Table des matières

Introduction générale **17**

1 Sécurité des systèmes de la robotique de service **21**
1.1 Introduction 21
1.2 Effets non désirés : les dommages 22
1.2.1 Dommage 23
1.2.1.1 Définition 23
1.2.1.2 Gravité d'un dommage 23
1.2.1.3 Occurrence d'un dommage 24
1.2.2 Notion de Risque 25
1.2.2.1 Définition 25
1.2.2.2 Mesure du risque 25
1.2.2.3 Risque acceptable 26
1.2.3 Sécurité 27
1.3 Causes : les dangers 28
1.3.1 Notion de danger 28
1.3.2 Phénomène dangereux et situation dangereuse 28
1.3.3 Événement dommageable 29
1.3.4 Incident et accident 29
1.3.5 Exemple d'utilisation des notions liées au risque 30
1.3.6 Analyse du danger 31
1.4 Moyens : la gestion du risque 31
1.4.1 Vue d'ensemble 31
1.4.2 Facteurs humains et gestion du risque pour des systèmes de la robotique médicale 32
1.4.3 Analyse du risque 34
1.4.3.1 Analyse du risque et processus de développement 34
1.4.3.2 Définition du système et de l'utilisation prévue 34
1.4.3.3 Identification des phénomènes dangereux 35
1.4.3.4 Estimation du risque 36
1.4.4 Évaluation du risque 37

1.4.5 Maîtrise du risque 40
1.5 Techniques utilisées pour l'analyse du risque 41
1.6 Techniques utilisées pour la maîtrise du risque 43
1.6.1 Conception matérielle 44
1.6.1.1 Architectures mécaniques et matériaux utilisés 44
1.6.1.2 Sécurité autour des actionneurs 45
1.6.1.3 Sécurité par la redondance des composants 47
1.6.1.4 Sécurité par la détection de contacts avec l'humain 48
1.6.2 Conception logicielle 49
1.6.2.1 Limitations logicielles des performances du robot 49
1.6.2.2 Sécurité lors de la mise sous tension 50
1.6.2.3 Surveillance du système 50
1.6.2.4 Conception des interfaces humain-machine 51
1.7 Assurance : la certification 52
1.7.1 Certification et risque 53
1.7.2 Classification des dispositifs médicaux 53
1.7.3 Processus de certification 54
1.8 Conclusion et problématique 55

2 Risques liés à l'utilisation des muscles artificiels de McKibben 57
2.1 Introduction 57
2.2 Description du muscle artificiel 59
2.2.1 Description générale 59
2.2.1.1 Fabrication du muscle 59
2.2.1.2 Utilisation du muscle 60
2.2.2 Évaluations des efforts en statique et dynamique 61
2.2.2.1 Force statique 61
2.2.2.2 Force dynamique 62
2.2.3 Compliance 63
2.2.4 Actionnement d'une articulation 63
2.2.5 Contrôle des muscles artificiels 65
2.2.6 Validation sur un bras sept axes anthropomorphe 66
2.3 Identification des dangers et estimation du risque 71
2.3.1 Phénomènes dangereux de type défaillances 72
2.3.1.1 Défaillances de la tresse et de son système de maintien . . 72
2.3.1.2 Défaillance du tube de caoutchouc 74
2.3.2 Situations dangereuses 76
2.4 Solutions possibles pour la maîtrise du risque 77
2.4.1 Moyens de prévention 77
2.4.2 Moyens de protection 77

2.5 Synthèse et conclusion 78

3 La modélisation orientée objet avec UML 81
3.1 Introduction 81
3.2 UML : notation pour le développement orienté objet 82
3.2.1 Le paradigme objet 82
3.2.1.1 La complexité des systèmes 82
3.2.1.2 La notion d'objet 83
3.2.1.3 Les classes d'objet 83
3.2.2 Le langage de modélisation UML 84
3.2.2.1 Pourquoi modéliser ? 84
3.2.2.2 Bref historique de UML 84
3.2.2.3 Les diagrammes UML 85
3.2.2.4 Le *métamodèle* 86
3.2.3 Processus de développement et UML 87
3.2.3.1 Piloté par les cas d'utilisation 88
3.2.3.2 Itératif et incrémental 88
3.2.3.3 Centré sur l'architecture 89
3.3 Travaux actuels sur le développement orienté objet et la sûreté de fonctionnement 90
3.3.1 Prévention des fautes 92
3.3.2 Prévision des fautes 93
3.3.3 Tolérance aux fautes 95
3.3.4 Élimination des fautes 95
3.3.5 Conclusions sur la sûreté de fonctionnement et le développement orienté objet 96
3.4 Conclusion 96

4 Intégration d'UML pour l'analyse du risque 99
4.1 Introduction 99
4.2 Description du système et de son utilisation 100
4.2.1 Modélisation du métier 101
4.2.2 Intégration du dispositif technologique dans la tâche de service 102
4.2.3 Définition des frontières du système 103
4.2.4 Description des tâches des acteurs 105
4.3 Analyse des dangers et estimation du risque 106
4.3.1 Problématique 106
4.3.2 Analyse préliminaire des dangers 107
4.3.3 Analyse des modes de défaillance 108
4.3.3.1 Analyse des modes de défaillance des messages 109

4.3.3.2 Modèles d'erreur des messages . . . 111
4.3.3.3 Proposition de tableau générique de l'AMDEC pour une analyse système . . . 114
4.3.4 Application aux différents domaines . . . 116
4.3.4.1 Analyse des modes de défaillance des messages provenant des acteurs de type dispositif extérieur . . . 117
4.3.4.2 Analyse des modes de défaillance des messages provenant des acteurs humains . . . 119
4.3.4.3 Analyse des modes de défaillance des composants électroniques . . . 121
4.3.4.4 Analyse des modes de défaillance des composants mécaniques . . . 125
4.3.4.5 Analyse des modes de défaillance du logiciel . . . 127
4.3.5 Analyse des arbres de fautes . . . 130
4.4 Conclusion . . . 132

