opérateur	Arrêt moteur Absence seuil min cuve		Out.VanCuve <> Cd.VanCuve	
	Réponse intempestive	Défaillance mécanique (DM) Défaillance électrique (DE)	Arrêt moteur Absence seuil min cuve	Arrêt sys. Remplissage Absence niveau min Rés	Out.AlarmeMaxRes = 1 et Out.VanCuve = 1 Cd.VanCuve = 1 et Out.VanCuve =0	

Tableau 4.1 : Analyse du système par l'AMDE (suite)

		Erreur Opérateur	ou atteinte seuil max cuve	ou Atteinte seuil max Res		
Capteur Température.P	Signal intempestif de haute ou basse température	Dérive capteur	défaillance capteur température P	arrêt système remplissage	In.tempP <> Out.tempP	
	Défaillance à la sollicitation	Défaut capteur	défaillance capteur température P	arrêt système remplissage	In.tempP <> Out.tempP	
Capteur Température.S	Signal intempestif de haute ou basse température	Dérive capteur	défaillance capteur température S	arrêt système remplissage	In.tempS <> Out.tempS	
	Défaillance à la sollicitation	Défaut capteur	défaillance capteur température S	arrêt système remplissage		
Capteur Volume Cuve	Fonctionnement intempestif	Dérive capteur	arrêt moteur	arrêt système remplissage	Out.VolCuve < min et Out.AlarmeMinCuve=1	
	Défaillance à la sollicitation	Défaut capteur	arrêt moteur	arrêt système remplissage	Out.VolCuve > max et Out.Alarme.MaxCuve=0	
Capteur niveau min Cuve	Fonctionnement intempestif	Dérive capteur	arrêt moteur	arrêt système remplissage	Out.VolCuve < min et Out.Alarme.MinCuve=1	
	Défaillance à la sollicitation	Défaut capteur	arrêt moteur	arrêt système remplissage	Out.VolCuve >min et Out.AlarmeMinCuve=0	
Capteur niveau max Cuve	Fonctionnement intempestif	Dérive capteur	arrêt moteur	arrêt système remplissage	Out.VolCuve < max et Out.AlarmeMaxCuve=1	
	Défaillance à la sollicitation	Défaut capteur	arrêt moteur	arrêt système remplissage	Out.VolCuve > max et Out.AlarmeMaxCuve=0	

Tableau 4.1 : Analyse du système par l'AMDE (suite)

Sous-Système de remplissage (S2)

Composant	Modes de défaillances	Causes possibles	Effets sur S2	Effets sur tout le système	Détection	Actions de l'Opérateur
Doseur Mel	Destruction de dosage (DD)	Usure du doseur défaillance accidentelle	mauvais remplissage du godet	arrêt système préparation	Out.QteMel <> Cs.Godet	
	Défaillance à la sollicitation (DS)	Usure du doseur Entrée non valide par l'op.	mauvais remplissage du godet	arrêt système préparation	Out.QteMel = 0	
	Signal intempestif de fin de dosage (SI)	Usure du doseur Erreur de l'op. dans le réglage de la valeur de la consigne	mauvais remplissage du godet	arrêt système préparation	Out.FindosMel= 1 et Out.QteMel<> Cs.Godet Ou Out.FindosMel= 0 avec temps dépassé	
Réservoir	Absence seuil min dans le réservoir	Blocage vanne ouva fermée Blocage vanne sécurité ouverte Fuite Réservoir Fuite conduite Rés-DosMel Fuite conduite Rés-vanne sécurité Bouchage conduite Rés-Vanne Ouva	arrêt remplissage	arrêt système préparation	Out.AlarmeMinRes = 0	

Tableau 4.1 : Analyse du système par l'AMDE (suite)

	Atteinte seuil max dans le réservoir	Blocage vanne ouva ouverte Bouchage conduite Rés-Vanne Sécurité Bouchage conduite Rés-DoseurMel	arrêt remplissage	arrêt système préparation	Out.AlarmeMaxRes = 1	
Vanne Sécurité	Refus Ouverture	Défaillance mécanique (DM) Défaillance électrique (DE) Erreur Opérateur	Arrêt remplissage Problème évacuation	arrêt système préparation Perturbation niveau cuve	Out.VanSécurité <> Cd.VanSécurité	
	Refus Fermeture	Défaillance mécanique (DM) Défaillance électrique (DE) Erreur Opérateur	Arrêt remplissage Problème évacuation	arrêt système préparation Perturbation niveau cuve	Out.VanSécurité <> Cd.VanSécurité	
	Réponse intempestive	Défaillance mécanique (DM) Défaillance électrique (DE) Erreur Opérateur	Arrêt remplissage Problème évacuation	arrêt système préparation Perturbation niveau cuve	Out.AlarmeMaxRes= 1 et Out.VanSécurité = 0 Out.AlarmeMinRes= 0 et Out.VanSécurité =1	
Capteur Température Mel	Signal intempestif de haute température	Dérive capteur	défaillance capteur température Mel	arrêt système remplissage	Out.tempMel > 35°C	
	Défaillance à la sollicitation	Défaut capteur	défaillance capteur température Mel	arrêt système remplissage	Out.tempMel > 35°C	

II.1.2.2- Elaboration de l'arbre des défaillances (AdD)

Rappelons que la méthode de l'arbre des défaillances (AdD) suppose les événements indésirables du système à étudier comme étant connus, et procède à une analyse plus fine de ces événements en précisant leurs causes éventuelles et ce, en se référant aux résultats de l'analyse AMDE menée précédemment.

Dans le cas de notre application, l'événement redouté principal du système est "l'arrêt de son fonctionnement". Cet événement peut être dû soit à l'arrêt du sous-système de préparation (S1), soit à l'arrêt du sous-système de remplissage (S2).

L'idée est donc d'analyser de près chacun de ces deux événements indésirables :
- (E1) : arrêt du sous-système de préparation (S1)
- (E2) : arrêt du sous-système de remplissage (S2)

Chacun de ces 2 événements (E1 et E2), renvoie à un état de dysfonctionnement relatif (D1 et D2), que l'opérateur devrait restituer. Les tâches de correction relatives à ces deux états de dysfonctionnement seront analysées et définies dans l'étape suivante.

L'élaboration de l'AdD permet à ce niveau, de venir en aide au diagnostic des causes éventuelles. Les figures 4.4, 4.5, 4.6, 4.7, 4.8 et 4.9 ci-après présentent le résultat du développement de l'arbre des défaillances relatif à l'événement redouté du système étudié : "arrêt système".

L'AdD permet d'identifier les combinaisons possibles d'événements élémentaires qui causent ces dysfonctionnements. Ces événements élémentaires reviennent, en fait, à des défaillances des composants matériels du système. Les principales défaillances discernées par l'AdD sont les suivantes :

- problème de dosage de la solution primaire P,
- problème de dosage du produit S,
- perturbation du niveau du réservoir,
- perturbation du niveau de la cuve,
- problème d'évacuation de la solution du réservoir,
- problème de remplissage du godet,
- arrêt du moteur,
- défaillances des capteurs (de température et de volume),
- problèmes au niveau des vannes (vanne cuve et vanne sécurité)

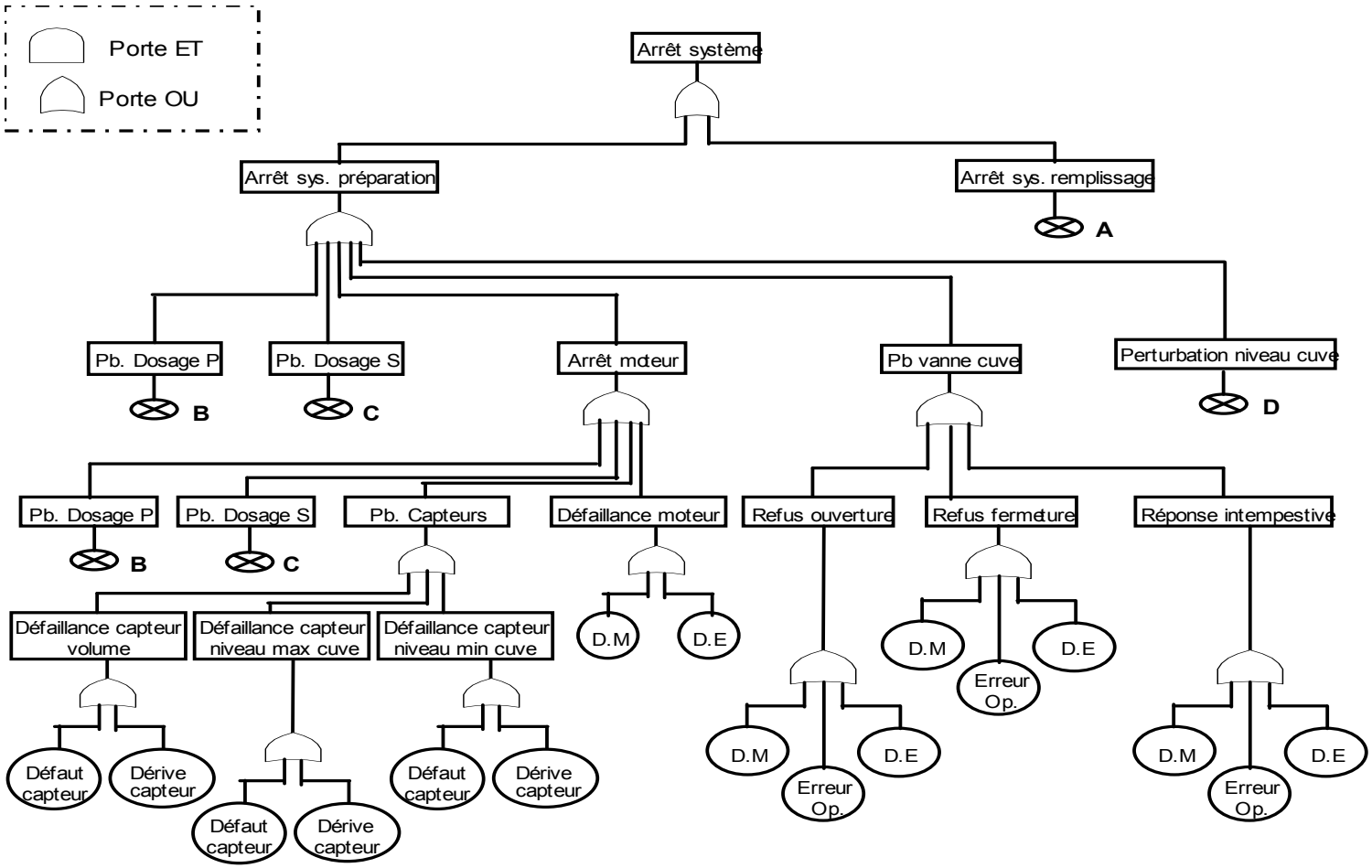


Figure 4.4: *Elaboration de l'AdD pour l'événement redouté 'arrêt système'*

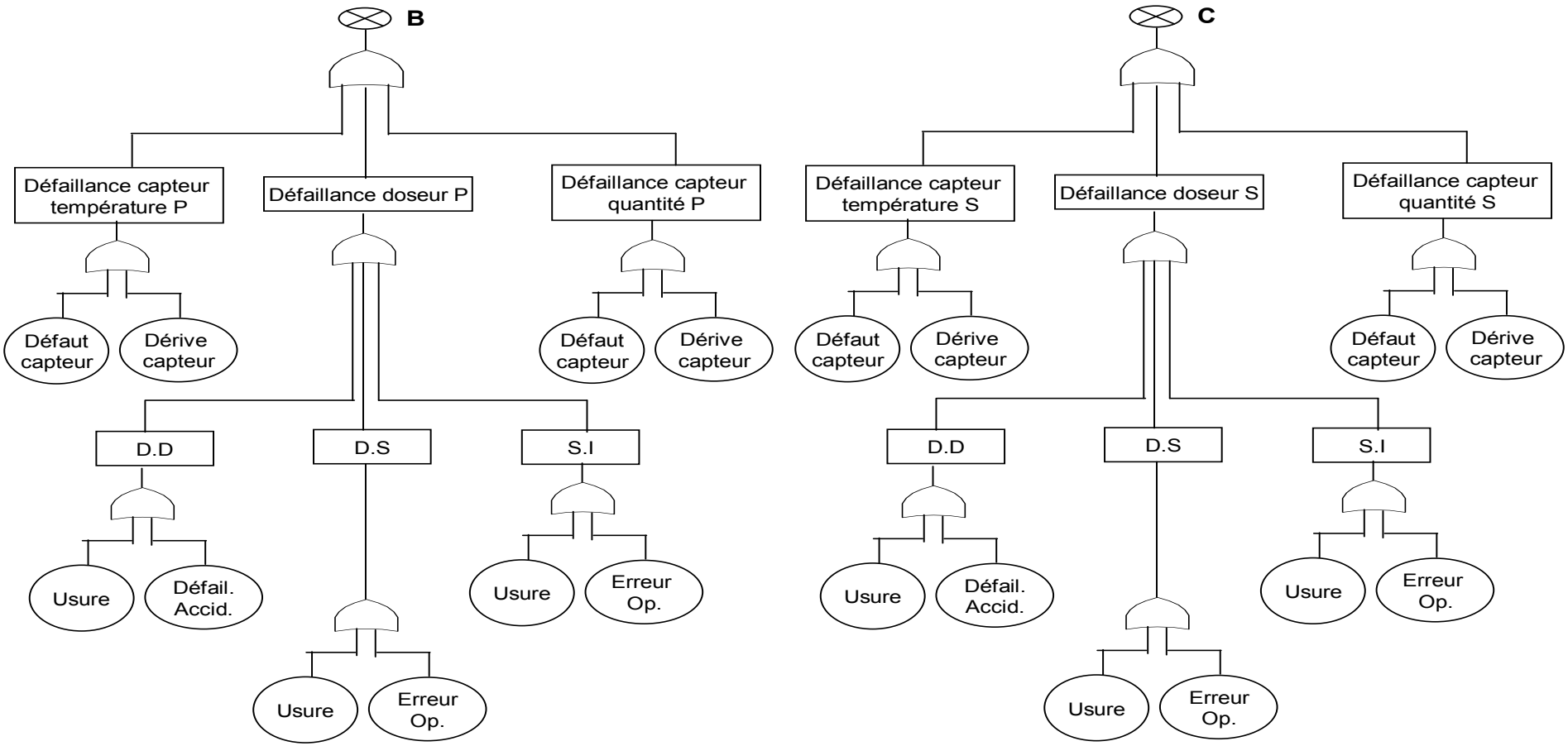


Figure 4.5: *Développement de l'AdD pour les deux événements: 'Pb Dosage P' et 'Pb Dosage S'*

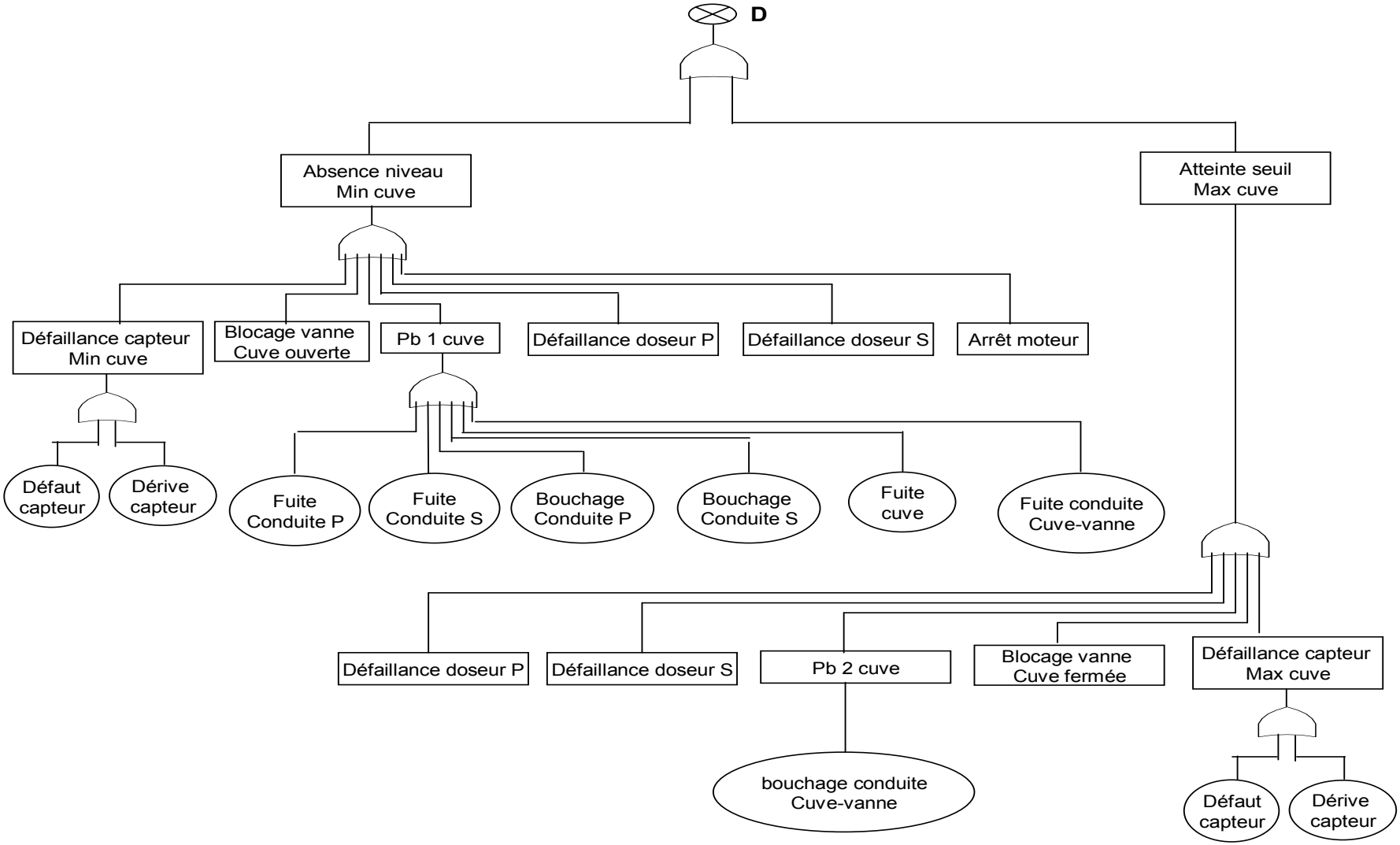


Figure 4.6: *Développement de l'AdD pour l'événement: 'Perturbation niveau cuve'*