5 Application de l'analyse du risque au développement en UML d'un robot télé-échographe 137
5.1 Introduction . . . 137
5.2 Présentation du projet TER . . . 138
5.3 Définition du système et de son utilisation . . . 139
5.3.1 Modélisation du métier : l'examen échographique . . . 139
5.3.2 Système robotisé pour la télé-échographie et allocation des tâches . . . 143
5.3.3 Définition des frontières de l'analyse . . . 146
5.3.4 Analyse des tâches . . . 148
5.3.4.1 Cas d'utilisation *Réaliser tâche* . . . 148
5.3.4.2 Cas d'utilisation *Gestion patient* . . . 153
5.3.4.3 Cas d'utilisation *Configuration du robot* . . . 153
5.3.5 Diagramme global des cas d'utilisation, structure dynamique et statique du *Système de contrôle TER* . . . 155
5.4 Identification des dangers et estimation du risque . . . 157
5.4.1 Analyse préliminaire des dangers . . . 158
5.4.2 Analyse des modes de défaillance des acteurs . . . 159
5.4.2.1 Modes de défaillance du *Site maître* . . . 161
5.4.2.2 Modes de défaillance de l'*Opérateur* . . . 165
5.4.3 Analyse des modes de défaillance des composants matériels . . . 167
5.4.4 Analyse des modes de défaillance des composants électroniques . . . 167
5.4.5 Analyse des modes de défaillance du logiciel . . . 169
5.4.6 Analyse des arbres de fautes . . . 175
5.5 Conclusion . . . 178

Synthèse, contributions majeures et perspectives **181**

A La notation UML **185**
A.1 Concepts transversaux 185
A.2 Diagramme des cas d'utilisation 186
A.3 Diagramme de classes 186
A.4 Diagramme d'états-transitions 188
A.5 Diagrammes d'interaction 189
A.6 Diagramme de composants 190
A.7 Diagramme de déploiement 190

B Compléments à l'analyse des modes de défaillance du robot TER **191**
B.1 Diagramme des cas d'utilisation simplifié 191
B.2 Diagramme de séquence principal du cas d'utilisation *Réaliser tâche* 192
B.3 Diagramme de séquence principal du cas d'utilisation *Contrôler l'exécution de la tâche* 194
B.4 Diagrammes de séquence du cas d'utilisation *Configuration du robot* 195

Index **199**

Bibliographie **201**

Table des figures

1.1 Exemple de tableau pour l'estimation du risque 26
1.2 Exemple d'utilisation de la terminologie 30
1.3 Activités de la gestion des risques, figure tirée de la norme ISO 14971 (2000) 32
1.4 Exemple de diagramme des niveaux d'acceptabilité du risque et de la zone ALARP 38
1.5 Illustration de l'acceptabilité du risque en fonction du coût, figure tirée de HSE (1999) 39
1.6 Exemple d'arbre de fautes 42
1.7 Exemple de tableau de l'AMDEC 42

2.1 Photo de l'extrémité d'un muscle artificiel 59
2.2 Coupe d'une extrémité d'un muscle artificiel avant le sertissage de la jupe métallique 60
2.3 Branchements entre muscle artificiel et son pré-actionneur 60
2.4 Force statique de plusieurs muscles à 5 bars 62
2.5 Site expérimental pour les tests d'un muscle en dynamique 63
2.6 Réponse en longueur et force dynamique d'un muscle à un échelon pour une charge de 15 kg 64
2.7 Puissance instantanée en réponse à échelon pour une charge de 15 kg 64
2.8 Forces dynamiques de deux muscles pour une masse soulevée de 10 kg . . . 65
2.9 Principe de l'actionnement d'une articulation au moyen de deux muscles antagonistes 66
2.10 Sens de rotation des 7 axes du bras anthropomorphe 67
2.11 Vue de dos du bras anthropomorphe 7 axes 68
2.12 Vue de face du bras anthropomorphe 7 axes 68
2.13 Vue de profil du bras anthropomorphe 7 axes 69
2.14 Manipulation d'un verre par le bras anthropomorphe 7 axes 70
2.15 Débattements des axes 1 à 3 du bras anthropomorphe 7 axes 71
2.16 Débattements des axes 4 à 7 du bras anthropomorphe 7 axes 72
2.17 Analyse des modes de défaillance de la tresse et du système de maintien . . . 73
2.18 Analyse du mode de défaillance du tube de caoutchouc 74

2.19 Réponse en longueur à une crevaison 75
2.20 Réponse en pression à une crevaison 75

3.1 Interactions entre les objets 83
3.2 Exemple de deux instances de la classe CapteurPosition 84
3.3 Évolution des versions du langage UML 85
3.4 Objets, classes et métamodèle 87
3.5 Développement piloté par les cas d'utilisation (CU) 88
3.6 Le modèle d'architecture à 4+1 vues, figure tirée de Kruchten (1999) 89

4.1 Vue globale de la démarche présentée dans ce chapitre 100
4.2 Modélisation d'un geste médical (a), et d'un cas de télé-médecine robotisée (b) 103
4.3 Exemple de diagramme des cas d'utilisation 104
4.4 Exemple de fiche de cas d'utilisation 104
4.5 Exemple de diagramme de séquence pour un scénario du cas d'utilisation *Réaliser tâche* de la figure 4.3 105
4.6 Interactions principales lors de l'utilisation d'un sous-système 106
4.7 Exemple de tableau pouvant servir pour l'analyse préliminaire des dangers . . 108
4.8 *Package Behavioral Elements*, diagramme tiré de la spécification 1.4 d'UML 110
4.9 Métamodèle des éléments *Interaction* et *Message* dans la spécification 1.4 d'UML 112
4.10 Éléments d'une interaction réalisée par l'envoi de deux messages 112
4.11 Exemple de tableau de l'AMDEC 115
4.12 Diagramme de séquence illustrant des interactions entre un site maître et un système esclave 117
4.13 Analyse d'un mode de défaillance du message *Controler_mouvements()* . . . 118
4.14 Diagramme d'états prenant en compte un mode de défaillance de *Contrôler_mouvements()* 119
4.15 Procédure de mise en route d'un système quelconque 119
4.16 Gestion du risque induit par les erreurs humaines 121
4.17 Exemple d'utilisation d'un diagramme de déploiement en informatique industrielle 122
4.18 Diagramme de classes représentant les dispositifs extérieurs au logiciel 122
4.19 Diagramme de blocs tiré de l'article de Dombre et al. (2001) 125
4.20 Diagramme de classes représentant les composants mécaniques d'une articulation 125
4.21 Package représentant les éléments du *Système de contrôle* de la figure 4.18 . . 128
4.22 Diagramme de classes d'un robot manipulateur d'une sonde quelconque pour l'examen d'un patient 131
4.23 Exemple d'arbre de fautes pour un effort de la sonde trop important sur le patient131

4.24 Résumé de la démarche présentée 133

5.1 Vue générale du système TER 139
5.2 Diagramme de déploiement du système d'échographie 140
5.3 Diagramme de classes des acteurs de l'examen échographique 140
5.4 Diagramme des cas d'utilisation de l'examen échographique 141
5.5 Diagramme des cas d'utilisation d'un robot 144
5.6 Diagramme des cas d'utilisation du site esclave 145
5.7 Diagramme de classes des acteurs 146
5.8 *Packages* et acteurs du système global 147
5.9 Diagramme des cas d'utilisation du système de contrôle esclave, que l'on note par la suite *Système de contrôle TER* 147
5.10 Vue de l'effecteur du robot esclave TER 148
5.11 Diagramme de séquence du scénario principal de *Réaliser tâche* 149
5.12 Cas d'utilisation en relation avec *Réaliser tâche* 150
5.13 Diagramme de séquence du contrôle de l'exécution de la tâche par l'*Opérateur* 152
5.14 Cas d'utilisation *Contrôler l'exécution de la tâche* 153
5.15 Diagramme de séquence de la configuration du robot 154
5.16 Diagramme des cas d'utilisation illustrant la relation *«include»* 155
5.17 Diagramme général des cas d'utilisation 156
5.18 Diagramme d'états du *Système de contrôle TER* 156
5.19 Diagramme de classes du *Système de contrôle TER* 157
5.20 Table d'estimation du risque 158
5.21 Table d'analyse préliminaire des dangers 160
5.22 Diagramme d'états illustrant l'implantation d'un arrêt d'urgence 161
5.23 Diagramme de séquence principal du cas d'utilisation *Réaliser tâche* 162
5.24 Utilisation du *package Communication* par le *Système de contrôle TER* 162
5.25 AMDEC du message *Contrôler_robot* émit par le *Site maître* 163
5.26 Diagramme d'états du Système de contrôle TER prenant en compte les modes de défaillance du message *Contrôler_robot* 164
5.27 AMDEC du message *Mettre la pression* 166
5.28 Système d'alimentation en air 166
5.29 Principaux composants mécaniques pour l'actionnement des anneaux de déplacement 167
5.30 AMDEC d'un muscle artificiel 168
5.31 Commande en boucle fermée avec un régulateur de type PIDa 168
5.32 AMDEC d'un capteur de position 169
5.33 AMDEC d'un convertisseur Intensité/Pression 170
5.34 AMDEC du Contrôleur TER 170

5.35 Diagramme de séquence des tests lors de l'initialisation (cas d'utilisation *Configurer Robot*) . . . 171
5.36 Contenu du *package Contrôleur TER* . . . 171
5.37 Diagramme de classes du logiciel . . . 172
5.38 Diagramme de séquence du logiciel du scénario principal de *Réaliser tâche* . 173
5.39 Diagramme de séquence du logiciel de l'asservissement en position . . . 173
5.40 AMDEC du message *recevoirConsigne()* . . . 174
5.41 AMDEC du message *envoyerCommande()* . . . 175
5.42 Arbre de fautes de l'élément racine *Pression trop importante sur le patient* . . 176
5.43 Arbre de fautes concernant l'installation du système . . . 176
5.44 Arbre de fautes illustrant l'introduction du sous-système d'arrêt d'urgence . . 177

A.1 *Package*, dépendance, note et stéréotype . . . 185
A.2 Acteur, cas d'utilisation et association . . . 186
A.3 Stéréotype «Acteur» . . . 186
A.4 Classe . . . 186
A.5 Association . . . 186
A.6 Agrégation, composition et cardinalité . . . 187
A.7 Généralisation et spécialisation . . . 187
A.8 États et transitions . . . 188
A.9 Emboîtement d'états . . . 188
A.10 Diagramme de séquence . . . 189
A.11 Diagramme de collaboration . . . 189
A.12 Composant . . . 190
A.13 Nœud et lien . . . 190

B.1 Cas d'utilisation du système de contrôle du site esclave de TER . . . 191
B.2 Réalisation de la tâche d'échographie . . . 192
B.3 AMDEC des messages émis par le patient lors de l'interaction *Réalisation de la tâche d'échographie* . . . 193
B.4 AMDEC du message *Contrôler_robot* émis par le site maître . . . 193
B.5 Contrôle de l'exécution de la tâche . . . 194
B.6 Configuration de la tâche . . . 195
B.7 AMDEC du message *Mettre en position initiale* . . . 196
B.8 Configuration du contrôleur . . . 196
B.9 AMEDC du message *Mettre la pression* . . . 197

Introduction générale

Les systèmes technologiques sont au cœur de nombreuses applications permettant d'aider l'humain dans des tâches dangereuses, trop complexes ou impossibles. Parmi ces applications, certaines peuvent blesser des humains, ou provoquer des dégâts sur les biens matériels ou l'environnement. De tels systèmes sont qualifiés de *systèmes à sécurité critique* et leur utilisation est conditionnée par la confiance que l'humain leur accorde. Cette confiance est d'autant plus importante que l'on assiste aujourd'hui à des transferts de responsabilités de l'humain vers les dispositifs technologiques. Cette problématique a conduit à l'émergence d'un nouveau domaine, la *sûreté de fonctionnement*, définie par Laprie (1992) comme « la propriété d'un système qui permet de placer une confiance justifiée dans le service qu'il délivre ». Ce concept permet de regrouper plusieurs propriétés que sont la sécurité-confidentialité (*security* en anglais), la fiabilité, la disponibilité, et la sécurité-innocuité (*safety*). Dans le cadre de notre étude, concernant l'intégrité des biens et des personnes, nous utiliserons le terme de sécurité en lieu de sécurité-innocuité. Les moyens utilisés pour garantir ces différentes propriétés consistent principalement en l'utilisation de techniques de conception, développées dans des secteurs de pointe comme l'aéronautique, le nucléaire, ou le spatial. Malgré ces moyens, la complexité grandissante des systèmes technologiques, induite notamment par l'apparition des systèmes dits socio-techniques, où l'humain et le dispositif technologique interagissent pour accomplir une tâche, rend impossible la garantie d'une sécurité absolue. Et, si dans sa définition première, le terme de sécurité revêt un caractère absolu, il est aujourd'hui utilisé pour exprimer une propriété relative. En effet, les concepteurs intègrent le fait qu'il subsiste pour toute application un risque résiduel. La sécurité est alors définie par la connaissance et l'acceptabilité de ce risque. Les concepteurs d'un système doivent donc être en mesure de gérer ce risque depuis les premières étapes du processus de développement jusqu'à l'utilisation.

La notion de risque, pressentie par la communauté des roboticiens qui se sont concentrés sur la notion de sécurité (Dhillon, 1991), est nouvelle dans le domaine de la robotique. En effet, depuis l'avènement des premiers automates et des manipulateurs simples pour les fonctions d'approvisionnement, jusqu'au robots manipulateurs polyvalents permettant le pick-and-place, la soudure par points, l'assemblage simple, et plus tard la peinture et la finition, ces machines se sont cantonnées à un milieu industriel protégé. Au sein des chaînes de fabrication de l'industrie automobile, de l'électro-ménager, de l'électronique grand public, ou de l'ensemble des secteurs de la mécanique, ces robots ont toujours été confinés à un espace

de travail interdit à l'humain. L'évolution des techniques et notamment de l'informatique, a permis d'adapter ces robots manipulateurs à des applications non manufacturières hors de l'usine, dans des environnements non protégés. Cette robotique de *service*, développée dans de nombreux domaines comme le spatial, le nucléaire ou le sous-marin, a connu dans les années 90 une dernière évolution vers des applications dites « amicales », évoluant dans un milieu humain. La robotique médicale est sans doute la plus spectaculaire de ces applications où l'exigence principale est la sécurité.