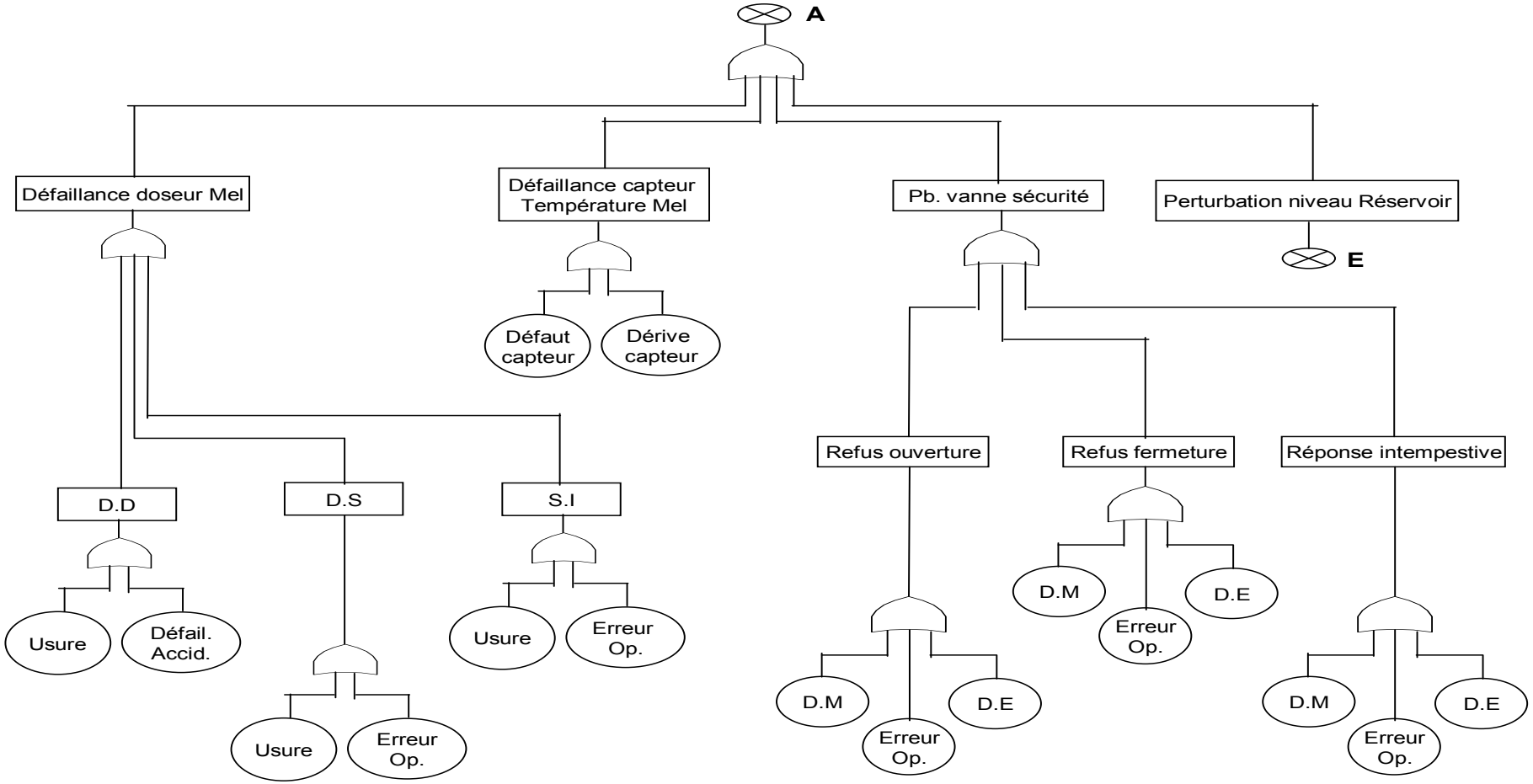


Figure 4.7: *Développement de l'AdD pour l'événement: 'arrêt système remplissage'*

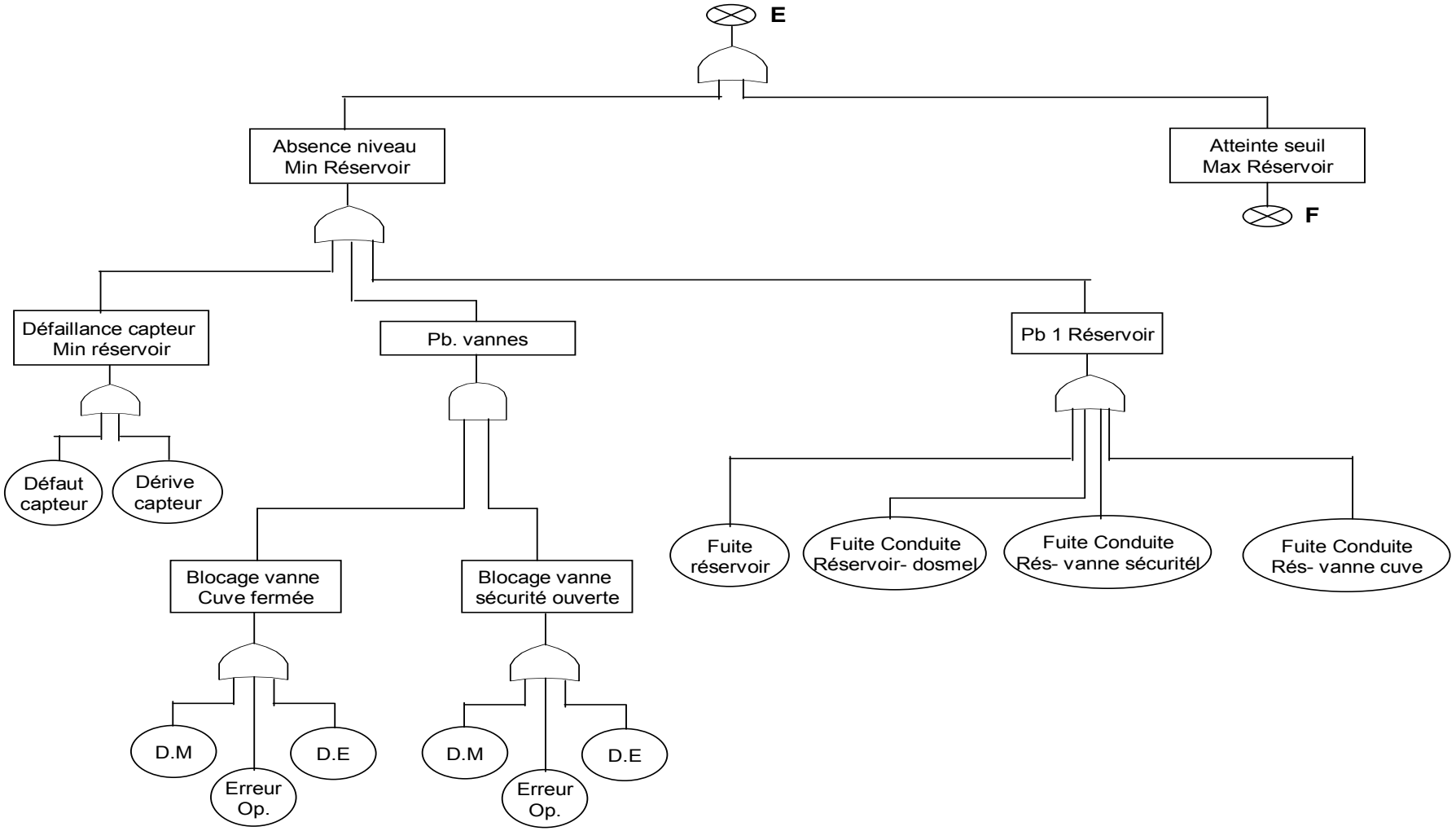


Figure 4.8: *Développement de l'AdD pour l'événement: 'Perturbation niveau réservoir'*

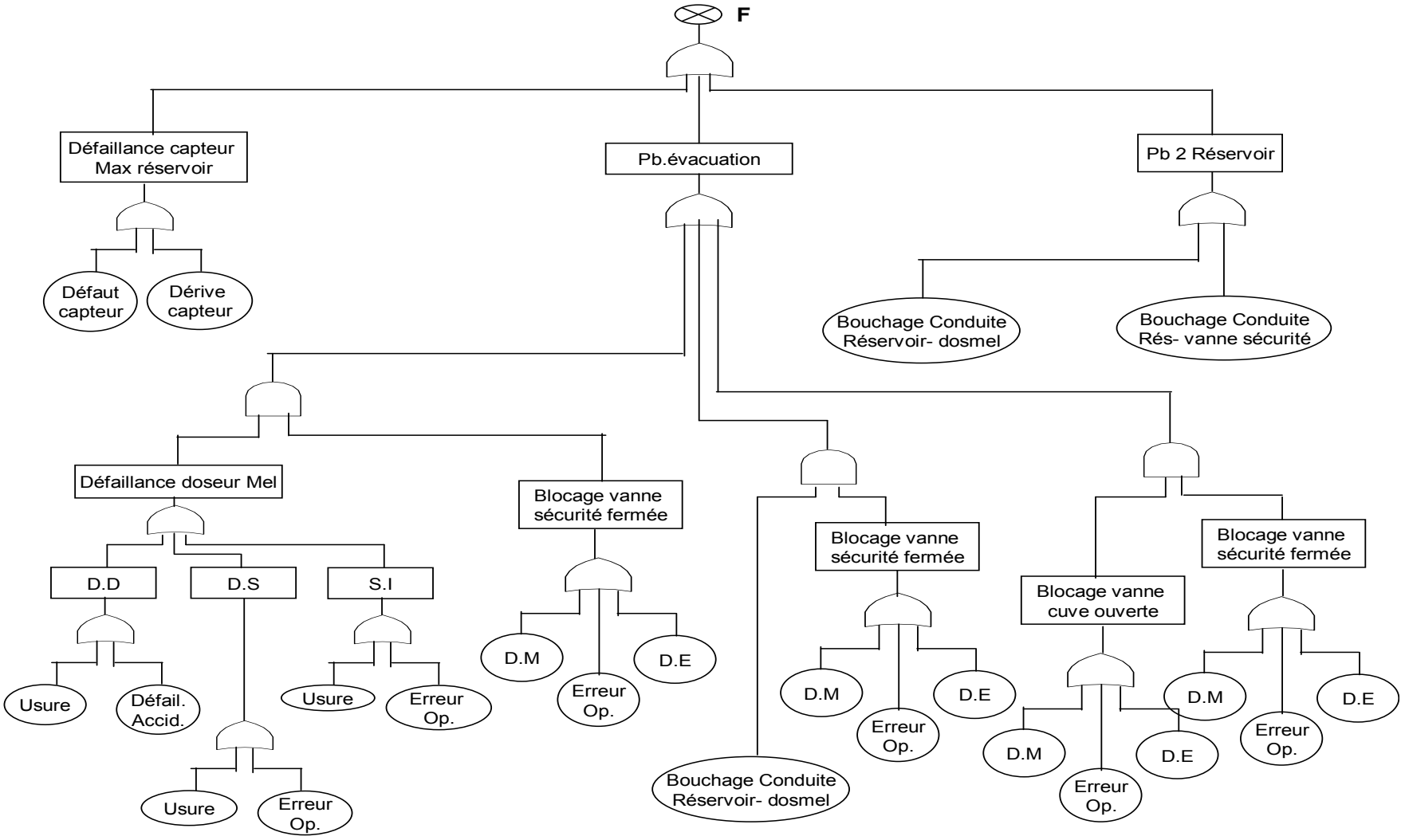


Figure 4.9: *Développement de l'AdD pour l'événement: 'atteinte seuil max réservoir'*

II.1.3- Analyse de la tâche opérateur

Une fois le fonctionnement du système étudié, les dysfonctionnements possibles analysés, l'étape qui suit consiste à analyser la tâche de l'opérateur humain face à ces différents états de fonctionnement du système (fonctionnement normal et fonctionnements anormaux).

En cas de fonctionnement normal, l'opérateur a besoin de savoir que tout marche bien et de surveiller les dérives éventuelles. Cet état de bon fonctionnement peut être reflété au niveau de l'interface par une représentation d'un ensemble de variables pertinentes synthétisant l'état courant du système.

Dans la réalité, les variables représentatives de l'état de bon fonctionnement d'un procédé sont identifiées en relation avec les experts de celui-ci [Taborin, 89]. Pour notre exemple d'application on considère que, l'état de fonctionnement normal peut être représenté par l'ensemble des informations suivantes :

- le volume courant dans la cuve avec les niveaux min et max (Out.VolCuve, Cs.MinCuve, Cs.MaxCuve),
- l'état du moteur (Out.Moteur),
- le volume courant dans le réservoir avec les niveaux min et max (Out.VolRes, Cs.MinRes, Cs.MaxRes),
- la température de la solution mélange dans le réservoir (Out.TempMel).

Cet ensemble de variables constitue les besoins informationnels de l'opérateur (BIO) en cas de fonctionnement normal.

En cas de dysfonctionnement, l'opérateur doit être averti du mauvais fonctionnement par des signaux d'alarme et un ensemble d'informations en termes de variables représentatifs au niveau de l'interface résumant la situation en cours. L'opérateur doit alors réagir le plus rapidement possible et intervenir sur un certain nombre de variables de commande, exécutant sa tâche de correction pour ramener le système en état de fonctionnement normal.

Cet ensemble de signaux d'alarme, de variables d'information et de variables de commande constitue les BIO pour la situation courante et dépendra du cas du dysfonctionnement en question et de la tâche de correction de l'opérateur. L'ensemble de BIO, pour le cas d'une telle situation, ne pourra donc être défini qu'après avoir mené une analyse détaillée de la tâche opérateur.

Une analyse préliminaire pour notre cas d'application révèle deux tâches principales de l'opérateur humain :

- une tâche de surveillance en cas de fonctionnement normal, et
- une tâche d'analyse et de reprise en cas de dysfonctionnement du système.

Cette dernière tâche doit être détaillée relativement à chaque événement 'élémentaire' mis en évidence par l'arbre des défaillances (voir ci-dessus). Pour chacun de ces événements élémentaires, l'opérateur doit accomplir une suite d'actions sur les composantes matérielles du système afin de restituer son bon état de fonctionnement.

Cette étude a pour objectif de construire un modèle de la tâche opérateur en termes d'actions élémentaires avec toutes les combinaisons possibles : parallèles, séquentielles, alternatives et conditionnelles (voir chapitre 3).

Les actions de l'opérateur sont déduites à partir du tableau AMDE et de l'arbre AdD. La manière de déduire la modélisation de l'interaction homme-machine à partir de cette analyse est expliquée dans le paragraphe suivant.

II.2- Modélisation de l'interaction homme-machine par les RdP

Une fois l'analyse des dysfonctionnements possibles du système effectuée, l'arbre des défaillances élaboré et les tâches opérateur de correction analysées, l'étape suivante de la démarche consiste à élaborer un modèle de l'interaction homme-machine en fonction de l'évolution de l'état du système et des interventions de l'opérateur humain afin de pouvoir déduire par la suite l'ensemble des BIO relatifs à chaque état.

Cette modélisation est basée sur une composition de structures élémentaires de RdP (voir chapitre 3) et peut être déduite à partir de l'AdD et du modèle de la tâche opérateur.

L'algorithme de passage de l'AdD et du modèle de la tâche opérateur au modèle RdP, est décrit ci-dessous.

Algorithme *Déduire-modèle-RdP* : /* pour un dysfonctionnement donné */

Début :

Etape 1 : parcourir l'arbre jusqu'à localiser le nœud $n_{i,j}$ correspondant à l'événement redouté relatif à cet état de dysfonctionnement

Etape 2 : Traduire la sous-arborescence ayant pour racine le nœud $n_{i,j}$ en un RdP conformément aux règles suivantes :

1- une branche OU est traduite par une composition choix inclusive de RdP *(composition ou inclusif)*
2- une branche ET est traduite par une composition parallèle de RdP
3- une branche OU exclusif est traduite par une composition alternative de RdP

Etape 3 : Formuler les conditions au niveau des différentes transitions du RdP, en fonction des formules de détection associées aux feuilles de la sous-arborescence (appel à la fonction formuler-conditions(nœud $n_{i,j}$))

Fin

Fonction formuler-conditions(nœud $n_{i,j}$)

Début

Si nœud $n_{i,j}$ = feuille
Alors formuler-conditions ← formule de détection
/ cette condition sera associée à la transition en entrée de la structure élémentaire relative à l'action opérateur de correction de l'événement indésirable élémentaire */*
Sinon
Selon (branche(nœud $n_{i,j}$)) ***faire***
Cas ET : */* composition parallèle */*

formuler-conditions ← ∧(formuler-conditions (nœud $n_{i+1, k}$)
pour k = 1 ….. L
(L étant la largeur de la branche)
/ la condition sera associée à la transition en entrée de la composition parallèle */*

Cas OU Inclusif : /* composition ou inclusif */
Pour k **de** 1 **à** L ***faire***
c_k ← formuler-conditions (nœud $n_{i+1, k}$)
FinPour
/ chaque condition c_k sera associée à une des 2 transitions sorties de P_k et sa négation sera associée à l'autre transition */*
formuler-conditions ← true

Cas Ou Exclusif : ***/****composition alternative */*
Pour k **de** 1 **à** L ***faire***
c_k ← formuler-conditions (nœud $n_{i+1, k}$)
FinPour

Pour k **de** 1 **à** L ***faire***
Affecter comme condition à transition T_k la conjonction :
$\wedge(\overline{c_i} : i \neq k) \wedge c_k$
FinPour
formuler-conditions ← true

FinSelon
FinSi
Fin

Le modèle de l'interaction homme-machine déduit pour le cas de cette application est présenté dans la figure 4.10.

La place P0 modélise un état de supervision du système en état de fonctionnement normal.La place P1 modélise l'état de dysfonctionnement : "atteinte du seuil max du réservoir" et P2 modélise l'état de dysfonctionnement "température non conforme"

Les transitions T1 et T2 modélisent donc la détection de ces dysfonctionnements moyennant les conditions qui leur sont affectées (c01 et c02) :

- La condition c01 est "CS.maxRes =1"
- La condition c02 est "Out.tempMel > 35°C"

Le bloc SRP2 représente le sous réseau qui modélise l'interaction homme-machine face au dysfonctionnement 2.

L'autre partie du réseau développée à partir de la transition T3 modélise l'interaction pour le cas du dysfonctionnement 1. Ce bloc est déduit de l'AdD conformément à l'algorithme de passage expliqué ci-dessus. Il est construit à partir d'une composition "ou inclusive" de trois sous réseaux, chacun d'eux modélisant l'interaction face à l'une des causes possibles de cet état de dysfonctionnement.

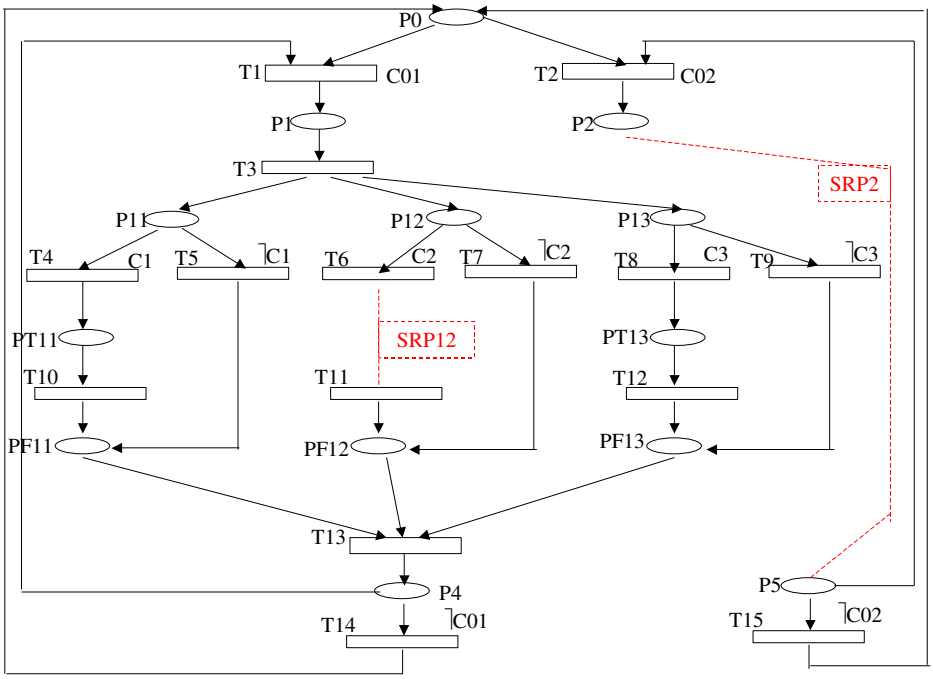


Figure 4.10 : modèle de l'interaction homme-machine déduit de l'AdD

A partir de l'AdD, nous pouvons distinguer trois causes possibles du dysfonctionnement "atteinte seuil max du réservoir" :

- une défaillance du capteur de mesure,
- un problème d'évacuation,
- un problème de bouchage de conduites du réservoir.