La complexité de ces applications et le fait de partager l'espace de travail avec l'humain, ou d'entrer en contact avec lui (voire à l'intérieur du corps humain), amène à classer ces dispositifs dans la catégorie des systèmes à sécurité critique. Ainsi, le concept de risque présenté précédemment, s'applique aux systèmes de la robotique de service évoluant dans un milieu humain. Pour traiter les risques, les ingénieurs doivent faire face à de nombreuses difficultés relevant de multiples domaines d'étude dont les facteurs humains, et ont à leur disposition de nombreuses techniques matérielles, logicielles, ou organisationnelles (liées au processus de développement). Choisir parmi ces techniques celles qui correspondent le mieux au problème, est complexe. Les concepteurs, guidés par leur savoir-faire, effectuent alors des choix souvent noyés dans le processus de développement, sans pouvoir les justifier vis-à-vis de la sécurité. Le travail que nous avons réalisé contribue à proposer aux ingénieurs des solutions pour appréhender cette problématique.

Tout d'abord, en partant de l'événement redouté, le dommage, nous étudions dans le chapitre 1, les concepts formant la base théorique de notre analyse. À partir de l'étude de cet effet non désiré, nous présentons le concept qui en dérive, le risque, puis ses causes et les moyens de le gérer. Dans le domaine de la robotique de service, cette notion de risque est encore nouvelle, et ce chapitre se consacre à intégrer les travaux de domaines transversaux. De plus, le cas de la robotique médicale, domaine de pointe de la robotique de service, est étudiée en estimant que cette analyse peut s'appliquer aux autres domaines de la robotique de service moins contraignants en terme de sécurité.

Ce chapitre montre que parmi les techniques de réduction du risque développées en robotique, un axe de recherche important concerne la découverte de nouveaux actionneurs moins lourds et moins rigides. Le poids et la compliance naturelle des muscles artificiels de McKibben développés depuis une dizaine d'années au sein du LESIA, semblent répondre à ces contraintes. Ainsi, le chapitre 2 analyse le risque lié à l'emploi de ces actionneurs.

L'utilisation des muscles artificiels n'étant qu'une technique parmi d'autre pour réduire les risques, les chapitres suivants présentent une démarche globale d'analyse du risque s'intégrant au processus de développement, et notamment en se basant sur l'utilisation du langage de modélisation orienté objet UML (*Unified Modeling Language*). Le chapitre 3 contient une présentation des principaux concepts de la modélisation orientée objet avec UML utilisés dans cette démarche, ainsi qu'un rapide état de l'art sur les études concernant les liens entre ce langage et la sûreté de fonctionnement. Le chapitre 4 est dédié à la présentation de cette démarche, et s'appuie sur les concepts présentés au chapitre 1. Il se limitera néanmoins à l'une

des activités de la gestion du risque, l'analyse du risque. Cette démarche se base sur la combinaison de techniques de l'ingénieur afin de pouvoir s'intégrer rapidement à tout processus de développement. Ainsi, des techniques comme l'AMDEC (Analyse des Modes de Défaillance, de leurs Effets et de leur Criticité) et l'analyse des arbres de fautes sont utilisées. De plus cette démarche intègre le concept de facteurs humains, rarement abordé par les concepteurs. La dimension cognitive de ce domaine est encore éloignée des sciences de l'ingénieur, et plus particulièrement en France (à l'opposé des pays anglo-saxons). Dans le contexte de la sécurité, nous nous limiterons cependant à trois de ses composantes fondamentales que sont l'analyse des tâches, l'allocation des tâches, et l'analyse des erreurs humaines.

Le chapitre 5 présente une application de cette démarche à l'analyse du risque d'un robot médical pour la télé-échographie actionné par des muscles artificiels. Ce projet, financé par le ministère de l'enseignement, de la recherche et de la technologie, intitulé TER pour Télé-Échographie Robotisée, initié en 1999, a regroupé plusieurs laboratoires, hôpitaux et industriels français.

L'aspect pluridisciplinaire de cette thèse m'a conduit à explorer des disciplines aux limites des sciences de l'ingénieur. Et, bien que l'application de cette thèse soit orientée vers la robotique de service, de nombreux éléments présentés dans ce mémoire peuvent s'appliquer à d'autres systèmes informatiques. Il s'agit donc avant tout d'un travail transversal, où la notion de risque vient s'inclure dans la théorie des systèmes, et où les spécialistes de la robotique ainsi que du langage UML, pourront trouver une ouverture vers la maîtrise de la sécurité.

Chapitre 1

Sécurité des systèmes de la robotique de service

1.1 Introduction

La robotique industrielle essentiellement dédiée à des processus manufacturiers a récemment utilisé la technologie de ses bras robots pour des applications hors de l'usine. Cette « robotique de service », et plus particulièrement sa composante de pointe, la robotique médicale (Dario et al., 1996), a fortement modifié la problématique de la sécurité des systèmes robotiques. Alors que les problèmes de sécurité des robots industriels relevaient essentiellement de défaillances des systèmes de protection (barrières, grillages, etc.), et que l'opérateur humain ne pénétrait l'espace de travail que pour des opérations de maintenance ou de programmation, la robotique de service nécessite de prendre en compte la proximité physique et l'étroite collaboration entre l'humain et le robot. L'impérieuse exigence de maîtriser la sécurité est d'autant plus urgente que les fonctionnalités et l'autonomie des robots de service ne cessent d'augmenter. Ainsi, le robot médical Robodoc[1], commercialisé pour effectuer le remplacement de la tête du fémur au niveau de la hanche (Pransky, 1997; Cain et al., 1993), réalise à l'issue de la planification de l'opération, la découpe et la préparation de la cavité fémorale en totale autonomie. Par ailleurs, on assiste aujourd'hui à l'apparition de systèmes de télé-médecine (médecine à distance) permettant la téléopération du robot par le spécialiste se trouvant à une grande distance du patient. Il existe aussi des systèmes de téléopération sur site, permettant au spécialiste d'effectuer des actions plus rapides et plus précises. Ainsi, dans un système comme AESOP (Sackier et Wang, 1994), le chirurgien guide avec sa voix les déplacements d'un endoscope, afin d'effectuer une opération minimalement invasive[2]. De

[1]ISS Inc, USA, http ://www.robodoc.com

[2]Cette technique opératoire minimise l'accès anatomique par le biais de petites incisions pour insérer un endoscope et des instruments chirurgicaux

façon plus générale, plusieurs projets de recherche, sur le modèle de l'arthroscopie[3], visent à réduire les opérations de chirurgie en miniaturisant les outils et en faisant appel à un guidage par endoscope. Des robots miniatures (ou micro-robots) pourraient répondre à cette demande, et permettre alors des opérations jusqu'ici impossibles, car trop dangereuses, comme de complexes opérations du cœur ou du cerveau.