Ces trois causes possibles sont modélisées respectivement par les places P11, P12 et P13.

Pour les première et troisième causes, les analyses effectuées par l'AdD renvoient à des événements élémentaires et par la suite à des actions élémentaires de l'opérateur pour changer le capteur ou changer les conduites bouchées. Chacune des ces actions est modélisée par une structure élémentaire (bloc [T4, PT11 et T10] pour la première action et bloc [T8, PT13 et T12] pour la deuxième action).

Pour la cause "problème d'évacuation" modélisée par la place P12, c'est un événement composé dû à :

- une défaillance du doseur du mélange avec un blocage de la vanne sécurité en état fermée, ou
- un bouchage de la conduite réservoir-doseur mélange avec un blocage de la vanne sécurité en état fermée, ou
- un blocage de la vanne cuve en état ouverte avec un blocage de la vanne sécurité en état fermée.

Le sous réseau SRP12 (figure 4.11) modélise cette éventualité. Pour chacun de ces trois cas possibles, une action élémentaire de l'opérateur est nécessaire. Elle consisterait à débloquer la vanne de sécurité de l'état fermée et changer le doseur, ou déboucher la conduite réservoir-doseur ou la changer ou alors débloquer la vanne cuve de l'état ouverte.

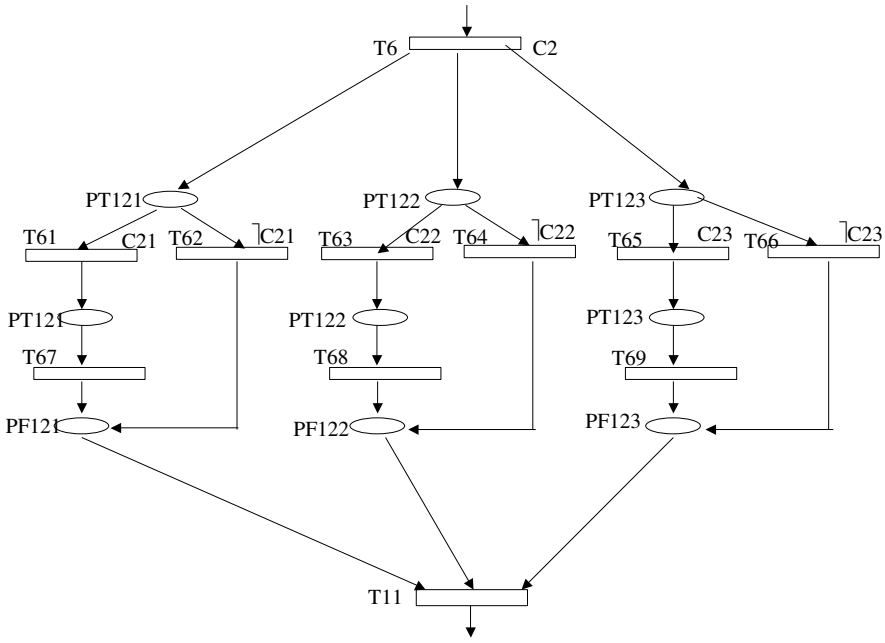


Figure 4.11 : le sous réseau SRP12

II.3- Déduction des BIO

L'étape de déduction des BIO consiste à associer au niveau de chaque état les variables d'informations nécessaires pour le représenter à l'opérateur, et éventuellement les variables de commande avec lesquelles l'opérateur pourrait accomplir ses tâches de correction.

Considérons donc principalement les places PT11, PT13, PT121, PT122 et PT123, qui représentent des états relatifs à la présence des événements particuliers en attente de l'accomplissement de ses tâches de correction. La liste des BIO associés à chacun de ces états est fournie ci-dessous :

- Pour la place PT11, l'opérateur a besoin de savoir les seuils min et max et le volume actuel ; ceci est décrit par l'ensemble des variables : x, n, Cs.MaxRes, Out.VolRes. Il aura à changer le capteur.

- Pour la place PT13, l'opérateur a besoin de savoir les seuils min et max, le volume actuel et les états des vannes. Ceci est décrit par l'ensemble des variables : x, n, Cs.MaxRes, Out.VolRes. Il aura à changer le capteur.

- Pour la place PT121, l'opérateur a besoin de savoir les seuils min et max et le volume actuel, mais aussi les états des vannes et la quantité dosée et il aura à agir sur les variables de commande de dosage et d'ouverture de la vanne réservoir. Ceci est décrit par l'ensemble des variables : x, n, Cs.MaxRes, Out.VolRes, Out.VanSecurite, Out.VanCuve, Out.QteMel, Cd.DebDosMel, Cd.OuvVanRes.

- Pour la place PT123, l'ensemble des BIO est décrit par les variables : x, n, Cs.MaxRes, Out.VolRes, Out.VanSecurite, Cd.OuvVanRes.

- Pour la place PT122, l'ensemble des BIO est décrit par les variables : x, n, Cs.MaxRes, Out.VolRes, Out.VanSecurite, Cd.OuvVanRes, Cd.OuvVanCuve.

II.4- Spécification de l'interface

Une fois, les BIO déduits, l'étape qui suit consiste à spécifier l'interface proprement dite. Cette tâche consiste à commencer par faire un découpage en vues graphiques puis à constituer ces vues à partir des BIO relatifs. Comme expliqué au niveau du chapitre 3, ce découpage en vues et association des BIO à des objets graphiques se fait conformément à des règles ergonomiques de présentation formalisées et stockées dans un système à base de connaissances.

Pour notre cas d'étude, le système pourrait en effet associer trois principales vues :
- une vue de supervision en état de fonctionnement normal,
- une vue de contrôle et commande pour l'état de dysfonctionnement 1, et
- une vue de contrôle et commande pour l'état de dysfonctionnement 2.

Pour chacune de ces vues, on y trouvera des objets graphiques représentant les BIO associés aux états du modèle de l'interaction couverts par ces vues.

Par exemple pour tout ce qui est information et consigne, le système pourrait associer des zones de texte et/ou des barre graphes principalement pour les informations sur le volume du réservoir. Pour les variables de commande des boutons poussoirs pourraient être attribués.

II.5- Vérification des propriétés de l'interface

Les bonnes propriétés de l'interface telle que l'absence de blocage, l'atteignabilité des services, la stabilité et la non ambiguïté sont vérifiées et garanties par l'approche suivie pour construction du modèle de l'interaction homme-machine (voir chapitre 3).

La dernière étape de la démarche consiste à la génération effective de l'interface et la gestion automatique du dialogue homme-machine. Cette étape détaillée au niveau du chapitre suivant.

Conclusion

L'objectif de ce chapitre était de montrer la faisabilité de l'approche méthodologique, proposée pour la spécification et la conception des IHM, à travers une étude de cas d'une application industrielle simplifiée.

La première partie du chapitre a été consacrée à la présentation de l'application industrielle et ses spécifications fonctionnelles et techniques. Par la suite, les résultats de l'application des différentes étapes de la démarche, depuis l'analyse du SHM jusqu'à la génération graphique, sont présentés.

L'outil informatique proposé pour supporter l'approche proposée de conception et de génération semi-automatique des IHM est baptisé Ergo-Conceptor+. Les détails de conception et réalisation de cet outil sont présentés dans le chapitre suivant.

Chapitre 5

Conception et réalisation de l'outil Ergo-Conceptor+ supportant la génération de l'IHM et perspectives de recherche

Plan du chapitre

Introduction
I. Architecture conceptuelle et fonctionnelle de l'outil Ergo-Conceptor+
- I.1- Rappel de l'architecture informatique de l'outil Ergo-Conceptor
- I.2- Présentation de l'outil Ergo-Conceptor+
- I.3- Ergo-Conceptor versus Ergo-Conceptor+

II. Conception détaillée de l'outil Ergo-Conceptor+
- II.1- Description du procédé
- II.2- Définition du modèle de l'interaction homme-machine
- II.3- Génération graphique de l'IHM
- II.4- Simulation et gestion de la dynamique du dialogue homme-machine
- II.5- Extrait de la modélisation objet avec UML
- II.6- Conclusion sur l'outil Ergo-Conceptor+

III. Perspectives de recherche
- III.1- Intégration de l'approche multi-agents dans la démarche de spécification et de génération des IHM
- III.2- Couplage d'un système de prédiction et d'aide à la conduite
- III.3- Prise en considération de l'aspect profil utilisateur

Conclusion

Ce chapitre est consacré à la présentation de l'outil Ergo-Conceptor+ supportant l'approche à base de modèle,s proposée dans le cadre de ce travail, pour la conception et la génération semi-automatique d'une partie du système interactif. Nous commençons par positionner, dans un premier temps, l'outil Ergo-Conceptor+ par rapport à l'outil Ergo-Conceptor étudié et développé lors de recherches antérieures. Nous proposons, par la suite, une description de la conception détaillée des différents modules de l'outil Ergo-Conceptor+.

Nous achevons le chapitre par la présentation d'un ensemble de perspectives de recherches et développements futurs à entreprendre sur ce travail.

I. Architecture conceptuelle et fonctionnelle de l'outil Ergo-Conceptor+

La dénomination de l'outil Ergo-Conceptor+ vient du fait qu'il présente une continuité des travaux de recherche de notre équipe. Il est basé sur l'outil Ergo-Conceptor, maquetté initialement dans le cadre de la thèse de Faouzi Moussa (déjà positionné au niveau du chapitre 1). Nous proposons de ce fait de commencer par rappeler l'architecture fonctionnelle de ce système. Nous présentons, par la suite, l'architecture fonctionnelle de Ergo-Conceptor+. Une analyse comparative de ces deux outils sera faite.

I.1- Rappel de l'architecture informatique de l'outil Ergo-Conceptor

Rappelons que l'outil Ergo-Conceptor se base sur démarche de conception ergonomique des IHM dédiées à la supervision des procédés industriels. Il a pour objectif de fournir des spécifications de l'imagerie incluant des facteurs ergonomiques, utiles à la génération des vues [Moussa, 92][Moussa et al., 00] (voir chapitre 1, §III.7).

Cet outil se compose de trois modules interconnectés (figure 5.1) :

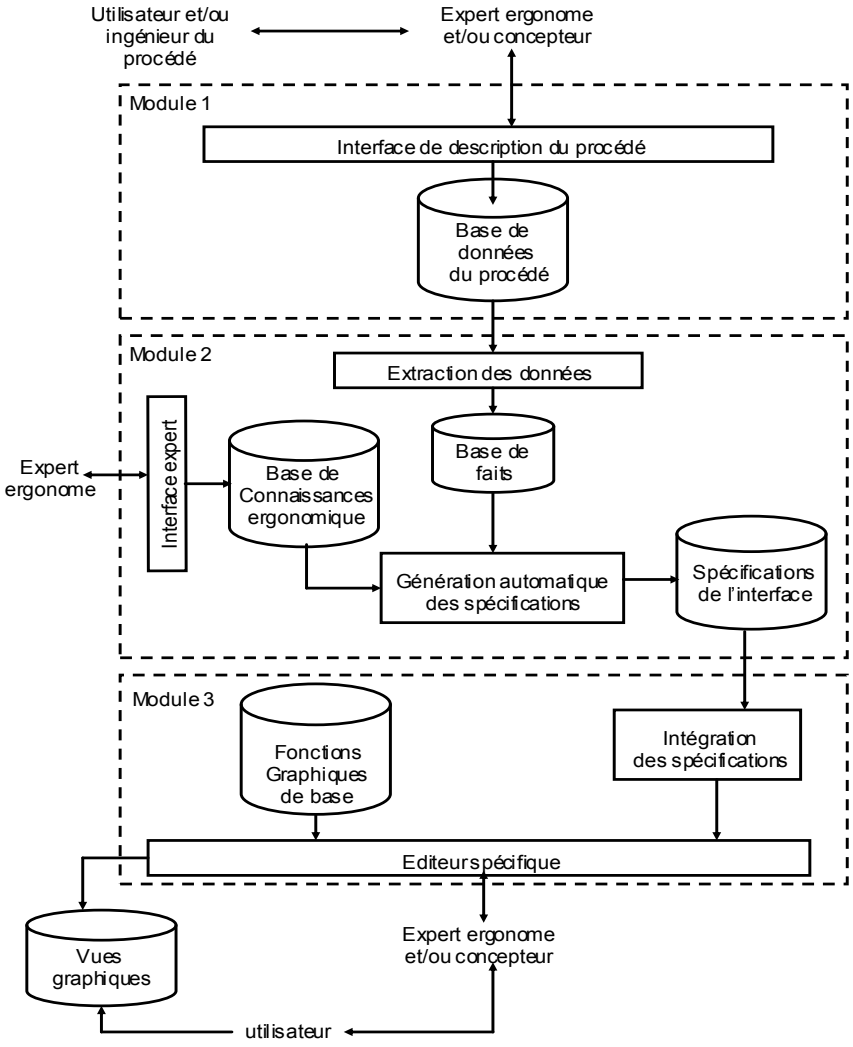


Figure 5.1 : Architecture Informatique du système Ergo-Conceptor [Moussa et al., 00]

- Le premier module sert à décrire un procédé selon une méthodologie de description de procédés guidée par les besoins informationnels des opérateurs, permettant de décrire les

variables du système, les groupements fonctionnels simples (GFS) et les groupements fonctionnels composés (GFC) et les réseaux d'influence entre les différentes variables constituant ces groupements.

- Le second module permet la génération automatique, à partir des résultats issus du modèle précédent, d'un fichier de spécifications d'interface en exploitant des connaissances ergonomiques stockées dans une base de connaissances. Cette base est alimentée à partir de l'interface expert.

- Le dernier module assure la génération des vues graphiques et ce, en proposant au concepteur une interface avec le système, lui permettant de créer ses vues. Le noyau de ce module est un générateur automatique de vues graphiques qui exploite les spécifications déduites du module précédent. L'interface assure toutefois au concepteur des degrés de liberté lui permettant d'intervenir au niveau le plus bas de la conception des vues, à l'aide de fonctions d'édition graphique « classiques ».

La principale critique faite à l'égard de l'outil Ergo-Conceptor est qu'il ne gère pas le dialogue homme-machine mais plutôt l'impact des variables de contrôle et de commande du système par rapport à l'interface. Il ne génère que des vues statiques et ne permet pas de gérer certains aspects fonctionnels importants tels que le parallélisme et la synchronisation. En plus, vu l'aspect assez informel des spécifications générées, aucune vérification ne pourrait être menée.

L'outil Ergo-Conceptor+ a pour objet justement de dépasser ces limites en adoptant une démarche plus complète de spécification et de conception d'IHM de supervision, couvrant les aspects de la dynamique de l'interface. Cet outil est maintenant présenté.

I.2- Présentation de l'outil Ergo-Conceptor+

Le système Ergo-Conceptor+ (figure 5.2) est capable de décider des vues graphiques adéquates à associer à chaque sous-système. Pour ce faire, il prend en considération, d'une part, les caractéristiques de chaque sous-système (la liste de ses différents états de fonctionnements, l'ensemble des BIO relatifs à chacun de ces états, etc.) et d'autre part, un ensemble de recommandations ergonomiques spécifiques au cas des applications industrielles, formalisées et enregistrées dans sa base de connaissances, afin d'assurer une bonne qualité ergonomique.

Une fois les différentes vues graphiques identifiées, le système aura à décider des représentations graphiques et du dialogue à associer aux différents objets de ces vues conformément aux recommandations ergonomiques. Par la suite, le système assure la gestion du composant dialogue de l'interface en manipulant les différents RdP relatifs aux structures de contrôle des objets graphiques sélectionnés et au modèle de l'interaction homme-machine.

Le système Ergo-Conceptor+ est composé principalement de quatre modules (figure 5.3) :

- Le premier module permet d'alimenter la base de données du procédé. Il a pour rôle de décrire l'ensemble des variables du système, de définir les différents sous-systèmes à contrôler et leurs dysfonctionnements possibles.

- Le second module permet au concepteur de l'interface de définir les RdPI modélisant l'interaction homme-machine et de préciser par la suite l'association des BIO aux différents états du système. Les BIO à associer seront des variables de la base des données du procédé décrite par le premier module

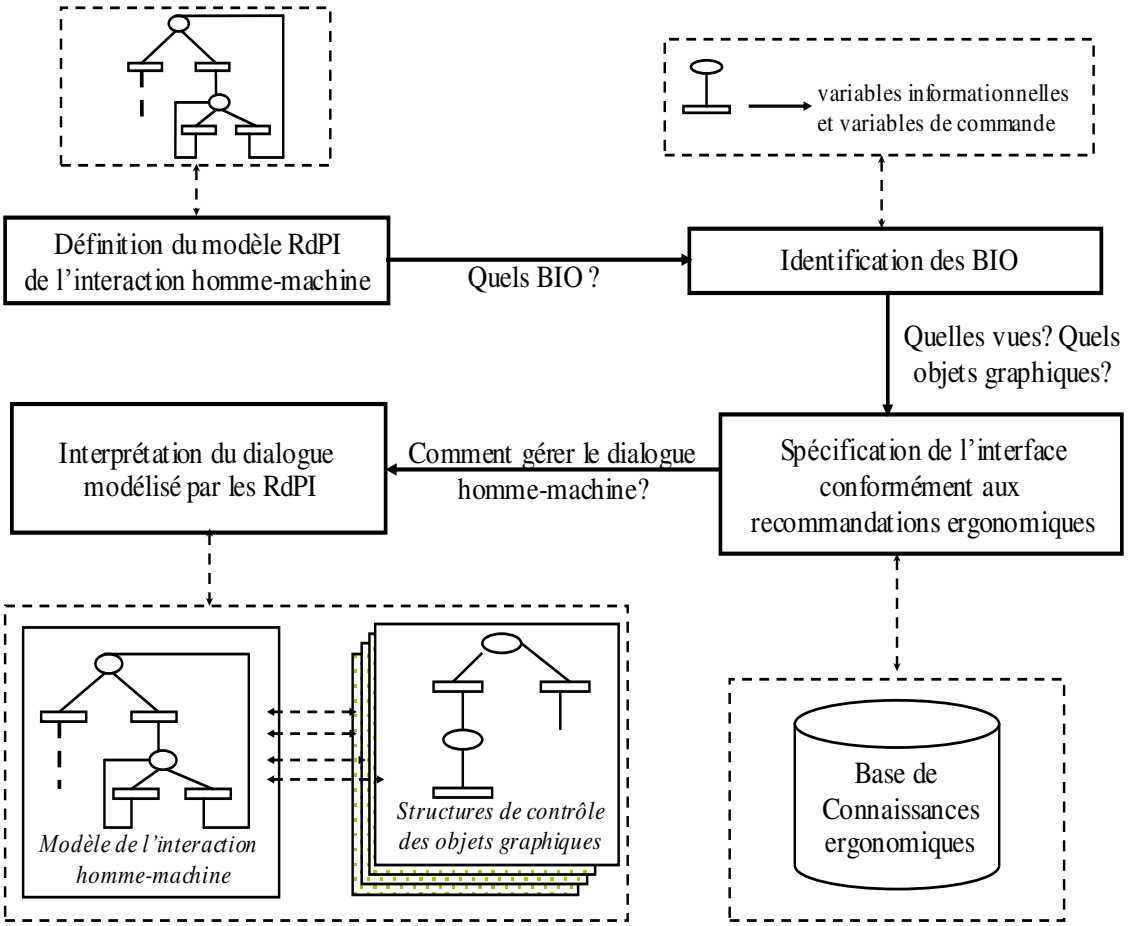


Figure 5.2 : Démarche suivie par le système Ergo-Conceptor+ [Moussa et al., 99]

- Le troisième module assure la spécification et la génération graphique de l'interface pour la supervision et le contrôle du procédé industriel. Il se base pour ce faire, sur l'ensemble des informations du procédé et du processus de contrôle décrites dans les deux modules précédents. Afin de garantir une meilleure qualité de représentation graphique, on fait appel à un système à base de connaissances manipulant des recommandations ergonomiques de différents niveaux d'abstraction (de la variable jusqu'à la structuration d'une vue complète).