Dans ce contexte, les utilisateurs (patients et médecins), mais également les personnes se trouvant dans leur environnement ou les autorités acceptant leur mise sur le marché, peuvent s'interroger à juste titre sur la confiance qui peut être accordée à de tels systèmes. La sécurité, un des facteurs de cette confiance, a longtemps été considérée comme une propriété résultant de l'utilisation d'un ensemble de techniques sans qu'il y ait de lien entre elles. Il existe néanmoins aujourd'hui une science de la sécurité, parfois appelée science du danger, science du risque, ou plus récemment cyndinique[4].

Bien que ces concepts soient étudiés depuis plusieurs années dans d'importants secteurs comme l'aéronautique ou le nucléaire, la spécificité de la robotique médicale nécessite une remise en question de ces concepts. Ce chapitre fonde ses analyses sur la notion d'événement non désiré qu'est le dommage (section 1.2). À partir de cette notion, et de l'ensemble des normes qui lui sont dédiées, les parties suivantes traitent des causes des dommages (section 1.3), des moyens que l'on peut mettre en œuvre pour les traiter (section 1.4) et des techniques appropriées à ces moyens (section 1.5 et section 1.6). Nous abordons ensuite la notion d'assurance que l'on peut obtenir concernant la sécurité de ces systèmes grâce à la certification (section 1.7).

Après un premier travail de synthèse sur la sécurité des robots (Guiochet, 2001), nous avons par la suite privilégié le domaine de la robotique médicale (Guiochet et al., 2003a). Nous pouvons en effet penser que nos analyses s'applique largement aux autres domaines de la robotique de service moins contraignants en terme de sécurité.

1.2 Effets non désirés : les dommages

Afin d'analyser la notion de sécurité inhérente aux robots médicaux, il nous a paru fondamental de revenir aux notions de base, et de comprendre le mécanisme d'apparition d'un accident. Pour tout système, les effets non désirés sont définis par la notion de dommage.

[3]Examen d'une cavité articulaire au moyen d'une caméra appelée endoscope permettant par la suite d'effectuer des opérations sans ouvrir comme en chirurgie

[4]http ://www.cindynics.org

1.2.1 Dommage

1.2.1.1 Définition

La notion de dommage est particulièrement stable, car il est possible de retrouver la même définition dans le domaine médical (ISO 14971, 2000), en robotique (Prasad, 1988) et au sein des normes génériques (IEC 60300-3-9, 1995). La définition générique du ISO/CEI Guide 51 (1999) est utilisée dans le domaine médical où le dommage est défini comme :

Dommage	*Blessure physique ou une atteinte à la santé des personnes, ou dégât causé aux biens ou à l'environnement*

Il est intéressant de comparer cette définition avec celle de la sécurité fréquemment utilisée en robotique industrielle, définie comme la *prévention de dégâts sur l'humain, le robot et les éléments avec lesquels le robot interagit* (Dhillon, 1991). On retrouve deux composantes de la notion de dommage :

- les *personnes* qui peuvent être les patients, les spécialistes ou les assistants, identifiés ici comme les humains,
- les *biens*, correspondants aux dispositifs médicaux utilisés (le robot lui-même, les outils, les appareils de mesures, etc.) ,

Cependant, dans la définition adoptée dans le domaine médical, on trouve une composante supplémentaire qui est l'environnement. En dehors des biens et des personnes, il est en effet possible d'identifier les dommages sur l'environnement en terme de destruction, pollution, etc. Ceci est particulièrement important pour les nouvelles applications de robotique de service dont la tâche peut influer sur l'état de l'environnement.

Le dommage peut ensuite être qualifié par sa nature (brûlure, écrasement, coupure, etc.) et mesuré par sa gravité et sa probabilité d'occurrence.

1.2.1.2 Gravité d'un dommage

Pour des systèmes robotiques médicaux, la notion de gravité est identique à de nombreux autres domaines technologiques. Elle évalue la nuisance des dommages. Cependant, les normes relatives aux dispositifs médicaux ne prescrivent aucune échelle de graduation de ces nuisances, et laissent ainsi le choix aux fabricants (à l'opposé d'autres domaines où les niveaux sont prédéfinis). La liste ci-dessous donne un exemple de métrique sur la gravité des dommages sur les seuls humains :

- Catastrophique : décès d'une ou plusieurs personnes
- Majeur : blessures ou maladies graves, infirmité permanente
- Mineur : blessures ou maladies mineures, nécessitant un traitement médical
- Minime : légères blessures relevant des premiers soins (ne nécessitant pas un traitement médical)
- Négligeable : incident n'exigeant aucun traitement médical

Notons que dans cette échelle de mesure, le décès d'une ou de plusieurs personnes est classé comme *catastrophique* alors que dans d'autres domaines technologiques le niveau catastrophique est réservé à l'occurrence du décès de plusieurs personnes. C'est le cas par exemple pour les dommages d'origine naturelle (séismes, etc.). Pour les dispositifs médicaux, il est évident que l'enjeu est la santé d'un patient, et que son décès est donc catastrophique. La notion de gravité en fonction du nombre de morts est par conséquent un concept inexistant en médecine. Il existe malgré tout un exemple de dispositif médical ayant provoqué un ensemble de décès (six), le Therac-25, utilisé aux États-Unis, qui a provoqué un ensemble d'effets secondaires à cause d'une trop forte exposition de rayons X. Ce cas est présenté plus en détail dans la section 1.4.3.3.

Le terme de *sévérité* est parfois utilisé pour exprimer la gravité. Cela provient en partie de la comparaison avec le terme anglais *severity*, mais son utilisation, « critiquée » d'après le *Nouveau Petit Robert* (1995), ne fait qu'ajouter une définition supplémentaire au domaine du risque déjà lourd en synonymes.

1.2.1.3 Occurrence d'un dommage

En plus de la gravité, un dommage est qualifié par sa fréquence d'occurrence. Dans la norme médicale ISO 14971 (2000), la *probabilité d'un dommage* est exprimée en terme d'*occurrence d'un dommage*. Même si cette mesure n'est ici que qualitative (fréquent, occasionnel, etc.), le terme de *probabilité* est conservé. Cette norme évoque le fait qu' « une bonne description qualitative est préférable à une inexactitude quantitative ». Ce point de vue que l'on retrouve dans beaucoup d'ouvrages aujourd'hui, a un aspect novateur par rapport aux prescriptions qui ont été faites lors des débuts des études de sûreté de fonctionnement. Il provient en particulier de la difficulté d'estimer les probabilités obtenues selon trois approches :

- utilisation de données historiques
- utilisation de techniques d'analyse ou de simulations
- utilisation de jugements d'experts

En robotique médicale, il n'existe pas à notre connaissance de données historiques exploitables. L'absence de telles informations pour ces systèmes pourrait être palliée en se référant à des données sur des dommages liés aux contrôleurs de robots industriels, des dommages causés par l'utilisation de logiciels dans le milieu médical, des dommages induits par l'utilisation de dispositifs médicaux autres que des robots, etc.

La liste ci-dessous fournit un exemple de niveaux de probabilité d'occurrence, utilisables pour une estimation qualitative, on donne malgré tout une fréquence indicative par année indiquée entre parenthèses :

- Fréquente (>1)
- Probable ($1 - 10^{-1}$)
- Occasionnelle ($10^{-1} - 10^{-2}$)
- Rare ($10^{-2} - 10^{-4}$)

– Improbable ($10^{-4} - 10^{-6}$)
– Invraisemblable ($<10^{-6}$)

Dans cette classification, *Fréquente* indique une fréquence d'occurrence du dommage plusieurs fois par an, et *Invraisemblable* correspond à une occurrence tous les 1 000 000 ans. Cependant, selon chaque application, cette classification peut être modifiée, certaines utilisant des mesures des fréquences comprises entre une occurrence journalière et une toute les 5 ans. Comme précédemment signalé, et contrairement à d'autres domaines présentant des risques de dommages, il n'existe pas dans le domaine médical de norme spécifiant des métriques sur les fréquences d'occurrence : le choix est laissé aux concepteurs. Pour les systèmes technologiques complexes comme ceux de l'aéronautique, la limite inférieure est une probabilité de l'ordre de 10^{-9} occurrence par an correspondant à la probabilité d'apparition d'une catastrophe naturelle. L'échelle de valeur que l'on a donnée ci-dessus n'est qu'un exemple d'illustration pour bien exprimer ce qu'est une probabilité d'occurrence. Il est néanmoins évident que lors de l'utilisation d'un robot médical, cette estimation sera fortement dépendante du nombre d'intervention au cours d'une année, et que le nombre d'occurrence annuelle de dommage n'est pas significatif. La plupart de normes sur le sujet évite ce problème en laissant aux concepteurs le choix de la définition, et donc du calcul, des niveaux de probabilité.

1.2.2 Notion de Risque

1.2.2.1 Définition

Afin de rendre compte de l'interaction entre la gravité et l'occurrence d'un dommage, la notion de risque a été introduite. La définition de la norme médicale ISO 14971 (2000) est identique à celle du ISO/CEI Guide 51 (1999) :

Risque	*Combinaison de la probabilité d'un dommage et de sa gravité*

La notion de risque connaît des fluctuations dans sa définition et est parfois présentée comme la probabilité du *danger* (par opposition avec la probabilité d'occurrence du *dommage*) combinée avec la gravité du dommage. C'est notamment le cas dans les versions précédentes des normes médicales sur le risque (EN 1441, 1997; Pr ISO 14971, 1999). Dans le domaine médical, c'est la notion de probabilité du *dommage* qui est aujourd'hui utilisée. On définit plus généralement pour un risque la probabilité d'un dommage en exprimant, par exemple, la probabilité qu'il y ait un décès ou des complications ou des effets secondaires, sans préciser quels événements ont provoqué ce dommage. La détermination des circonstances de l'apparition du dommage est illustré par les notions liées au danger exposées ultérieurement (cf. 1.3).

1.2.2.2 Mesure du risque

Afin de quantifier le niveau du risque, il est possible pour chaque couple (probabilité, gravité) de définir un niveau. Certains couples peuvent être alors comparés, et choisis de même

Fréquence d'occurrence	Fréquence indicative (par année)	Gravité du dommage				
		1 Catastrophique	2 Majeure	3 Mineure	4 Minime	5 Négligeable
Fréquente	>1	H	H	H	H	I
Probable	$1\text{-}10^{-1}$	H	H	H	I	I
Occasionnelle	$10^{-1}\text{-}10^{-2}$	H	H	I	I	L
Rare	$10^{-2}\text{-}10^{-3}$	H	I	I	L	T
Improbable	$10^{-4}\text{-}10^{-6}$	I	I	L	T	T
Invraisemblable	$<10^{-6}$	I	L	T	T	T

FIG. 1.1 – Exemple de tableau pour l'estimation du risque

niveau. Sur la base des classifications de gravité et probabilité d'occurrence présentées en sections 1.2.1.2 et 1.2.1.3, le tableau de la figure 1.1 estime le risque suivant quatre niveaux, H (risque fort), I (risque intermédiaire), L (risque faible), et T (risque insignifiant). La probabilité est exprimée en fréquence d'occurrence, plus facilement mesurable dans une démarche qualitative.

1.2.2.3 Risque acceptable

La notion de risque est essentielle pour caractériser la confiance attribuée à un système. En effet, si on admet souvent comme potentiels des dommages sévères, seule leur faible probabilité d'occurrence nous les font accepter. Par exemple, nous continuons à prendre l'avion malgré les accidents possibles du fait que la probabilité d'un écrasement conduisant aux décès des passagers est extrêmement faible.

Nous établissons généralement cet arbitrage en fonction des risques que nous encourront par ailleurs, comme ceux induits par des phénomènes naturels : tremblements de terre, avalanches, inondations, etc. On définit ainsi la notion de *risque acceptable* dérivée du ISO/CEI Guide 51 (1999), comme *un risque accepté dans un contexte donné basé sur des valeurs courantes de notre société*. Notons que l'acceptabilité concerne le risque et non la gravité du dommage ou sa probabilité d'occurrence considérées séparément.

Cette définition souligne également le fait que l'acceptabilité dépend de *valeurs courantes de notre société* souvent fondées sur des données associées à des phénomènes naturels. Ainsi on accepte de prendre le risque de mourir en prenant l'avion si la probabilité de ce décès par cette cause est identique voire inférieure à la probabilité de décès induit par un séisme ou une crise cardiaque (pour un corps sain).

La définition précise par ailleurs que l'acceptabilité est *fonction du contexte*. Ce contexte peut tout d'abord caractériser l'état d'un savoir sur des pratiques ou sur une technologie de mise en œuvre. Ainsi le risque accepté par les passagers des premiers aéronefs était-il bien supérieur à celui actuellement toléré par les passagers des avions de ligne.