- Le quatrième et dernier module du système Ergo-Conceptor+ assure la gestion de la dynamique du dialogue homme-machine et simule le changement de l'état de l'interface en fonction (i) de l'évolution de l'état du procédé et (ii) des actions de correction apportées par l'opérateur.

Tous ces modules sont présentés et détaillés au §II. Nous présentons ci-dessous un tableau récapitulatif comparant ces deux outils.

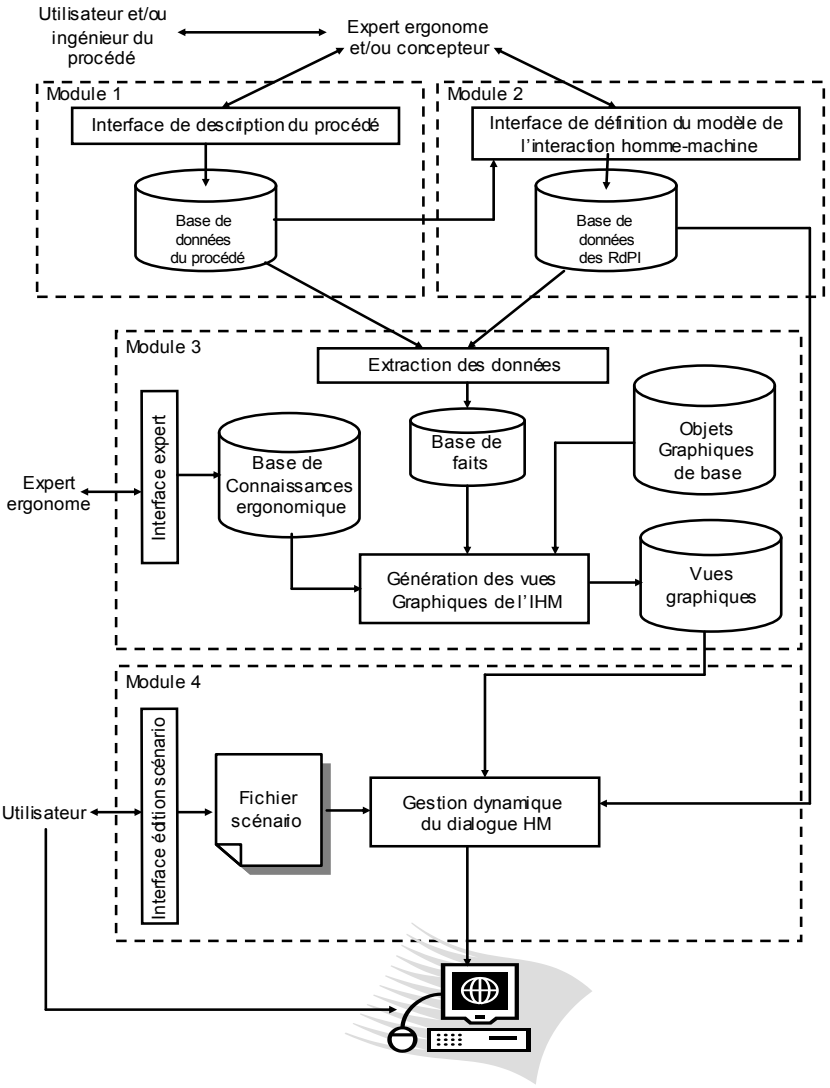


Figure 5.3 : Architecture informatique du système Ergo-Conceptor+

I.3- Ergo-Conceptor versus Ergo-Conceptor+

La principale différence entre les deux outils réside dans le fait que le premier outil, Ergo-Conceptor, ne gère que des interfaces statiques alors que le second, Ergo-Conceptor+, assure la gestion de la dynamique au niveau des interfaces générées. Le tableau 5.1 suivant récapitule les apports et les limites de chacun de ces deux outils.

On remarque en effet, que les deux outils se basent sur une description du procédé à contrôler. Cependant, chacune de ces descriptions est menée selon un point de vue différent. Pour le premier système, on décrit le procédé en termes de variables, de groupements fonctionnels et de réseaux d'influence. Alors que pour le second, on décrit l'ensemble des sous-systèmes à contrôler, leurs variables et leurs dysfonctionnements possibles. Ergo-Conceptor+ intègre un module de définition de la dynamique de l'interaction homme-machine moyennant les RdPI offrant ainsi la possibilité d'assurer une animation de l'interface et une gestion automatique de sa dynamique et surtout l'aspect de vérification de l'interface avant sa génération. Alors que le système Ergo-Conceptor s'arrête au niveau de la génération statique de l'interface. Il n'offre aucun moyen de validation de l'interface avant sa génération.

Tableau 5.1 : Tableau comparatif des deux systèmes Ergo-Conceptor et Ergo-Conceptor+

Système / Fonctionnalité	Ergo-Conceptor	Ergo-Conceptor+
Description du procédé	En termes de variables, groupements fonctionnels et réseaux d'influence (approche personnelle [Moussa, 92])	En termes de sous-systèmes, variables et dysfonctionnements (SADT, AMDE et AdD)
Définition de la dynamique de l'IHM	Non	Se basant sur un modèle RdPI
Spécification et génération graphique de l'IHM	Se basant sur un système à base de connaissances	Se basant sur un système à base de connaissances
Intervention assistée du concepteur sur les vues créées	A l'aide de fonctions d'éditions graphiques	Non encore défini dans l'état actuel des recherches
Animation et simulation de la dynamique de l'interface	Non	A l'aide d'un joueur de RdPI
Possibilité de vérification de l'interface	Non	Oui (sur base du modèle RdPI)

Pour la spécification et la génération graphique de l'interface, les deux systèmes se basent sur le même principe. Ils intègrent un système à base de connaissances assurant la prise en considération des recommandations ergonomiques de présentation. Néanmoins, Ergo-Conceptor intègre un module d'édition graphique permettant au concepteur d'intervenir d'une manière assistée sur les vues générées. Ergo-Conceptor+ ne propose pas encore un tel module. Il sera à mettre au point dans le cadre de recherches ultérieures.

II. Conception détaillée de l'outil Ergo-Conceptor+

L'outil Ergo-Conceptor+ se base principalement sur quatre modules. Ces modules sont maintenant détaillés.

II.1- Description du procédé

Le premier module de base du système Ergo-Conceptor+ permet de décrire le procédé à contrôler. Cette description se base sur les résultats des analyses en mode de fonctionnement normal et anormal élaborées au niveau des premières étapes de la démarche de conception des IHM suivie par l'outil Ergo-Conceptor+. Ainsi, la description se fait en termes de sous-systèmes, de variables associées et de dysfonctionnements possibles (figure 5.4).

Chaque sous-système est décrit par :
- son nom,
- une liste des variables le composant et
- une liste des dysfonctionnements susceptibles de l'affecter.

Une variable est définie par :
- son nom,
- son type (informationnelle ou de commande),
- sa valeur courante,
- sa valeur de consigne, et
- ses valeurs maximale et minimale tolérées.

Un dysfonctionnement quant à lui est défini par :

- son nom,
- son degré de gravité et
- la liste des variables impliquées pour la détection et pour la correction.

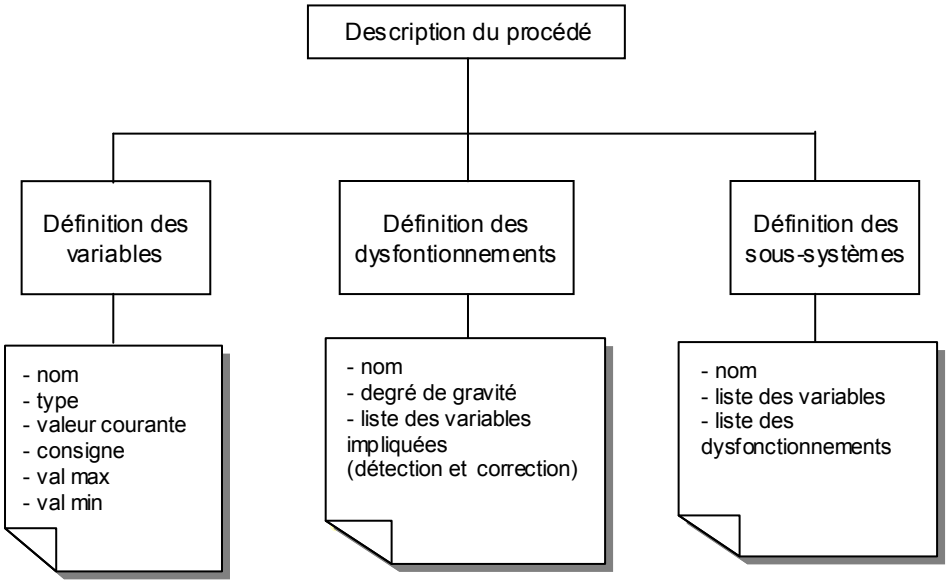


Figure 5.4 : Module de description du procédé

II.2- Définition du modèle de l'interaction homme-machine

Le second module de l'outil est destiné à la définition du modèle de l'interaction homme-machine. Il permet de définir le RdPI relatif au modèle de l'interaction et ceux correspondant aux structures de dialogue génériques des objets graphiques. La définition se fait en trois étapes (figure 5.5) :

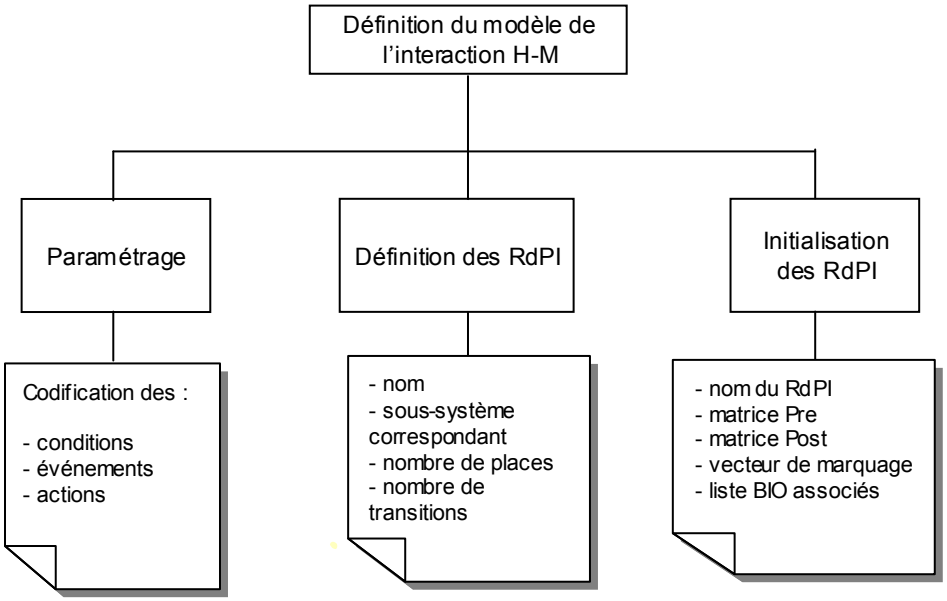


Figure 5.5 : Module de définition du modèle de l'interaction homme-machine

- La première consiste en un paramétrage en termes de codification des conditions, événements et actions à associer par la suite aux différents RdPI.
- La deuxième consiste à définir les RdPI du modèle de dialogue. Un RdPI est défini par :
  - son nom,
  - le sous-système correspondant,
  - le nombre de places,
  - le nombre de transitions.

- La troisième partie permet d'initialiser les différents RdPI en précisant pour chaque réseau :
  - sa matrice Pre,
  - sa matrice Post,
  - son vecteur de marquage, et
  - la liste des BIO associés aux différentes transitions du réseau.

La matrice Pre définit les marquages nécessaires ainsi que la condition et l'événement indispensables pour le franchissement d'une transition donnée. La matrice Post, quant à elle, précise les marques à placer dans les différentes places ainsi que les actions à accomplir au tir d'une transition donnée. Le vecteur de marquage précise le marquage des places du réseau à un instant donné.

La définition du modèle de l'interaction homme-machine se complète par l'association à chaque transition du RdP de la liste des variables représentant les besoins informationnels (BIO) relatifs à une situation donnée.

La figure 5.6 représente une image écran de l'outil permettant l'initialisation des matrices Pre et Post d'un RdPI.

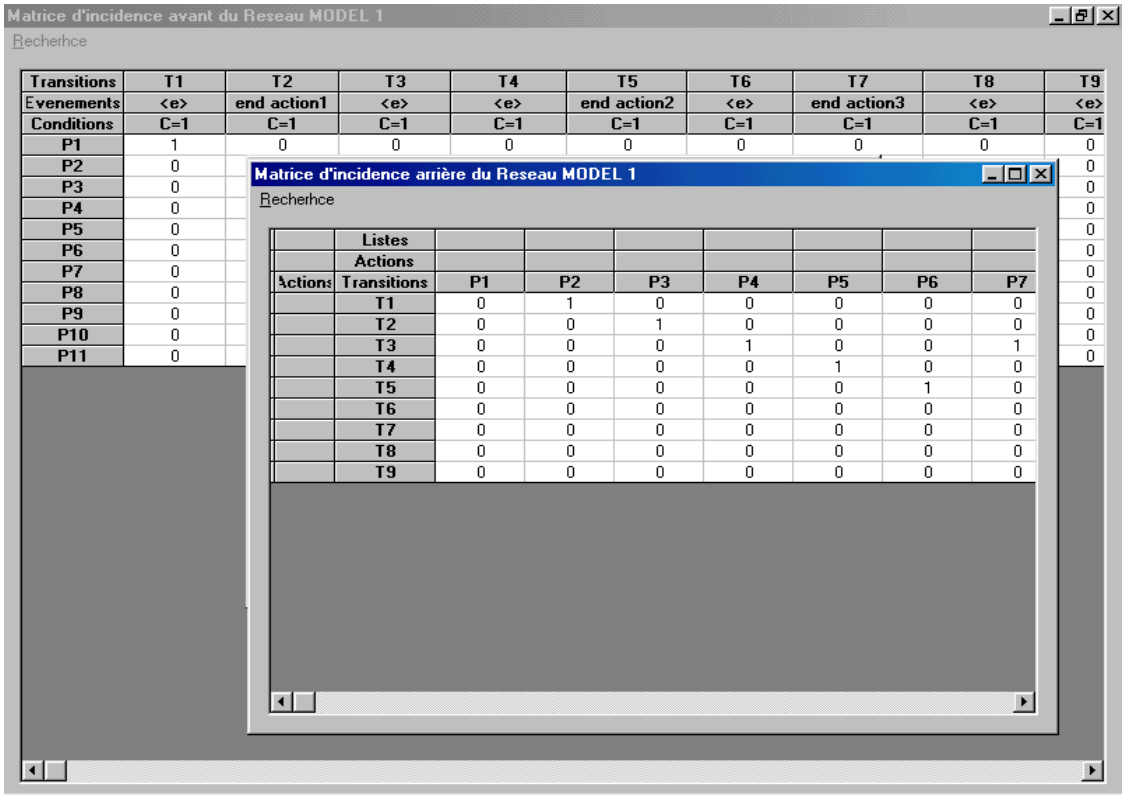


Figure 5.6 : Les matrices Pre et Post édités à l'aide de l'outil Ergo-Conceptor+

II.3- Génération graphique de l'IHM

Ce module est basé sur un système à base de connaissances comme celui intégré dans le système Ergo-Conceptor. Il s'appuie sur :

- une base de connaissances ergonomiques formalisées sous forme de règles,
- une base de faits relative à l'application contrôlée précisant l'ensemble des états possibles d'un sous-système donné, la quantité des BIO associée, le degré d'urgence, etc., et
- un moteur d'inférence de premier ordre avec chaînage avant.

Le résultat de ce module se présente sous forme de fichiers de spécification précisant l'ensemble des vues à générer et le contenu de chacune d'elle. La génération graphique se fait dans une deuxième étape grâce à une bibliothèque d'imagerie proposant un ensemble d'objets graphiques de base pour la constitution de ces vues.

Ce module se compose donc de trois sous-modules (figure 5.7) :

- un module de configuration du système expert qui consiste à définir sa base de règles ergonomiques ainsi qu'une grammaire de spécification graphique (BNF) ; cette grammaire reprend celle proposée dans [Moussa, 92], des exemples se trouvent au chapitre 3 (voir chapitre3 §II.4.1),

- un module de spécification graphique qui permet de générer le fichier des spécifications à partir d'une base de faits alimentée par les spécifications du procédé en question, et

- un module de génération graphique de l'IHM qui à partir des fichiers de spécifications générées procède à l'habillage graphique des vues.

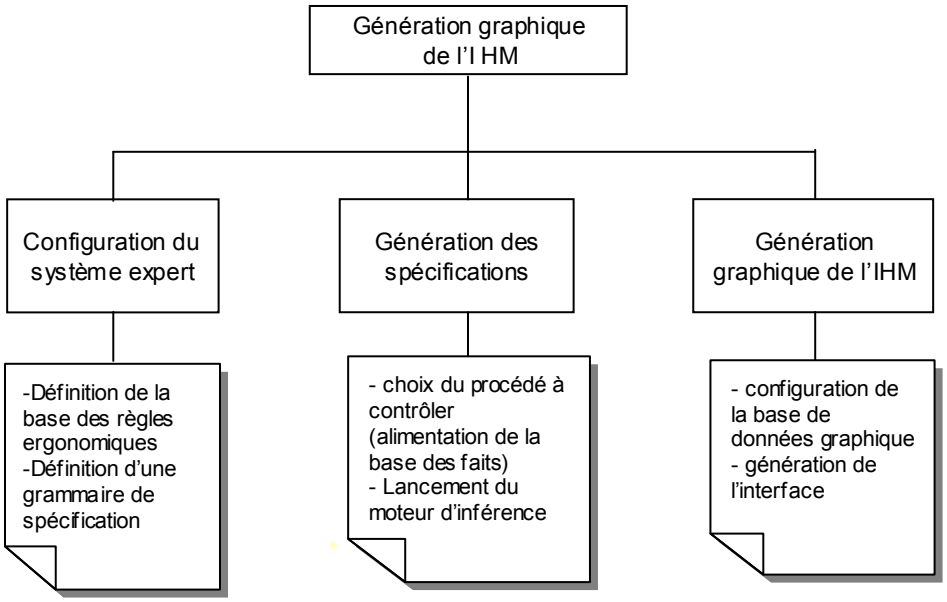


Figure 5.7 : Module de génération graphique de l'IHM

La figure 5.8 montre un écran graphique du système Ergo-Conceptor+ permettant l'initialisation de sa base de connaissance via la description des règles ergonomiques de spécification de l'interface.

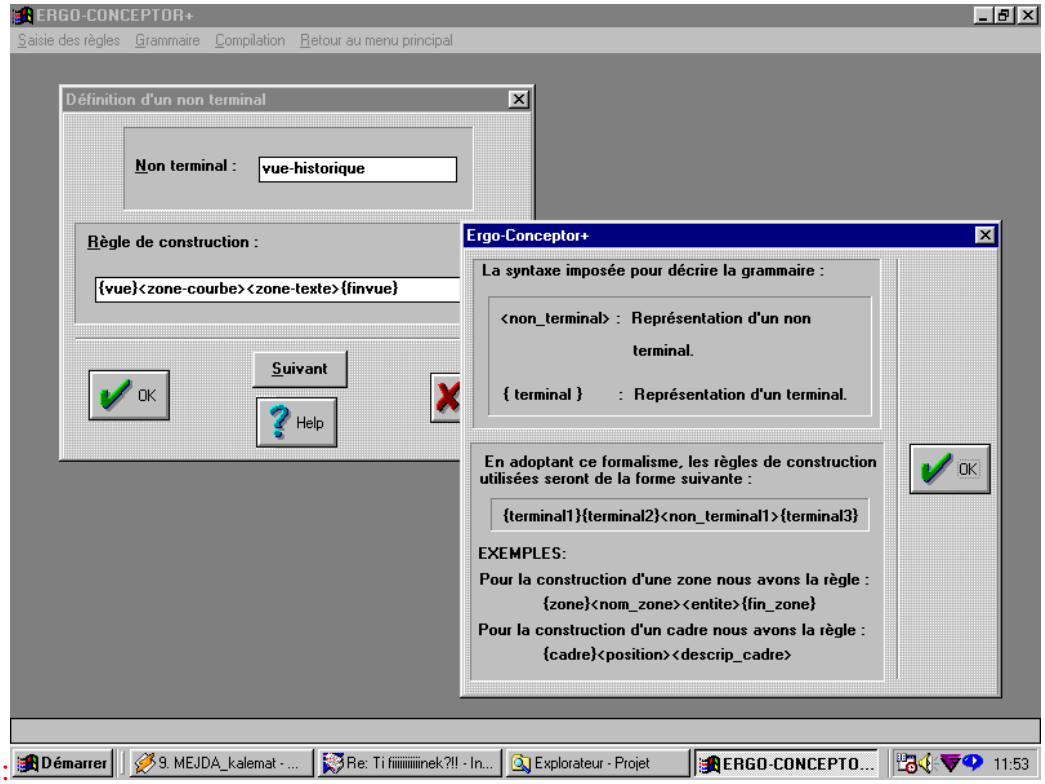


Figure 5.8 : Une image écran du système Ergo-conceptor+ pour l'initialisation de sa base de connaissance

Cette interface offre à l'utilisateur la fenêtre permettant la définition de la grammaire proposée pour la description des règles ergonomiques de présentation de l'IHM de supervision générée.

En effet, une pseudo-grammaire est proposée par l'outil Ergo-Conceptor+ pour décrire la grammaire de spécification de l'interface. Une entité cadre par exemple est définie avec la syntaxe suivante : <entité-cadre> := {entité-cadre} <position> <description>.

<tout ce qui se trouve entre crochets <> est un "non-terminal" dont on doit fournir une description et tout ce qui se trouve entre accolades {}correspond à "un terminal"décrit dans le fichier de spécification.

Ainsi, le bouton help de la première fenêtre de cette page écran (figure 5.8) propose à l'utilisateur le deuxième fenêtre précisant plus d'indications quant à la syntaxe de saisie des règles de constructions.

II.4- Gestion de la dynamique du dialogue homme-machine

Ce module assure la mise en œuvre des RdPI. Il s'appuie sur un joueur de RdPI exploitant la base regroupant les vues graphiques générée au module précédent. Cette base est exploitée avec celle des RdPI modélisant l'interaction homme-machine pour assurer la gestion de la dynamique de l'affichage. Ce module permet de (figure 5.9) :

- choisir le procédé à contrôler et le scénario à appliquer, et
- lancer le joueur de RdPI pour animer l'interface générée.

Le joueur de RdPI a pour rôle d'interpréter les différents RdPI en fonction des événements externes (actions de l'opérateur) et de l'état des variables du système. Le RdPI est considéré, en effet, comme un système particulier de règles décrivant les changements d'état du système. Le marquage du RdPI représente l'état courant du système. Le comportement de ce dernier est simulé par des tirs de transitions qui changent l'état du système.

Lors de l'occurrence d'un événement, le joueur de RdP cherche la transition à laquelle est associée cet événement et teste si elle est franchissable (c'est-à-dire le nombre des marques nécessaires existe dans les places en amont de cette transition). Si oui (la transition est franchissable), il teste alors la condition associée. Si cette dernière est vérifiée, alors il y a déclenchement du tir. Ce tir consiste à franchir la transition concernée ainsi que toutes les transitions devenues franchissables par le nouveau marquage et ce jusqu'à atteindre un état stable du réseau.

Ainsi, l'algorithme du joueur des RdPI est décrit par les trois étapes suivantes :

A. Initialisation : cette étape consiste à déterminer la liste des transitions franchissables, en fonction des événements produits, des conditions et de l'état des vecteurs de marquage de l'ensemble des RdPI.

B. Tir des transitions : dans cette étape, on procède au tir des transitions franchissables déterminées dans l'étape précédente. Le tir d'une transition consiste à :

1- mettre à jour l'état des différents vecteurs de marquage en enlevant les marques respectives des places en amont de cette transition et en produisant des marques dans les places en aval, et

2- exécuter l'action associée à cette transition

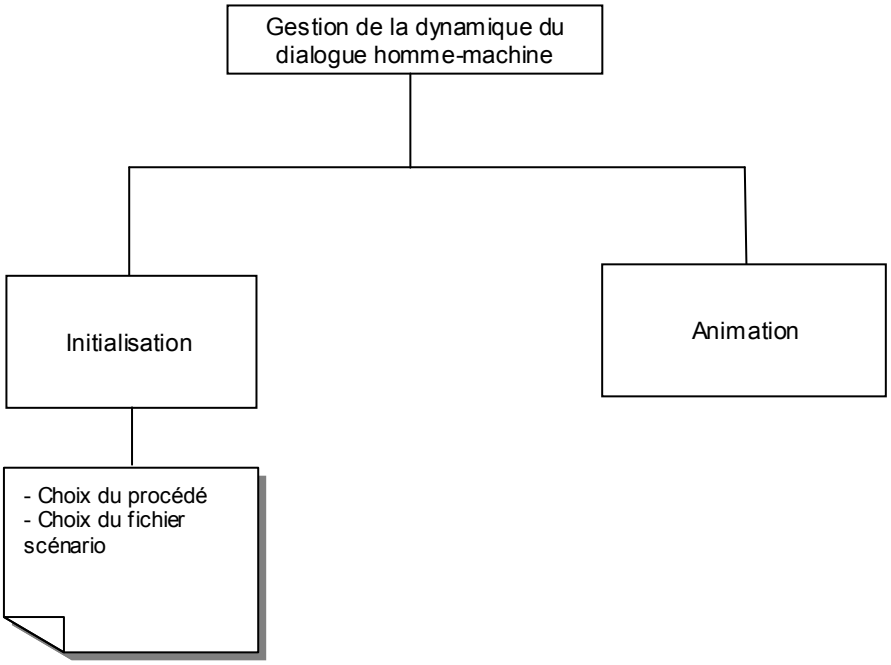


Figure 5.9 : Module de gestion de la dynamique du dialogue homme-machine

C. Echo : une fois que le tir des transitions est terminé le réseau devient instable. Pour cela, on doit tirer toutes les transitions synchronisées sur l'événement toujours présent <e>.

Les figures 5.10 et 5.11 montrent des exemples de vues générées à l'aide du système Ergo-Conceptor+ pour deux situations de fonctionnement : normal et anormal.

La première relative à une situation normale de fonctionnement montre une vue avec deux modes de représentations graphiques (figure 5.10). Une étoile représentant l'état de six variables en norme et trois barres graphes relatives aux trois variables les plus significatives de ce sous-système.

La deuxième reflète un état de dysfonctionnement du sous-système (figure 5.11). On y remarque l'apparition d'un nouveau barre graphe relatif à la variable qui a dévié de la norme avec deux boutons de commande permettant d'agir dessus. Le bouton '+' est activé et le bouton '-' est inhibé. En effet, pour la situation de dysfonctionnement en question, il est demandé plutôt d'augmenter le débit de la variable et non le diminuer.

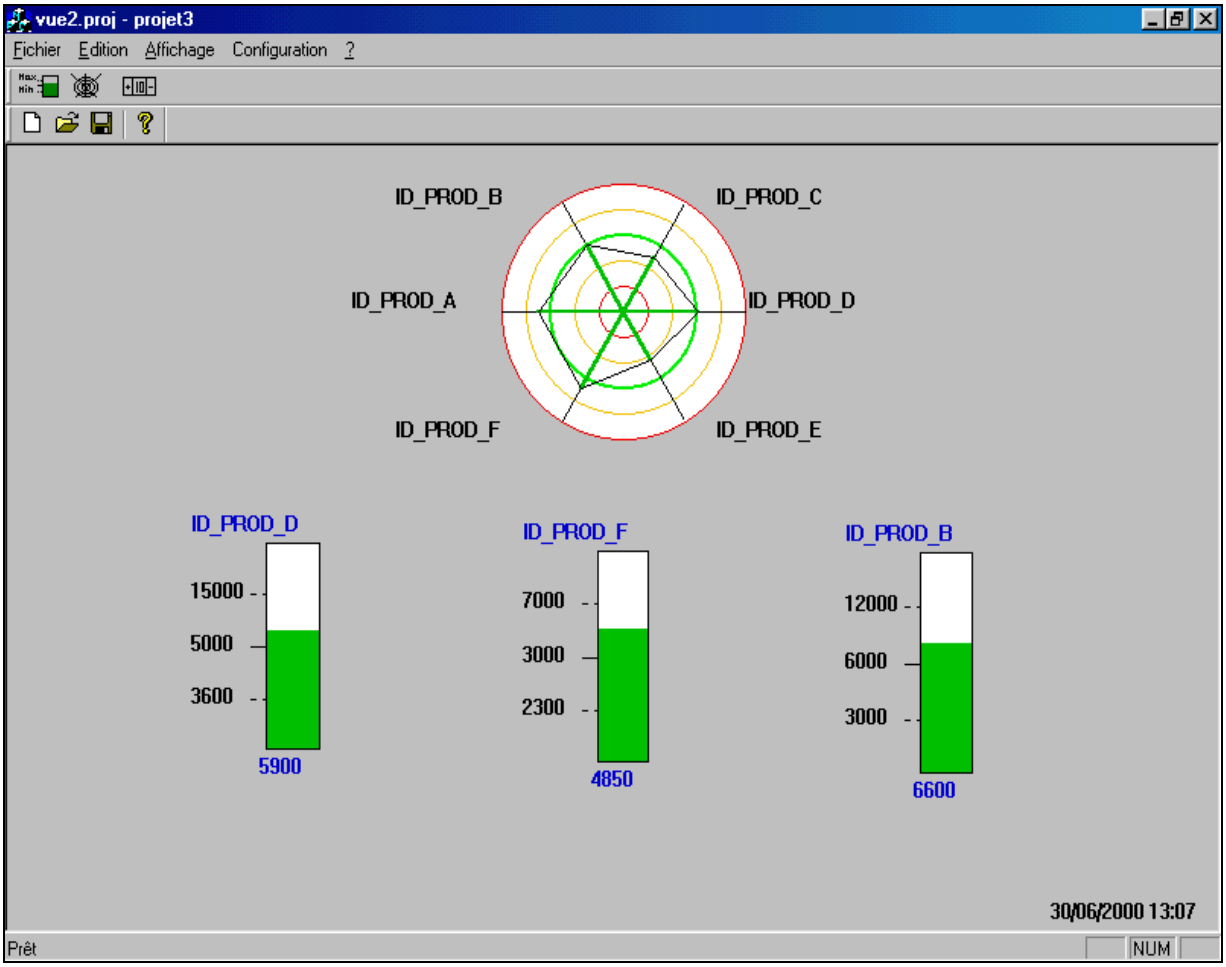


Figure 5.10 : Une vue graphique générée à l'aide de l'outil Ergo-Conceptor+ reflétant un état de fonctionnement normal du système

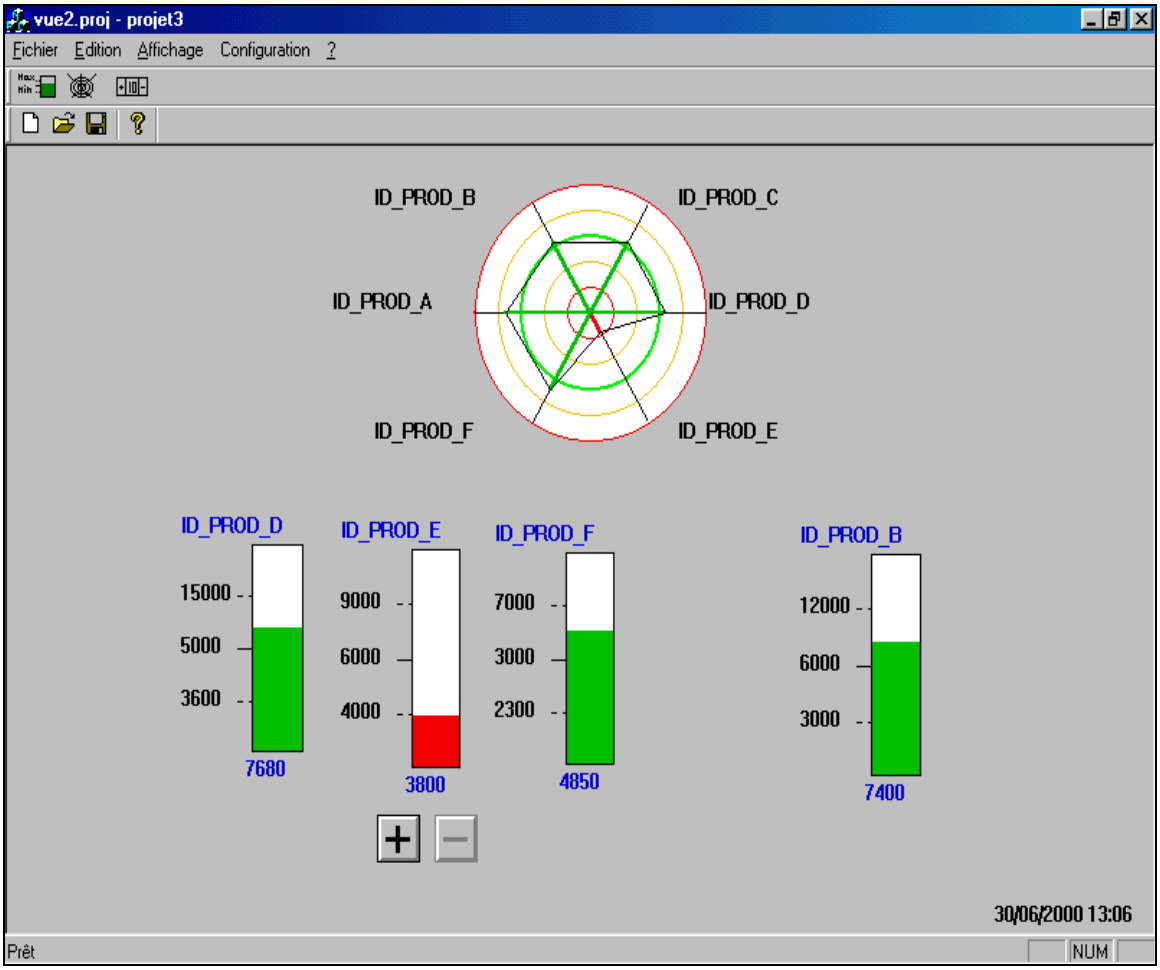


Figure 5.11 : Une vue graphique générée à l'aide du système Ergo-Conceptor+ reflétant un état de dysfonctionnement du système (relatif à la variable ID-PROD-E)

II.5- Extrait de la modélisation objet avec UML

L'implémentation de l'outil Ergo-conceptor+ s'est faite en technologie objets. Sans souci d'exhaustivité, nous proposons ci-dessous les principales composantes analysées au moyen de la méthode UML (Unified Modeling Language).

UML propose de modéliser toute application informatique selon différents modèles de vues et fournit neuf types de diagrammes [Jacobson et al, 00][Muller et al., 00]. Nous nous limitons ici au modèle statique avec le diagramme de classes et au modèle dynamique avec le diagramme de séquence.

II.5.1- Diagramme de classes

Dans la conception de l'outil Ergo-Conceptor+, nous pouvons distinguer principalement 8 classes :

- La classe Application : pour définir les objets applications à contrôler (nom, date, …)
- La classe Variable : pour définir les objets variables d'une application (type variable, valeur courante, seuils max et min, …)
- La classe RdP : pour définir les objets réseaux de Petri (nom, nombre de places, nombre de transitions,…)

- La classe Place : pour définir les objets Places d'un RdP (nom, numéro, …)
- La classe Transition : pour définir les objets Transitions d'un RdP (nom, numéro, …)
- La classe Simulateur : pour définir les objets permettant la simulation de contrôle d'une application donnée
- La classe Vue : pour définir les objets vues graphiques (nom, type,…)
- La classe Objet-Graphique : c'est une classe générique relative aux objets graphiques à manipuler, dont hérite les cinq sous-classes suivantes :[5]

    - La classe Barre-Graphe
    - La classe Etoile
    - La classe Potentiomètre
    - La classe Courbe
    - La classe Zone-Texte

- Le diagramme de classes est représenté par la figure 5.12 suivante.

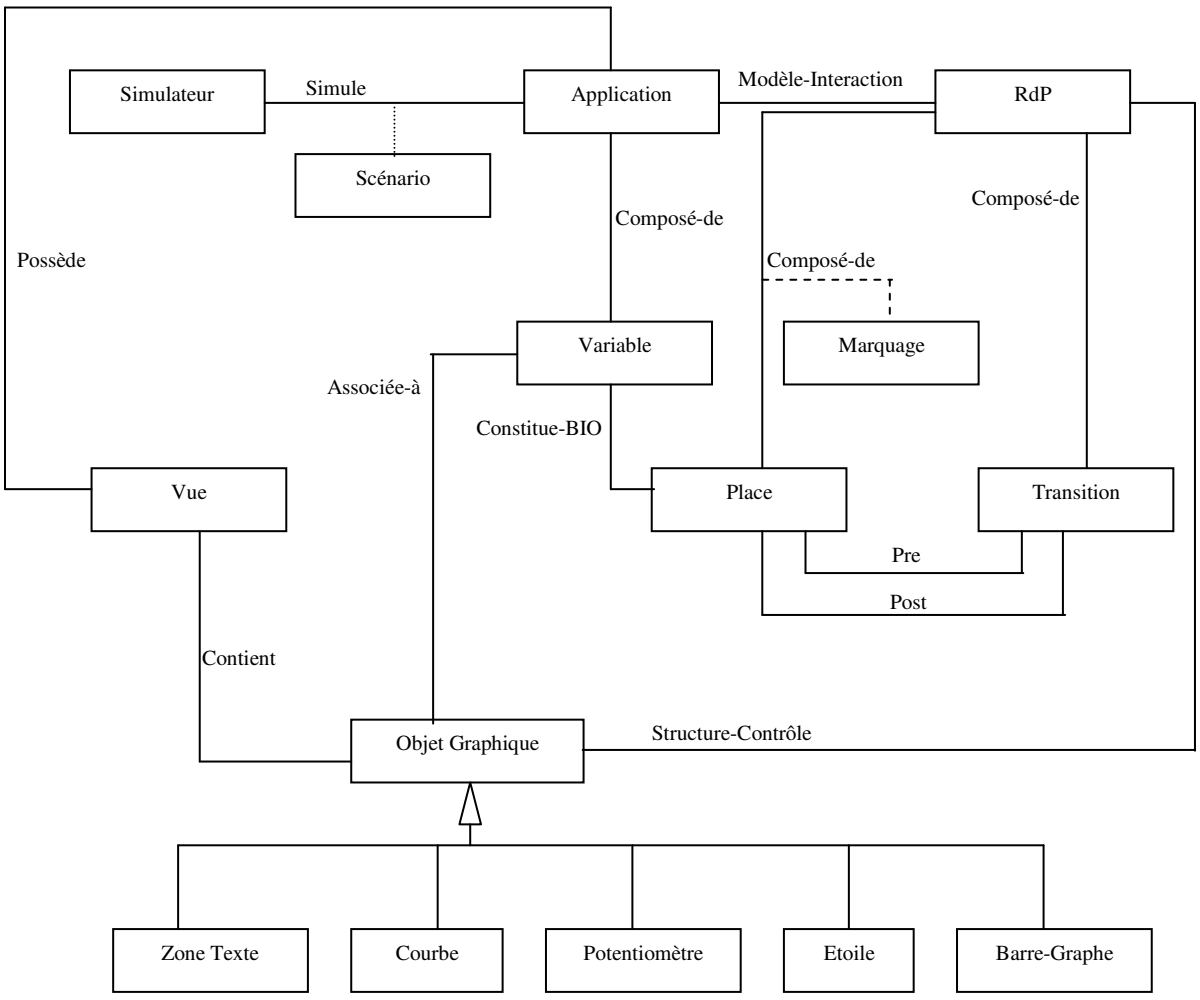


Figure 5.12 : Diagramme des classes

[5] Bien entendu, cette liste de types d'objets graphiques est loin d'être exhaustive. Elle est avant tout représentative, notre objectif étant de montrer la faisabilité de l'approche dans le cadre de cette recherche.