Le contexte exprime également l'apport escompté malgré le risque pris. Ainsi, une opération chirurgicale ou la prise d'un médicament récemment mis au point a une probabilité non négligeable de conduire à des dommages graves. Leurs risques sont acceptés du fait de la guérison espérée ou des dommages inéluctables causés par une évolution naturelle de la maladie.

Dans sa définition idéale, un risque associé à un robot médical serait acceptable si le pronostic des patients s'améliorait (ceci est vrai pour tout dispositif médical). Au vu de la complexité de l'évaluation du risque associé aux applications robotiques médicales, il est difficile d'appliquer cette simple règle. Nous reviendrons sur cette notion dans la partie 1.4.4 où nous présenterons les moyens qui existent actuellement pour spécifier un risque acceptable.

1.2.3 Sécurité

Le concept de sécurité est devenu un des enjeux les plus importants pour de nombreux domaines technologiques. Il est parfois utilisé pour exprimer *l'absence d'accident ou de perte*. De la même manière, dans le domaine de la robotique industrielle, la sécurité est définie comme la prévention de dégâts sur l'humain, le robot et les éléments avec lesquels le robot interagit (Dhillon, 1991). Ces définitions donnent une dimension absolue à la sécurité. Or le domaine médical prouve bien qu'il n'y a qu'une sécurité relative ; il existe toujours un risque résiduel. Par exemple, sur un nombre donné d'opérations chirurgicales, il est établi qu'il existe un certain nombre de rejets, d'infections, ou d'autres complications allant parfois jusqu'au décès du patient. Au regard des bienfaits pour la majorité des patients, la société permet de prendre ce risque. Dans ce contexte, le terme *sécurité* est employé dans le domaine médical pour exprimer un niveau de sécurité atteint en réduisant le risque à un niveau acceptable. C'est la définition du ISO/CEI Guide 51 (1999) que nous adopterons pour la suite de notre étude :

Sécurité — *Absence de risque inacceptable*

Le *risque acceptable* est, par conséquent, le résultat d'un équilibre entre l'idéal de la sécurité absolue, les exigences auxquelles doit répondre le système concerné, et des facteurs comme le bénéfice pour les patients, le coût effectif, les règles et les conventions de la société concernée. La sécurité, dans cette optique, est donc un concept lié à une connaissance du niveau de risque accepté. Ainsi, pour beaucoup d'opérations chirurgicales, la gravité et la probabilité des dommages possibles (donc le risque) sont des informations que le praticien donne au patient. Ce dernier fait alors confiance ou non au niveau de sécurité de l'intervention médicale.

Il est important de noter que même si certaines études de fiabilité et de sécurité se recoupent, ces deux notions sont bien distinctes au sens où l'une s'intéresse à la continuité du service délivré et l'autre à la notion de dommage. Il est tout à fait possible d'envisager une solution très fiable mais avec un niveau de sécurité très bas, et vice versa. Par exemple, un

système d'échographie dont la mise sous tension ne provoque aucun effet est peu fiable mais très sûr.

1.3 Causes : les dangers

1.3.1 Notion de danger

Afin de prévenir les dommages, la notion de danger est introduite en tant que concept en amont. Le danger a été défini de différentes manières. Dans MORT (Management Oversight and Risk Tree présenté par Johnson (1980)), il est principalement caractérisé par un transfert d'énergie. De manière similaire, en robotique industrielle, le danger se rapporte à une « accumulation d'énergie aboutissant à un accident » (Kumamoto et al., 1986). Le danger a aussi été défini comme une propriété inhérente d'un objet, d'une substance ou d'un sous-système qui a la capacité de provoquer un dommage. Dans le domaine médical, il est évident que les dangers ne se réduisent pas à un transfert d'énergie ou à une propriété inhérente. On peut citer comme exemple le rejet d'un organe transplanté, le développement d'un effet secondaire ou le fonctionnement irrégulier (voire l'arrêt) d'un pacemaker.

Dans le cadre d'un système automatisé comme un robot médical, il est important de prendre en compte l'état dans lequel le système se trouve et les conditions d'environnement pour définir un danger. Ces considérations se rapprochent de la définition donnée pour les systèmes informatiques par Leveson (1995), où le danger (*hazard*) est défini par « un état ou un ensemble de conditions d'un système (ou d'un objet), qui, couplés avec d'autres conditions de l'environnement du système (ou de l'objet) mèneront inévitablement à un accident ».

Les normes IEC 60300-3-9 (1995) et EN 1441 (1997) définissent le danger comme « source potentielle de dommage ». Cette définition se différencie de celle de Leveson en posant une condition de potentialité sur l'aboutissement à un dommage. La notion de source est dans ce cadre ambiguë et peut être interprétée de différentes manières. Il est en effet parfois difficile de faire la différence entre le danger, sa cause ou ses effets. Par exemple de telles normes fournissent des listes de dangers communs aux dispositifs médicaux pour aider le concepteur à les identifier, et l'on peut retrouver dans ces listes à la fois des effets et des causes. C'est en cela que le terme *source* reste large et peut désigner une défaillance, une faute ou une erreur dans la terminologie de Laprie et al. (1995). Le concept de *source* ne souligne pas non plus le fait que c'est une combinaison de circonstances qui peuvent aboutir à un dommage.

1.3.2 Phénomène dangereux et situation dangereuse

Les disparités entre les définitions précédentes ont été réduites avec la publication de la dernière version de la norme sur la gestion du risque pour les dispositifs médicaux (ISO 14971,

2000)[5]. Le terme de *phénomène dangereux* a été substitué au terme de *danger* et la notion de *situation dangereuse* a été introduite.

Phénomène dangereux — *Source potentielle de dommage*

On regroupe sous cette appellation l'ensemble des sources et des facteurs pouvant contribuer à la création d'un dommage. Ainsi, un bord coupant est un phénomène dangereux, mais cela ne provoquera pas obligatoirement un dommage. En revanche, en introduisant la notion de *situation dangereuse*, les normes utilisées ici rejoignent la définition de danger (*hazard* en anglais) énoncée en théorie des systèmes informatiques (Leveson, 1995).

Situation dangereuse — *Situation dans laquelle des personnes, des biens ou l'environnement sont exposés à un ou plusieurs phénomènes dangereux*

1.3.3 Événement dommageable

Le terme d'*événement dommageable* est défini dans le ISO/CEI Guide 51 (1999) comme *l'événement déclencheur qui fait passer de la situation au dommage*. Ce concept n'est pas repris dans les normes dédiées aux dispositifs médicaux. En revanche, pour une application robotique il peut être intéressant d'utiliser ce concept. Il permet en effet de spécifier l'événement qui fait passer le système socio-technique (système incluant les humains et la technologie) d'une situation dangereuse à un dommage spécifique. En robotique médicale, de tels événements sont principalement des interactions entre l'outil du robot et une partie du corps du patient (coupure, arrachement, étirement, brûlure, etc.) dont résulte un dommage (saignements, destruction de liaisons, destruction de fonctionnalités, perforation, etc.).