Nous pouvons y distinguer principalement les relations suivantes :

- Constitue-BIO : modélise l'association des objets variable aux objets places de RdP.
- Pre : modélise l'association "une place est Pre d'une transition".
- Post : modélise l'association "une place est Post d'une transition".
- Modèle-Interaction : associe un RdP à une application modélisant son modèle d'interaction.
- Structure-Contrôle : associe un RdP à un objet graphique modélisant sa structure de contrôle.
- Contient : modélise l'association des objets graphiques aux différentes vues.
- Associé-A : modélise l'association des variables aux objets graphiques.
- Scénario : définit une association entre le Simulateur et une application conformément un scénario donné.
- Marquage : définit une association de marquage des places d'un réseau.
- Composé-de : définit l'association des places et des transitions à un réseau de Petri.

II.5.2- Diagramme de séquence

Le diagramme des séquence simplifié suivant (figure 5.13) représente une séquence d'évènements entre les différents objets pour simuler une évolution d'états de l'application à contrôler.

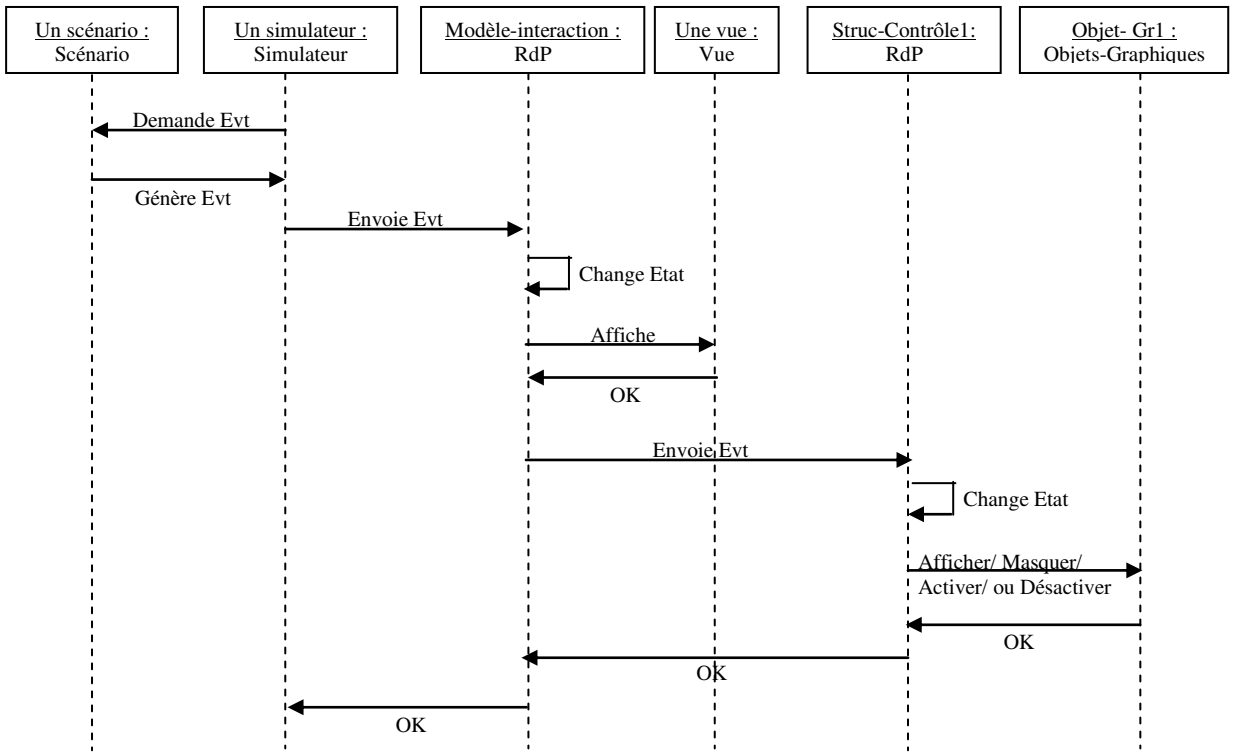


Figure 5.13 : Diagramme de séquence

L'objet simulateur demande un événement d'un objet scénario. Cet événement est envoyé par la suite à l'objet RdP représentant le modèle d'interaction de l'application en question. Ce dernier répond à l'événement reçu en changeant d'état. Il envoie ensuite un message à l'objet vue associée à cet état pour s'afficher. Il envoie de même des messages aux objets RdP relatifs aux structures de contrôle des objets graphiques pour s'afficher, se masquer, s'activer ou se désactiver selon qu'ils

font partie de l'ensemble des BIO de l'état en question ou non. Ces messages sont transférés vers les objets graphiques correspondants.

II.6 Environnement technique de réalisation

II.6.1 Environnement matériel

La machine utilisée pour la réalisation de l'outil Ergo-Conceptor+ est un micro-ordinateur compatible IBM-PS2 avec :

- carte mère monoprocesseur intel Pentium III 500 MHZ
- mémoire vive 64 MO
- capacité de stockage de 4 GO
- carte graphique de 2MO Ram vidéo

II.6.2 Environnement logiciel

Pour l'environnement logiciel de la réalisation, nous précisons les choix effectués pour le système d'exploitation, le langage de programmation et le gestionnaire de fichiers.

a- Le système d'exploitation

Le système d'exploitation choisi pour développer l'application est celui de Microsoft Windows 9x. En effet, ce système d'exploitation offre un environnement de programmation 32 bits et par la suite, une utilisation optimale de la puissance de calcul du processeur pentium. D'où la rapidité de l'exécution est assurée.

En outre, Microsoft Windows offre un environnement graphique puissant. Ce qui facilite l'implémentation de l'aspect ergonomique de présentation des interfaces.

b- Le langage de programmation

Développer une application exigeant une interface utilisateur sophistiquée est désormais une tâche ardue. Il faudrait utiliser pour cela un langage assez puissant et attractif. Le langage de programmation choisi est le Visual C++. Il a été choisi pour :

- *sa modularité* : il permet de bien implémenter les applications orientées objets en termes de classes et relations inter classes ;

- *sa réutilisabilité* : il offre la possibilité de compiler certaines classes et en former une bibliothèque qui pourra être utilisée dans un autre projet ;

- *sa portabilité* : il offre la possibilité d'exporter le code sur presque toutes les autres plates-formes du marché.

c- Le gestionnaire de fichiers

Le gestionnaire de fichiers utilisé est le C-Tree Plus de Faircom Corporation. On distingue principalement pour ce gestionnaire :

- *La convivialité* : Ecrit en C, C-Tree Plus est conçu pour s'intégrer facilement dans toute application C/C++ grâce à ses bibliothèques écrits 100% en C. Ces bibliothèques offrent des fonctions aussi bien de haut et de bas niveau de programmation. Il est fourni avec un pilote ODBC qui fait de lui un outil puissant de programmation sous windows.

- *La sécurité* : C-Tree Plus offre quatre niveaux de sécurité à savoir :

    - un système d'authentification des utilisateurs pour la confidentialité des données ;
    - un système de récupération automatique après incidents ;
    - un système de mirroring des fichiers de données ;
    - un système de progression transactionnelle pour assurer l'intégrité des données.

- *La performance* : Avec ce gestionnaire on a un accès rapide aux données utilisant la technique B++Tree et on dépasse les limites du nombre de fichiers ouverts précisées par le système d'exploitation. C-Tree Plus offre aussi la portabilité des applications développées sur n'importe quelle autre palte-forme supportant le langage C et assure la gestion des accès concurrents par la méthode du double verrouillage.

Afin de donner une petite idée sur ce gestionnaire, nous proposer de donner ici ces quelques chiffres :

- le nombre maximum de connexions sur une base : illimité
- le nombre maximum de tables dans une base : illimité
- le nombre maximum d'enregistrements dans une table : $2*10^9$
- le nombre maximum d'accès concurrents à une table : 405
- la taille maximale d'un enregistrement : 2 GO.

II.7- Conclusion sur l'outil Ergo-Conceptor+

L'outil Ergo-Conceptor+ est une première maquette montrant la faisabilité de l'approche globale de spécification et de conception des IHM, pour le contrôle des systèmes industriels complexes, proposée dans le cadre de nos travaux de recherche. Il constitue donc un exemple représentatif d'outil susceptible de venir en support de la démarche globale proposée dans le chapitre 3. Ces activités de recherche continuent et un ensemble de travaux futurs sont envisagés.

Ces travaux à entreprendre et ces perspectives de recherche sont maintenant présentés.

III. Perspectives de recherche

L'approche de spécification et de conception des IHM proposée dans le cadre de cette recherche visait à être la plus complète possible couvrant toutes les étapes nécessaires depuis l'analyse du SHM jusqu'à la génération des vues graphiques et la gestion automatique du dialogue homme-machine. Toutefois, certains points méritent plus de réflexion et d'approfondissement.

Nous proposons de présenter ci-dessous différents points retenus pour la continuité de ces travaux de recherche.

III.1- Intégration de l'approche multi-agents dans la démarche de spécification et de génération des IHM

Une étape clé de la démarche proposée consiste à spécifier l'interface proprement dite en termes d'objets graphiques et de dialogue selon des critères ergonomiques et des règles de style d'interface, particulièrement au moyen d'outils à base de connaissance (Approche de type *Tools For Working With Guidelines*). En effet, en dépit du large éventail de recommandations ergonomiques proposé

dans la littérature pour la spécification des interfaces, force est de constater que seules quelques tentatives existent et essayent de produire des outils formels automatisables.

Le choix effectué par notre équipe de recherche jusqu'ici repose sur l'utilisation d'un système à base de connaissances pour la spécification graphique et la génération semi-automatique de l'interface. Nous proposons d'étudier pour cette fin, une nouvelle approche selon laquelle l'interface graphique du système homme-machine serait conçue comme un système multi-agent (au sens de [Ferber, 95][Mandiau et Grislin, 02]) faisant coopérer un ensemble d'agents détenant chacun un savoir particulier (organisation du synoptique, codage coloré, rafraîchissement de l'affichage, etc.) [Riahi et al., 01].

Ce partage de connaissance et cet aspect coopératif des différents agents pourrait donner des performances meilleures et aboutir, à terme, à un système de génération "temps réel" des vues de l'interface.

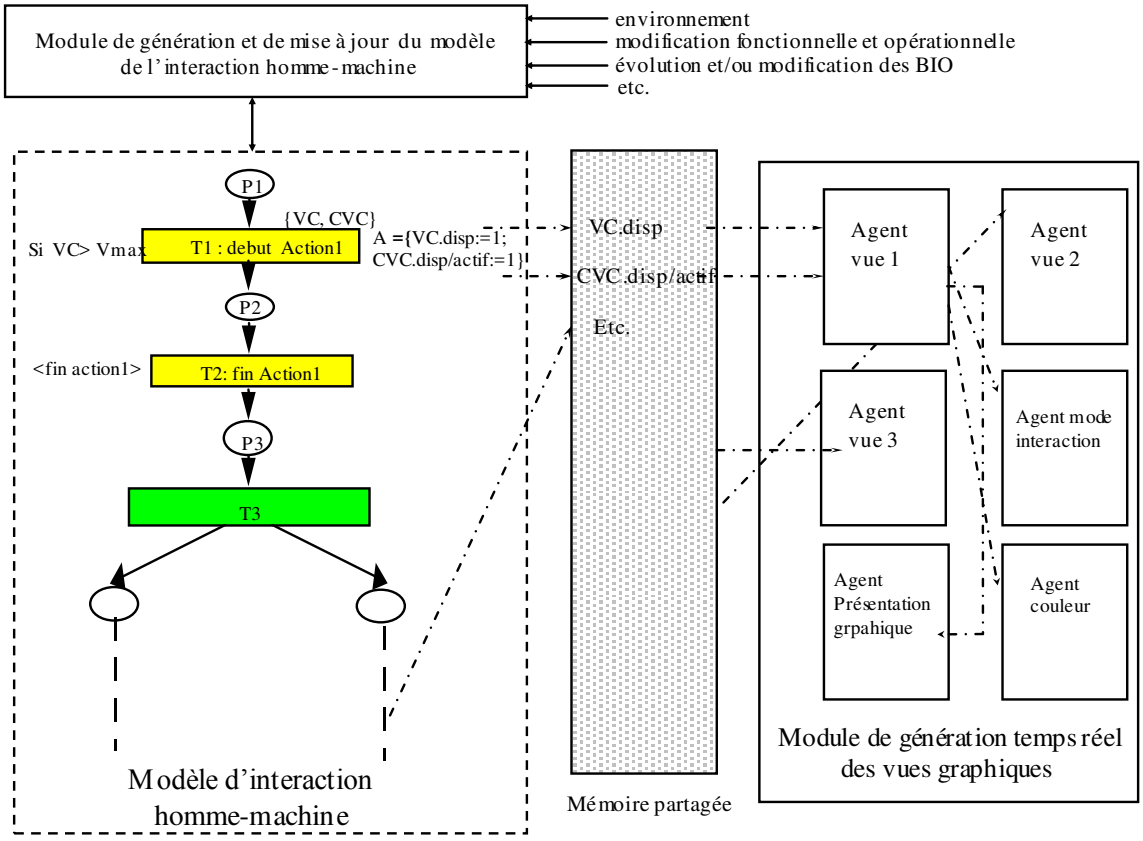


Figure 5.14 : système multi-agent de génération temps réel d'IHM

En effet, une approche de génération temps réel prend tout son intérêt dès lors que le modèle de l'interaction homme-machine devient évolutif et adaptatif par rapport à son environnement, aux variations inter et intra opérateurs humains, aux modifications fonctionnelles et opérationnelles du procédé, à l'évolution des BIO, etc.

Dans ce sens, la répercussion de toutes ces variations sur le modèle de l'interaction homme-machine pourrait être prise en compte d'une manière automatique grâce au module de génération et de mise à jour précisé dans la figure 5.14. La génération des spécifications ainsi que la création graphique des vues de l'interface se feraient en temps réel à chaque variation significative du modèle de l'interaction homme-machine ou de l'évolution du contexte de fonctionnement du procédé industriel.

III.2- Couplage d'un système de prédiction et d'aide à la conduite

L'objectif de ce point de réflexion consiste à réaliser des modules supplémentaires et les coupler à la maquette Ergo-Conceptor+, avec pour objectif de générer de manière semi-automatique et en temps réel une interface de prédiction et d'aide à la conduite. Ces modules pourraient consister à prédire les dysfonctionnements et les alarmes avant leur apparition, en fonction de l'évolution de l'état du procédé et conformément à certaines données précisant les influences mutuelles entre les différentes variables du système. Les résultats de cette analyse de prédiction seraient affichés en temps réel à l'opérateur humain l'incitant à agir dans les délais pour éviter l'apparition de ces alarmes.

Ce système aiderait l'opérateur à agir en lui fournissant des explications et justifications diverses sous différents formats et en lui proposant un plan d'action adapté.

Un rapprochement avec des équipes de recherche travaillant sur les systèmes d'aide (intelligents ou non) serait particulièrement fructueux.

III.3- Prise en considération de l'aspect profil utilisateur

Ce point considère l'aspect particulier de l'intégration du modèle de l'opérateur humain dans la démarche de spécification de l'interface. Il consisterait à ajouter au système un module permettant la prise en considération du profil de l'utilisateur dès l'étape de sa connexion sur l'application de supervision.

Une fois l'utilisateur identifié et ses caractéristiques reconnues (telles que son niveau d'expertise, ses privilèges, ses modalités d'actions sur le système, etc.), certaines d'entre elles pouvant d'ailleurs varier au cours du temps, seraient automatiquement installées dans le système afin d'adapter l'interface à son profil d'utilisation.

Cette réflexion pourrait se baser sur les nombreuses recherches, menées sur les profils utilisateur, au niveau international ; cf. par exemple à ce sujet [Chan, 00][Kay, 01][Petit-Rozé, 03].

III.4- Validation et évaluation sur cas réel

Dans le quatrième chapitre, nous avons présenté une étude de cas représentative d'applications industrielles sur lequel nous avons appliqué les différentes étapes de la démarche proposée. Le but de cette étude était de montrer la faisabilité de l'approche proposée.

L'objectif, ici est de proposer un projet de collaboration avec l'industrie afin de pouvoir valider et évaluer l'approche sur un cas réel d'application industrielle complexe et détailler les différentes étapes depuis l'analyse du SHM jusqu'à la génération des vues graphiques de l'interface de supervision et de contrôle. Puis procéder à une évaluation sur terrain avec les opérateurs de contrôle du système.

Conclusion

Ce chapitre a été scindé en trois parties. La première a été consacrée à la présentation générale de l'architecture conceptuelle et fonctionnelle de l'outil Ergo-Conceptor+ proposé pour supporter la démarche de conception et de spécification préconisée dans le cadre de cette recherche.

Une présentation détaillée de l'outil Ergo-Conceptor+ a fait l'objet de la partie suivante. Cet outil se compose principalement de quatre modules. Le premier assure la description du procédé en termes de variables du système à contrôler et dysfonctionnements possibles. Le second permet de définir le modèle de l'interaction homme-machine avec les RdPI et d'associer l'ensemble des BIO aux différentes places du réseau. Le troisième est responsable de la génération graphique des différentes vues de l'interface et le dernier assure la gestion automatique de la dynamique du dialogue homme-machine.

Un extrait de la modélisation de cet outil avec UML a été aussi présenté précisant deux types de diagrammes : le diagramme de classes et le diagramme de séquence.

Ergo-Conceptor+ est avant tout une maquette de recherche visant à montrer la faisabilité de l'approche globale faisant l'objet de cette recherche. Dans la dernière partie, nous avons exposé un ensemble de travaux futurs à entreprendre pour compléter ces activités de recherche. Ces perspectives et travaux de recherche concernent en particulier : l'intégration d'un système multi-agents dans la partie spécification et génération semi-automatique de l'interface, le couplage à Ergo-Conceptor+ d'un système de prédiction et d'aide à la conduite et la prise en considération de l'aspect profil utilisateur pour fournir des interfaces adaptées et personnalisées.

Conclusion générale

L'objet de ce travail de recherche est d'apporter une contribution à l'élaboration d'une démarche globale de conception des interfaces homme-machine dédiées particulièrement au contrôle de procédés industriels. Cette démarche se veut être la plus complète possible, s'appuyant sur l'utilisation d'outils formels pour couvrir de façon progressive et cohérente l'ensemble des étapes depuis l'analyse du SHM et la description du processus d'interaction homme-machine jusqu'à la génération et la gestion automatique des objets de l'affichage graphique.

Pour ce faire, une étude bibliographique relative à ce domaine a d'abord été effectuée. Dans le premier chapitre, nous avons commencé par rappeler les concepts et aspects généraux de l'interaction homme-machine. Puis, nous avons exposé les principales approches et méthodologies proposées dans la littérature pour la conception des interfaces homme-machine, en mettant en évidence leurs apports et leurs limites. Dans le deuxième chapitre, nous avons présenté les différentes méthodes et outils techniques qui peuvent être utilisés pour supporter de telles approches, notamment dans le domaine des applications industrielles.

Les contributions essentielles rapportées dans la littérature n'appréhendent, en général, qu'une partie de la problématique de la conception des interfaces homme-machine. Certaines de ces contributions se sont intéressées exclusivement à l'analyse de la tâche et à la manière de déduire les besoins informationnels des opérateurs sans toucher à la problématique de la spécification de l'interface. D'autres se sont focalisées plus sur l'aspect ergonomique de la présentation et la manière d'intégrer les recommandations ergonomiques dans le processus de spécification des vues et objets graphiques de l'interface. Plusieurs travaux ont, par ailleurs, traité de l'aspect génération et/ou évaluation de l'interface. Néanmoins, un intérêt grandissant est porté à l'utilisation de techniques formelles qui préparent le terrain sur des vérifications théoriques des spécifications et une validation *a priori* des interfaces permettant de gagner un temps considérable dans le cycle de développement de celles-ci.

Notre objectif était d'éviter de telles restrictions et de proposer une démarche complète qui couvre les différents aspects de la conception des IHM. Le but étant d'assister l'opérateur dans ses réactions face à des situations de dysfonctionnement, cette démarche doit satisfaire un certain nombre de critères tels que principalement :

1- Procéder par une analyse des tâches opérateurs ainsi qu'une analyse des différents états de fonctionnement du système ;
2- Identifier les BIO (besoins informationnels de l'opérateur) à partir de ces analyses et déduire les objets de l'interface à partir de ces BIO ;
3- Utiliser des techniques formelles (notre choix s'étant porté sur les réseaux de Petri Interprétés) qui préparent le terrain à des vérifications théoriques des spécifications et une validation *a priori* des interfaces.
4- Utiliser des techniques basées sur les recommandations ergonomiques formalisées pour la spécification graphique de l'interface.

La présentation détaillée de cette approche a fait l'objet du chapitre 3. Une attention particulière a été accordée à l'outil formel préconisé pour la modélisation de l'interaction homme-machine, les réseaux de Petri interprétés (RdPI), et à son apport pour la vérification des bonnes propriétés et l'évaluation *a priori* de l'interface.

Dans le quatrième chapitre, une étude de cas représentative d'applications industrielles réelles a été présentée. Les différentes étapes de l'approche ont été appliquées sur ce cas d'application industrielle depuis l'analyse du SHM jusqu'à la génération des vues graphiques de l'interface.

Le chapitre 5 a été consacré à la présentation de l'outil informatique Ergo-Conceptor+ proposé pour supporter la démarche préconisée. Une présentation détaillée de l'architecture conceptuelle et fonctionnelle de cet outil a fait l'objet de la première partie de ce chapitre. Par la suite, certaines perspectives de recherche ont été présentées. Elles concernent entre autres l'étude des modules relatifs à la spécification graphique de l'interface et la génération semi-automatique des vues. Une réflexion est retenue sur la manière de décrire des vues synthétiques de supervision pour un ensemble de sous-systèmes donné et sur la manière d'intégrer des techniques de formalisation des recommandations ergonomiques. L'étude et la contribution de voies nouvelles et d'approches originales telles que les systèmes multi-agents (issus de l'intelligence artificielle distribuée) sont envisagées. Le couplage au système actuel Ergo-Conceptor+, d'un système complémentaire de prédiction d'alarmes et d'aide à la conduite est également étudié. Un autre axe de recherche complémentaire est aussi considéré. Il concerne la prise en compte de l'aspect profil utilisateur lors de la spécification des interfaces afin de pouvoir générer des interfaces plus adaptées.

Références Bibliographiques

[Abed, 90] M. Abed. Contribution à la modélisation de la tâche par outils de spécification exploitant les mouvements oculaires : application à la conception et l'évaluation des interfaces homme-machine. Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis, Septembre 1990.

[Abed et Ezzedine, 98] M. Abed, H. Ezzedine, Vers une démarche intégrée de conception-évaluation des systèmes Homme-Machine. Journal of Decision Systems, vol. 7, pp. 147-175, 1998.

[Abed, 01] M. Abed, Méthodes et modèles formels et semi-formels pour la conception et l'évaluation des systèmes homme-machine, habilitation à diriger des recherches. Université de Valenciennes et du Hainaut-Cambrésis, France, mai 2001.

[Abed et al., 01] M. Abed, Modélisation des tâches dans la conception et l'évaluation des systèmes interactifs : la méthode SADT/Petri. In Kolski C. (Ed.), Analyse et Conception de l'IHM. Interaction Homme Machine pour les SI, Volume 1, pp. 145-174. Paris: Éditions Hermes, 2001.

[Ait-Ameur et al., 98] Y. Ait-Ameur, P. Girard and F. Jambon, A uniform approach for specification and design of interactive systems: the method B, Proceedings DSV-IS'98, Coesner's House, Abingdon, UK, June 3-5 1998, pp. 333-352.

[Alty et al., 85] J.L. Alty, P. Elzer, O. Holst, G. Johannsen, S. Savory, Literature and user survey of issues to man-machine interfaces for supervision and control systems. Report of Esprit project P600, Copenhagen: C.R.I., 1985.

[Amalberti et al., 98] R. Amalberti, J.M. Hoc, Analyse des activités cognitives en situation dynamique: pour quels buts? Comment ?, Le travail humain vol 61, n°3, 1998, pp. 209-234.

[Apple, 94] Apple Computer Inc, Inside MacIntoch :Quickdraw Gx Programmer's Overview, Addison-Wesley Publisher, 1994.

[Bass et al., 91] L. Bass, R. Little, R. Pellegrino, S. Reed, S. Seacord, S. Sheppard, M. Szesur, The Arch model: Seeheim revisited. Proceedings of User Interface Developpers' Workshop, at CHI'91, pp. 2-26, avril 1991.

[Bastide et al., 98] R. Bastide, P. Palanque, D. Le, J. Munoz, Proceedings of design specification and verification of intercative systems, DSV-IS'98, Springer Verlag, pp. 171-191.

[Bastien, 96] J. M. C. Bastien, Les critères ergonomiques : un pas vers une aide méthodologique à l'évaluation des systèmes interactifs. Thèse de doctorat d'ergonomie cognitive, Paris, France, 1996.

[Bastien et Scapin, 01] J.M.C. Bastien, D.L. Scapin. Évaluation des systèmes d'information et Critères Ergonomiques. In C. Kolski (Ed.), Systèmes d'information et interactions homme-machine. Environnement évolués et évaluation de l'IHM. Interaction homme-machine pour les SI (Vol. 2, pp. 53-79). Paris: Hermès, 2001.

[Bergot et al., 95] M. Bergot, L. Grudzien, Sûreté et diagnostic des systèmes industriels : principaux concepts, méthodes, techniques et outils, Revue diagnostic et sûreté de fonctionnement, vol 5, n°3, 1995, pp. 317-344.

[Berthomieu et al., 01] B. Berthomieu, M. Boyer, M. Diaz, C. Girault, S. Haddad, P. Moreaux, J.F. Pradat-Peyre, P. Sénac, F. Vernadat , Les Réseaux de Petri, Modèles Fondamentaux , Ed. Hermes 2001, ISBN 2-7462-0250-6.

[Birell, 85] A.D. Birrell, "Secure Communications Using Remote Procedure Calls," ACM Transactions on Computer Systems, Vol.3, No.1, February 1985, pp.1-14.

[Bodart et al., 95] F. Bodart, A. Hennebert, J.M. LeHeureux, I. Provot, J. Vanderdonckt et G. Zuchinetti, Key activities for a development Methodology of Interactive Applications', Chapter 4 in 'Critical Issues in User Interface Systems Engineering', D. Benyon and Ph. Palanque (Eds), Springer-Verlag, 1995.

[Boehm, 88] B.W. Boehm, A spiral model of software development and enhancement, computer, may 1988.

[Brams, 83] G.W. Brams, Réseaux de Petri: Théorie et Pratique, Tome 2: modélisation et application, Editions Masson, Paris, 1983.

[Brun et al., 97] P. Brun, F. Jambon, Utilisation des spécifications formelles dans le processus de conception des interfaces homme-machine. IHM'97, 9ème journées sur l'ingénierie de l'interaction homme-machine, 10-12 septembre 1997, France, pp. 23-29.

[Calvez, 93] J.P Calvez, Spécification et conception des systèmes : une méthodologie, pp. 93-157, Edition Masson, 1993.

[Cambacau, 98] M. Cambacau, Contribution à la surveillance hiérarchisé des sytèmes complexes, habilitation à diriger des recherches. LAAS du CNRS, Toulouse, France, Juin 1998.

[Card et al., 83] S.K. Card, T.P. Moran and A. Newell, The psychology of human-computer interaction. Hillsdale, NJ: Erlbaum, 1983.

[Cassar et al., 94] J. P. Cassar, R. G. litwak, V. Cocquempot, M. Staroswiecki, Approche structurelle de la conception des systèmes de surveillance pour les procédés industriels complexes, Revue diagnostic et sûreté de fonctionnement, vol 4, n°2, 1994, pp. 179-202.

[Chan, 00] P.K. Chan. Constructing web user profiles: A non-invasive learning approach. In B. Masand and M. Spiliopoulou (eds), Web Usage Analysis and User Profiling. Springer, 2000.

[Coutaz, 90] J. Coutaz, Interfaces Homme-Ordinateur: Conception et réalisation, Editions Dunod Informatique, Bordas, 1990.

[D'Ausbourg et al., 96] B. D'Ausbourg, G. Durrieu and P. Rocher, Deriving a formal model of interactive system from its UIL description in order to verify and to test its behaviour, in Proceedings of DSV-IS'96, springer verlag, pp. 104-122.

[D'Ausbourg et al., 97] B. D'Ausbourg, G. Durrieu and P. Rocher, vers un environnement de validation automatique des IHM, IHM'97, 9ème journées sur l'ingénierie de l'interaction homme-machine, 10-12 septembre 1997, France.

[David et al., 92] R. David and H. Alla, Du GRAFCET aux Réseaux de Petri, Editions HERMES, Paris, 1992.

[De Keyser et al., 92] V. De Keyser, The nature of Human Expertise, rapport intermédiaire établi dans le cadre de la convention RFO/AI/18, Université de liège, faculté de psychologie et des sciences de l'éducation, 189 p., Belgium, Mars 1992.

[De Montmollin, 95] M. De Montmollin, (ed.), Vocabulaire de l'ergonomie, Octares, Toulouse, 1995.

[De Nicola, 91] R. De Nicola, Action and State-based Logics for Process Algebras. CONCUR '91, 2nd International Conference on Concurrency Theory, Amsterdam, The Netherlands, August 26-29, 1991, Proceedings. Lecture Notes in Computer Science 527 Springer 1991, ISBN 3-540-54430-5.

[De Rosis et al., 98] P. De Rosis, Formal Description and Evaluation of User Adapted Interfaces, Int. Journal of Human-Computer Studies, vol. 49, 1998, pp. 95-120.

[Dhillon et al., 81] B.S. Dhillon, C. Singh, Engineering reliability, Jhon Wiley and Sons New York, 1981.

[Diaper et Stanton, 01] D. Diaper et N. Stanton (Eds), the handbook of task analysis for human-computer interaction, Lawrence Erlbaum Associates, London 2001.

[Duce et Paterno., 93] D. A. Duce, F. Paternò, A Formal Specification of a Graphics System in the Framework of the Computer Graphics Reference Model. Comput. Graph. Forum 12(1): 3-20 (1993).

[Embrey, 86] D. Embrey, A systematic human error reduction and prediction approach, in proceeding of international topical meeting on advances in human factors in nuclear power systems, Knoxville Tenessee, USA, 1986.

[Ericson, 69] C. A. Ericson, System safety analytical technology, Preliminary Hazards Analysis, the Boeing Co., Seattle, report. D2, 113072, 1, 1969.

[Ezzedine, 02] H. Ezzedine, Méthoodes et modèles de spécification et d'évaluation des interfaces homme-machine dans les systèmes industriels complexes, Mémoire d'Habilitation à diriger des recherches, LAMIH, Université de Valenciennes et du Hainaut Cambrésis, décembre 2002.

[Fadier, 90] E. Fadier, Fiabilité Humaine: Méthodes d'analyse et domaine d'application. In: Leplat, J., de Terssac, G. (Eds.): Les facteurs humains de la fiabilité dans les systèmes complexes. Editions Octarès, Marseille, 1990, pp. 47-80.

[Fekete et al., 01] J. D. Fekete, P. Girard, Environnements de développement des systèmes interactifs, chapitre 1, dans "Environnements évolués et évaluation de l'IHM : interaction homme-machine pour les SI 2", C. Kolski (Ed.), editions Hermes, 2001, pp. 23-52.

[Ferber, 95] J. Ferber. Les systèmes multi-agents. InterEditions, Paris, 1995.

[Geary et al., 97] Geary D.M., McClellan A.L., Graphic java : mastering the AWT, SunSoft Press, 1997.

[Gilmore et al., 89] W.E. Gilmore, D.I. Gertman and H.S. Blackman, User Computer Interface in process Control: A Human factor Engineering Handbook, Academic Press, 1989.

[Girault et Valk, 02] C. Girault, R. Valk, Petri Nets for Systems Engineering: A Guide to Modeling, Verification, and Applications, Springer-Verlag, 2002, ISBN: 3-540-41217-4.

[Gomes et al., 01] L. Gomes, J. P. BARROS, A. Costa, Man-Machine interface for real-time telecontrol based on Petri nets specification, In T. Bahill, F. Y. Wang (Ed.), IEEE SMC 2001 Conference Proceedings (e-systems, e-man and e-Cybernetics), Arizona, USA: IEEE Press, pp. 1565- 1570, 2001.

[Grammenos et al., 99] D. Grammenos, D. Akoumianakis, C. Stephanidi, Integrated support for working with guidelines: the guideline management system, Interacting with Computers, 12, 2, 1999, pp. 281–311.

[Grislin, 95] M. Grislin, Définition d'un cadre pour l'évaluation *a priori* des Interfaces Homme-Machine dans les systèmes industriels de supervision, Thèse de doctorat de l'Université de Valenciennes et du Hainaut-Cambrésis, Novembre 1995.

[Grislin et al, 97] M. Grislin, C. Kolski, Proposition d'une démarche d'évaluation à priori des IHM de supervision: application au étapes de spécification fonctionnelle et de conception préliminaires, RAIRO-APII-JESA, vol 31, n°7, 1997, pp. 1111-1154.

[Halbwachs et al., 91] N. Halbwachs, P.Caspi, P. Raymond, D. Pilaud, Programmation et vérification des systèmes réactifs : le langage Lustre, Techniques et Sciences Informatiques, vol 10, n°2, 1991, pp. 139-158.

[Hammouche, 93] H. Hammouche, De la modélisation des tâches à la spécification d'interfaces utilisateur, Rapport de recherche, INRIA, n 1959, Juillet 1993.

[Harel et al., 87] D. Harel, Statecharts: a visual formalism for complex systems. In Science of Computer Programming, volume 8, 1987.

[Harrison et al., 90] M. Harrison and M.Thimbelbey, In formal methods in human computer interaction, Harrison & Thimbelby (Eds), Cambridge university Press, 1990.

[Hartson et al., 89] H.R. Hartson, D. Hix, Humain computer interface development: concepts and systems for its management. ACM : computer surveys, vol 21, number1, pp. 5-92, 1989.

[Hassl et al., 65] D.F. Hassl, Advanced concepts in fault tree analysis. System Safety Symposium, Seattle, 8-9 Juin 1965.

[Hedin et al., 81] F. Hedin, A. Le Coguiec, C. Le Floch, M. Llory, A. Villemeur : The Failure Combination Method (FCM). 1981, annual reliability and maintainability symposium, congrès IEEE, Philadephie, Etats Unis, janvier 1981.

[Hix et Hartson, 93] D. Hix, H. R. Hartson, Developing user inetrface: Ensuring usability through. Product process, John Wiley Sons, New York, 1993.

[Hoc and Amalberti, 95] J.M. Hoc, R. Amalberti. Diagnosis: some theoretical questions raised by applied research. Current Psychology of Cognition, 14 (1), pp. 73-101, 1995.

[IGL, 89] I.G.L. Technology, SADT, un langage pour communiquer. Eyrolles, Paris, 1989.

[Ilog, 94] Ilog S.A., ILOG views reference manual, version 2.1, 1994.

[IWC 99] Interacting with Computers, Special issue, Volume 12, 2, 1999.

[IWC 00] Interacting with Computers, Special issue, volume 12, 3, 2000, 2000.

[Jacko and Sears, 03] J. A. Jacko et A. Sears (Eds.) : The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications (human factors and ergonomics) ; Lawrence Erlbaum Associates, 2003.

[Jacob, 85] R.J.K. Jacob, A state transition diagram language for visual programming, IEEE Computer, vol. 18 (8), August 1985, pp. 51-59.

[Jacobson et al ., 00] I. Jacobson, G. Booch, I. Rumbaugh, Le processus unifié de développement logiciel. Editions Eyrolles, Paris, 2000.

[Jambon et al., 01] F. Jambon, Ph. Brun, Y. Aït-Ameur. Spécification des systèmes interactifs. In Kolski C. (Ed.), Analyse et Conception de l'IHM. Interaction Homme Machine pour les SI, Volume 1, pp. 175-206. Paris: Éditions Hermes, 2001.

[James, 91] M.G. James, Produser: Process for developping user interfaces. In taking software design seriously, J. Karat (Ed.), Academic Press, 1991.

[Jeffries, 97] R. Jeffries, The role of task analysis in the design of software, In Handbook of Human-Computer Interaction, Helander M.G., Landauer T.K., Prabhu P. (Eds), North Holland, Amsterdam, pp. 347-359, 1997.

[Johannsen, 95] G. Johannsen, Conceptual design of multi-human machine interfaces. Proceedings 6th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design and Evaluation of Man-Machine Systems, MIT, Cambridge, USA, June 27-29, 1995.

[Johnson et al., 92] C.W. Johnson et M.D Harrison, Using Temporal logic to support the specification and prototyping of interactive control systems, Int. J. Man-Machine Studies, vol. 37, pp. 357-385, 1992.

[Kay, 01] J. Kay User Modeling for Adaptation. In Stephanidis C. (ed.), User Interface for All. LEA, publishers London, 2001, pp. 271-294.

[Khelil, 01] N. Khelil, Modélisation sûre des interactions dans les interfaces homme-machine, mémoire de DEA, faculté des sciences de Tunis, octobre, 2001.

[Kneuper, 97] P. Kneuper, Limits of formal methods, formal aspects of computing, vol 9, 1997, pp. 379-394.

[Kolski et al., 91] C. Kolski and P. Millot, A rule-based approach for the ergonomic evaluation of man-machine graphic interface. International Journal of Man-Machine Studies, 35, pp. 657-674, 1991.

[Kolski et al., 96] C. Kolski and F. Moussa, Two examples of tools for working with guidelines for process control field: Synop and ERGO-CONCEPTOR, Third Annual Meeting of the International Special Interest Group on "Tools for working with guidelines", J. Vanderdonckt (Ed.), Namur, Belgium, 4 June 1996.

[Kolski, 97] C. Kolski, Interfaces homme-machine, application aux systèmes industriels complexes. Editions Hermes, Paris, 1997.

[Kolski, 01] C. Kolski (Ed.), Analyse et conception de l'IHM : Interaction Homme-Machine pour les SI 1, Editions Hermes, Paris, 2001.

[Kolski et al., 01] C. Kolski, H. Ezzedine, M. Abed, Développement du logiciel : des cycles classiques aux cycles enrichis sous l'angle de l'IHM, In C. Kolski (Ed.), Analyse et conception de l'IHM : Interaction Homme-Machine pour les SI 1, Editions Hermes, Paris, 2001.

[Lambert, 73] H.E. Lambert, System Safety Analysis and Fault Tree Analysis, UCID-16238, 31,9 Mai 1973.

[Lambert et al., 97] M. Lambert, B. Riera, B. Vilain, Application of some functional analysis techniques on a nuclear fuel reprocessing system, 5th international workshop on functional modeling complex technical systems, 1997.

[Larvet, 94] P. Larvet, Analyse des systèmes : de l'approche fonctionnelle à l'approche objet, Inter Editions, 1994.

[Lepreux et al., 03] S. Lepreux, M. Abed, C. Kolski. A human-centred Methodology Applied to Decision Support System Design and Evaluation in Railway Network Context. Cognition Technology and Work, 5, pp. 248-271, 2003.

[Long et al., 90] J.B. Long, I. Denley, Evaluation for practice, tutorial, ergonomics society annual conference, 1990.

[Lichter et al., 94] H. Lichter, M. Schneider-Hufschmidt, H. Zullighoven, prototyping in industrial software projects, bridging the gap between theory and practice, IEEE transactions on software engineering, 20(11), pp. 825-832, 1994.

[Mahfoudhi, 97] A. Mahfoudhi, TOOD: Une méthodologie de description orientée objet des tâches utilisateur pour la spécification et la conception des IHM: Application au contrôle du trafic aérien, Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis, France, 1997.

[Mandiau et Grislin, 02] R Mandiau, E. Grislin-Le Strugeon, systèmes multi-agents, In TI(Ed.), s7216, pp.1-17, Paris, les Techniques de l'Ingénieur, 2002.

[Marcus, 97] A. Marcus, Graphical user interfaces. In Handbook of Human-Computer Interaction, Helander M.G., Landauer T.K., Prabhu P. (Eds), North Holland, Amsterdam, pp. 423-440, 1997.

[Marti, 98] P. Marti, Structured task analysis in complex domains, Ergonomics, vol 41, n°11, 1998, pp. 1664-1677.

[McDermid et al., 84] J. McDermid, K. Ripkin, life cycle support in the ADA environment, University Press Cambridge, 1984.

[Meinadier, 91] J.P. Meinadier, L'interface Utilisateur pour une informatique plus conviviale, DUNOD Informatique, Paris, 1991.

[Microsoft, 00] Microsoft corporation, Microsoft® mastering MFC development using microsoft visual c++® 6.0, Microsoft Press, 2000.

[Millot, 90] P. Millot, Coopération homme-machine dans les procédés industriels. Dans « les facteurs humains de la fiabilité dans les systèmes complexes »., J. Leplat et G. De Terssac (éditeurs), Eition Octares/ entreprises, Marseille, 1990.

[Millot, 99] P. Millot, La supervision et la coopération Homme-Machine dans les grands systèmes industriels ou de transports. In J-g Ganascia (Ed.), Sécurité et Cognition, Hermès, Paris, pp. 124-145.

[Moalla, 85] M. Moalla, Réseaux de Petri interprétés et Grafcet', revue TSI de l'AFCET vol. 4 (1), 1985.

[Monin, 96] J.F. Monin, Comprendre les méthodes formelles: panorama et outils logiques, Edition Masson, Paris, 1996.

[Mori et al., 02] G. Mori, F. Paterno, C. Santoro, CTTE : Support for developing and analyzing task models for interactive system design, IEEE transactions on software engineering, vol. 28, n° 9, septembre 2002.

[Moussa et al., 90] F. Moussa, C. Kolski and P. Millot, Artificial intelligence approach for the creation and the ergonomic design of man-machine interfaces in control room. Ninth European Annual Conference on "Human decision making and manual control", Varese, Italy, September 10-12, 1990.

[Moussa, 92] F. Moussa, Contribution à la conception ergonomique des interfaces de supervision dans les procédés industriels, application au système ERGO-CONCEPTOR. Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis, 1992.

[Moussa et al., 99] F. Moussa, M. Riahi, M. Moalla and C. Kolski, C., A petri net method for specification and automatic generation of user interface. In « Human-Computer Interaction: Ergonomics and user interfaces, volume 1, Proceedings HCI'99, 8th International Conference on Human-Computer Interaction, Munich, Germany, August », H.J. Bullinger, J. Ziegler (Eds.), Lawrence Erlbaum Associates, 1999, pp. 988-992.

[Moussa et al., 00a] Moussa F., Riahi M., Use of formalized guidelines for semi-automated generation of GUI : the Ergo-Conceptor+ tool. In J. Vanderdonckt, C. Farenc (Ed.), Tools

for Working With Guidelines TFWWG'2000, Biarritz, October, (pp. 237-246), London: Springer, 2000.

[Moussa et al., 00b] F. Moussa, C. Kolski, M. Riahi, A Model Based Approach to Semi-Automated User Interface Generation for Process Control Interactive Applications. Interacting with Computers 12, 3, 2000, pp. 245–279.

[Moussa et al., 02] Moussa F., Riahi M., Kolski C., Moalla M., Interpreted Petri Nets used for Human-Machine Dialogue Specification. International journal: Integrated Computer-Aided Engineering (ICAE). Volume 9, N° 1, 2002, (pp. 87-98).

[Next, 92] Next Computer Inc., NextSTEP development tools and techniques, Addison-Wesley Publisher, 1992.

[Nielsen, 71] D. S. Nielsen: The Cause Consequence Diagram Method as a basic for quantitative accident analysis. Danish Atomic Energy Commission, Report Riso, M. 1374, Denmark, May 1971.

[Nielsen, 93] J. Nielsen, Usability engineering, Academic Press, 1993.

[Norman, 86] D.A. Norman, Cognitive engineering, In D.A. Norman & S.W. Draper (Eds), User centred system design: new perspectives on human computer interaction. Hillsdale N.J., Elbraum, 1986.

[O'Hara et al., 96] J. O'Hara, W. Brown, W. Stubler, J. Wachtel and J. Persensky, Human-Machine Interface Design Review Guideline (NUREG-0700, rev. 1). U.S. Nuclear Regulatory Commission, Washington, 1996.

[O'Hara et al., 99] J. O'Hara and W. Brown, Software Tool for the Use of Human Factors Engineering Guidelines to Conduct Control Room Evaluations. In: Büllinger, H.J., Ziegler, J. (Eds.): Human-Computer Interaction: communication, cooperation and application design. Lawrence Erlbaum Associates, London, pp. 973–977, 1999.

[Palanque, 97] P. Palanque, Spécifications formelles et systèmes interactifs: vers des systèmes fiables et utilisables. Habilitation à diriger des recherches, Université de Toulouse I, 1997.

[Palanque et al., 97] P. Palanque and F. Paterno (Eds.), Formal Methods in Human-Computer Interaction, Springer Verlag, 1997.

[Pardi, 99] W.J. Pardi, XML en action, Microsoft Press, 1999.

[Parnas, 69] D.L. Parnas, On the use of transition diagrams in the design of user interface for an interactive computer system, 24 th ACM Conference, 1969, pp. 379-385.

[Paterno et Faconti, 92] F. Paterno and G. Faconti, On the use of Lotos to describe graphical interaction, In proceedings of people and computer VII, HCI'92 conference, cambridge university press, 1992, pp. 155-174.

[Paterno et Mancini, 99] F. Paterno, C. Mancini, Designing usuable hypermedia, empirical software engineering, 4(1), pp. 11-42, 1999.

[Paterno, 01] F. Paterno, Towards a UML for interactive systems, Poc. Eng. Human-Computer Interaction conference, HCI'01, pp. 175-185, 2001.

[Petit-Rozé, 03] C. Petit-Rozé Organisation multi-agents au service de la personnalisation de l'information, application à un système d'information multimodale pour le transport terrestre de personnes. Thèse de Doctorat en Informatique, Université de Valenciennes et du Hainaut-Cambrésis, Décembre 2003.

[Petri, 62] C.A. Petri, Kommunication mit automaten, PhD Dissertion. University of Bonn, 1962.

[Pfaff, 85] G.E. Pfaff, User Interface Management Systems, Edition Eurographics Seminars, Springer-Verlag, 1985.

[PRA, 82] PRA Procedures Guide: A guide to performance of probabilistic risk assessments for nuclear power plants. US Nuclear Regulatory Commission, NUREG/CR 2300, 1982.

[Rasmussen, 86] J. Rasmussen, Intelligent Decision Support in Process Environments. A framework for cognitive Task Analysis in System Design In: Hollnagel, E., Mancini, G., Woods, D.D. (Eds.). NATO ASI series. Vol. F21. Springer-Verlag, Berlin (1986).

[Reason, 90] J. Reason, Human Error. Cambridge: Cambridge University Press, 1990.

[Recht, 66] J.L. Recht, Failure mode and effect. National Safety Council, 1966.

[Reed et al., 99] P. Reed, K. Holdaway, S. Isensee, E. Buie, J. Fox, J. Williams and A. Lund, User interface guidelines and standards: progress, issues, and prospects, Interacting With Computers, 12, 2, 1999, pp. 119–142.

[Riahi, 97] M. Riahi, Spécification formelle des interfaces homme-machine : application au cas du contrôle des procédés industriels, mémoire de DEA, faculté des sciences de Tunis, novembre, 1997.

[Riahi et al., 98a] M. Riahi, F. Moussa, M. Moalla et C. Kolski, Vers une spécification formelle des interfaces homme-machine basée sur l'utilisation des réseaux de Petri. In M.F. Barthet (Ed.), Actes du 6ème Colloque ERGO IA'98 Ergonomie et Informatique Avancée. (pp. 196-205). Bayonne: ESTIA/ILS, 1998.

[Riahi et al., 98b] M. Riahi, F. Moussa, M. Moalla et C. Kolski, Approche de spécification formelle des interfaces homme-machine basée sur une analyse du système homme-machine, application au cas du contrôle des procédés industriels. Sixième colloque Maghrébin sur les Modèles Numériques de l'Ingénieur, Tunis, Tunisie, novembre 1998.

[Riahi et al., 00] M. Riahi, F. Moussa, C. Kolski and M. Moalla, Use of interpreted petri nets for human-machine dialogue specification in process control. Proceedings ACIDCA'2000 International Conference on Artificial and Computational Intelligence for Decision, Control and Automation in Engineering and Industrial Applications, 22-24 March, Monastir, Tunisia.

[Riahi et al., 01] Riahi M., Moussa F., Contribution of the Petri Nets and the multi Agent system in HCI Specification. In M.J. Smith, G. Salvendy, D. Harris, R. Koubek (Ed.), Usability evaluation and Interface design: Cognitive Engineering, Intelligent Agents and

Virtual Reality, volume 1. (pp. 76-80). London: Lawrence Erlbaum Associate Publishers, 2001. (Actes de HCII2001, 9th International Conference on Human-computer Interaction, August New orleans, LA, USA.).

[Robert et al., 97] J. M. Robert, F Aubin, S. Jagannath, des tâches génériques aux interfaces génériques, IHM'97, 9ème journées sur l'ingénierie de l'interaction homme-machine, 10-12 septembre 1997, France, pp. 39-45.

[Rodriguez, 98] F.G. Rodriguez, Spécification et implémentation d'ALACIE: atelier logiciel d'aide à la conception d'interfaces ergonomiques. Thèse de doctorat, Université Paris sud, Paris XI, France, octobre, 1998.

[Rouse, 83] W.B. Rouse, Models of human problem solving : detection, diagnosis and compensation for system failures, Automatica, 19, 6, 1983, pp. 613-625.

[Rumbaugh et al., 91] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, W. Lorensen, Object oriented modelling and design, Prentice Hall, 1991.

[Sahraoui et al., 91] A. Sahraoui, L. Gilhodes, State charts, temporal logic and Petri nets to specify discrete event controllers: a comparative study on descriptive power, European control conference ECC 91, Grenoble, France, July 2-5 1991.

[Sahraoui, 96] A. Sahraoui, vers une intégration de méthodes fonctionnelles et de défaillances, 10ème colloque national de fiabilité et de maintenabilité, Saint-Malo, 1-3 octobre, 1996, pp. 737-750.

[Scapin, 89] D.L, Scapin et C. Pierret-Golbreich, MAD : une Méthode Analytique de Description des tâches, Actes du Colloque sur l'Ingénierie des Interfaces Homme-Machine, Sophia-Antipolis, 24-26 Mai, 1989.

[Scapin and Bastien, 97] D.L. Scapin, J.M.C.Bastien, Ergonomic criteria for evaluating the ergonomic quality of interactive systems, Behaviour and Information Technology, 16, p. 220-231, 1997.

[Scapin et al., 01] D.L. Scapin, J.M.C.Bastien, Analyse des tâches et aide argonomique à la conception : l'approche MAD*, In Kolski C. (Ed.), Analyse et Conception de l'IHM. Interaction Homme Machine pour les SI, Volume 1, pp. 85-116. Paris: Éditions Hermes, 2001.

[Scheifler et al., 86] Scheifler R.W., Gettys J., The X window system, ACM transactions on Graphics, vol 5, n°2, pp. 79-109, 1986.

[Schmucker, 87] Schmucker K.J., MacApp : an application framework, Readings in Human Computer Interaction : A multidisciplinary approach, Moragn-Kaufmann Publishers Inc., pp. 591-594, 1987.

[Shepherd, 98] A. Shepherd, HTA as a framework for task analysis, Ergonomics, vol 41, n°11, 1998, pp. 1537-1552.

[Sheridan, 85] T.B. Sheridan, Forty Five years of man machine system: history and trends. 2nd IFAC congress on "analysis, design and evaluation of man-machine systems", varese, italy, september, 1985.

[Sheridan et al., 97] T.B. Sheridan, Task analysis, task allocation and supervisory control. In Handbook of Human-Computer Interaction, Helander M.G., Landauer T.K., Prabhu P. (Eds), North Holland, Amsterdam, pp. 87-105, 1997.

[Shneiderman, 98] B. S. Shneiderman, Designingthe user interfaces: strategies for effective human-computer interaction, third edition, Reading MA, Addison Wesley, 1998.

[Sinclair et al., 65] I. A. G. Sinclair, R.G. Sell, R. G. Beishon, L. Bainbridge: Ergonomic study of L.D. Waste-heat boiler control room. J. Iron and steel Inst, 204, pp. 434-442, 1965.

[Smith, 86] S.L. Smith and J.N. Mosier, Guidelines for designing user interface software. Report EDS-TR-86-278, The MITRE Corporation, Bedford, MA, 1986.

[Stammers, 90] R.B. Stammers, M.S. Carey and J.A. Astley, Task analysis. In Evaluation of human work. A Practical Ergonomics Methodology, Wilson J. R. and Corlett E. N. (Eds.), Taylor & Francis, 1990.

[Suhner et al., 92] M. C. Suhner, M. Gabriel, L. Claude, Utilisation de l'AMDEC pour générer la base de connaissances d'un système expert d'aide au diagnostic : le cas des systèmes complexes automobiles, Revue diagnostic et sûreté de fonctionnement, vol 2, n°2, 1992, pp. 133-153.

[Swain, 63] A.D. Swain, A method for performing a human factors reliability analysis. Monograph SCR, 685, Sandia corporation, 1963.

[Swain, 64] A.D. Swain, THERP, Sandia lab, Albuquerque, W, Mex Rep SCR 64-1338, 1964.

[Swain et al., 81] A.D. Swain, H.G. Guttman, Handbook of human reliability analysis with emphasis on nuclear power plant applications. US Nuclear regulatory commission technical report NUREG/CR 1278, 1981.

[Szekely, 96] P. Szekely P. Retrospective and Challenges for Model-Based Interface Development. In J. Vanderdonckt (ed.), Proceedings of 2nd Int. Workshop on Computer-Aided Design of User Interfaces CADUI'96 (Namur, 5-7 June 1996), Presses Universitaires de Namur, Namur, pp. xxi-xliv, 1996.

[Tabary et al., 98] D. Tabary and M. Abed, TOOD: an object-oriented methodology for describing user task in interface design and specification - An application to air traffic control. La Lettre de l'Intelligence Artificielle, vol 134-135-136, pp. 107-114, 1998.

[Tabary, 01] D. Tabary, Contribution à TOOD, une méthode à base de modèles pour la spécification et la conception des systèmes interactifs. Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis, France, décembre, 2001.

[Taborin, 89] V. Taborin, Communication homme-machine au sein des systèmes de supervision de procédé continu, réalisation des interfaces entre opérateur et système d'aide à la décision, Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis, France, Mars 1989.

[Tardieu et al., 00] H. Tardieu, A. Rochfeld, R. Colletti, La méthode Merise - Principes et

outils, Editions d'Organisation, 2000.

[Traore, 98] I. Traore, Analyse dans l'ingénierie des systèmes, caractérisation et multiformalisme, thèse de doctorat, LAAS du CNRS, Toulouse, France, mai 1998.

[Terwilliger et al., 97] R. B. Terwilliger, P. G. Polson, Relationships between usersband interface task representations, CHI'97, 22-27 march 1997, pp. 99-106.

[Turner, 93] K.J. Turner, Using formal description techniques: An introduction to Estelle, Lotos and SDL, Edition John, 1993.

[Vallee, 92] F. Vallee, Statemate: une méthode et un outil pour la spécifications des systèmes réactifs, RGE, n° 4/92, pp. 46-51, Avril 1992.

[Vanderdonckt, 94] J. Vanderdonckt, Guide ergonomique des interfaces homme-machine. Presses Universitaires de Namur, Namur, Belgium, 1994.

[Vanderdonckt, 99] J. Vanderdonckt, Development Milestones toward a Tool For Working With Guidelines. Interacting with Computers 12, 2 (1999) 81–118.

[Vanderdonckt, 00] J. Vanderdonckt, A small Kowledge-Based System for selecting interaction styles, Proceedings of international workshop on Tools for Working with Guidelines, TFWWG-2000, Biarritz, 7-8 octobre 2000, Springer-verlag, Londres, pp. 247-262, 2000.

[Vicente et al., 01] K. J. Vicente, E. M. Roth, R. J. Mumaw, How do operators monitor a complex, dynamic work domain? The impact of control room technology, International Journal of Human-Computer Studies Vol. 54, pp. 831-856, Academic Press, 2001.

[Villemeur, 81] A. Villemeur, une méthode d'Analyse de la Fiabilité et de la sécurité des systèmes : la méthode des combinaisons des pannes (MCP), EDF-DER, HT/13//55/81, Octobre, 1981.

[Villemeur, 88] A. Villemeur, Sûreté de fonctionnement des systèmes industriels ; fiabilté, facteurs humains, informatisation, Edition Eyrolles, paris 1988.

[Ward et al., 85] P. Ward, S. J. Mellor: Structured Development for Real-Time. Prentice Hall, Englewood cliffs, New Jersey, 1985.

[Willem et al., 88] R. Williem and V. Biljon, Extending Petri Nets for specifying Man-Machine dialogues, International Journal of Man-Machine Studies, vol. 28, 1988, pp. 437-45.

[Winckler et Palanque, 03] M. Winckler et P. Palanque, StateWebCharts : a formal description technique dedicated to navigation modelling of web applications. Proceedings DSVIS'2003, Funchal, Portugal, June 2003.