1.3.4 Incident et accident

Les analyses de sécurité utilisent deux notions importantes, pourtant absentes de la norme médicale ISO 14971 (2000), mais qui apparaissent dans la majorité de la littérature sur le domaine. On définit alors la notion d'accident en s'inspirant de la définition proposée par Leveson (1995) :

Accident — *Événement non désiré et d'occurrence non prévue résultant en un niveau de dommage spécifié*

Cette notion repose sur celle de l'événement dommageable. Dans le cas d'un accident, l'événement dommageable produit un dommage de gravité non nulle ou non négligeable. À l'opposé, un événement dommageable produisant un dommage de gravité nulle ou négligeable (en général dû à des circonstances extraordinaires, on peut alors parler de chance) produira ce que l'on définit par un *incident*.

[5]Cette norme vient remplacer la EN 1441 (1997)

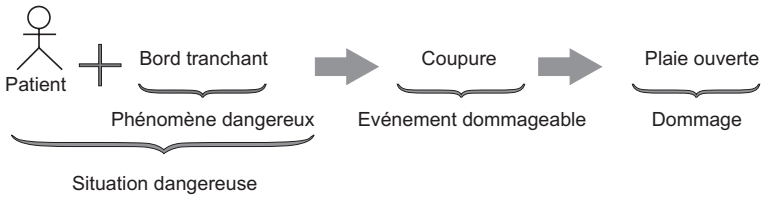

FIG. 1.2 – Exemple d'utilisation de la terminologie

Incident — *Événement qui ne conduit pas à des pertes, mais qui a le potentiel de créer des dommages en d'autres circonstances*

Ainsi un mouvement intempestif d'un bras de robot qui ne touche pas d'humain (ni de biens) alors que ceux-ci se trouvaient dans son espace de travail, est un incident s'il n'y a aucun dommage. C'est le cas si un robot frôle avec une grande vitesse la tête d'un humain. Même si ces définitions restent proches de leur utilisation dans le langage courant, elles peuvent parfois créer la confusion avec d'autres termes comme le *danger*. Dans le domaine médical les normes ne font pas référence à ces notions alors que dans le domaine des systèmes à sécurité critique (nucléaire, avionique, etc.) elles sont couramment utilisées.

1.3.5 Exemple d'utilisation des notions liées au risque

Les exemples de la robotique industrielle considèrent principalement les flux d'énergie comme phénomènes dangereux. En prenant l'exemple d'une coupure (événement dommageable), on peut donner une illustration différente des concepts liés au danger. La figure 1.2 illustre le fait qu'une situation dangereuse conduira à un dommage s'il existe un événement dangereux (ou dommageable).

L'annexe D de la norme ISO 14971 (2000) fournit un ensemble « d'exemples de phénomènes dangereux possibles et des facteurs qui y contribuent ». La classification présentée ne met pas en valeur les différents concepts que nous venons d'introduire, et il est possible de trouver à un même niveau de classification des causes de défaillances, des défaillances, des types génériques de sources de dommages, des propriétés inhérentes à des substances, etc. Le propos de cette norme n'est donc pas de donner une vue claire de la terminologie utilisée grâce à des exemples, mais de fournir une liste permettant éventuellement aux concepteurs de trouver des phénomènes dangereux ou des facteurs qui y contribuent, qui n'auraient pas été identifiés. On y trouve des termes génériques comme par exemple des attributs de l'énergie tels que l'électricité, la chaleur, la force mécanique, le rayonnement et dans le même temps des causes de ces transferts énergétiques comme les pièces mobiles, les mouvements intempestifs, les masses suspendues, etc. (ISO 14971, 2000, p.22). À partir de cette liste, il est tout de même possible de dresser pour les dispositifs médicaux une liste de phénomènes dangereux liés à un sous-système :

– propriétés inhérentes dangereuses (un bord tranchant, un gaz toxique)

– états dangereux du sous-système (en mouvement, masse suspendue)
– défaillances dangereuses incluant les erreurs humaines
– autres

1.3.6 Analyse du danger

Le terme de danger peut être utilisé en tant que notion générale regroupant trois concepts fondamentaux : le phénomène dangereux, la situation dangereuse, et l'événement dommageable. L'étude d'un accident peut alors être menée comme une étude globale faisant appel à ces trois événements, parfois qualifiée d'*analyse du danger*, sachant que son but est de fournir une base pour l'analyse du risque.

Le premier point concerne l'analyse de l'événement dommageable puisqu'il est le dernier événement apparaissant avant l'accident ou l'incident. L'identification est directe et peut s'appuyer sur les données historiques, le jugement d'experts ou des techniques d'analyse. Il est alors possible d'étudier le risque associé à cet événement. L'analyse des situations dangereuses est plus délicate, et peut se faire par une connaissance plus avancée des sous-systèmes. Il est aussi possible de calculer la gravité du dommage associé et la probabilité d'occurrence de cette situation. On obtient ainsi le risque de la situation dangereuse. Enfin, pour chaque phénomène dangereux, il est aussi possible de calculer le risque, toujours en considérant le dommage induit.

Le risque d'un dommage est ensuite calculé en combinant les différents résultats des analyses liées à ce dommage. Par exemple, en considérant le cas catastrophique du décès d'un patient, il est possible d'identifier plusieurs événements dommageables, situations dangereuses et phénomènes dangereux induisant ce dommage. Pour un niveau de gravité du dommage donné, la probabilité d'occurrence du dommage est ensuite estimée en fonction des différentes probabilités d'occurrence de ces trois éléments. Afin de guider l'étude de ces éléments un processus de gestion du risque est présenté dans la partie suivante.

1.4 Moyens : la gestion du risque

1.4.1 Vue d'ensemble

La figure 1.3 donne une représentation schématique des principales étapes du processus de gestion des risques. Ce processus correspond à celui présenté dans la norme ISO 14971 (2000) tiré du ISO/CEI Guide 51 (1999). Ce schéma, dédié aux dispositifs médicaux se retrouve en partie dans des normes génériques comme la IEC 60300-3-9 (1995). Il est aujourd'hui accepté d'utiliser cette approche dans de nombreux domaines technologiques. Une étape supplémentaire est parfois présente, concernant la gestion des informations post-production (informations concernant les risques, guides d'utilisation, etc.). Avant de détailler chaque étape de ce

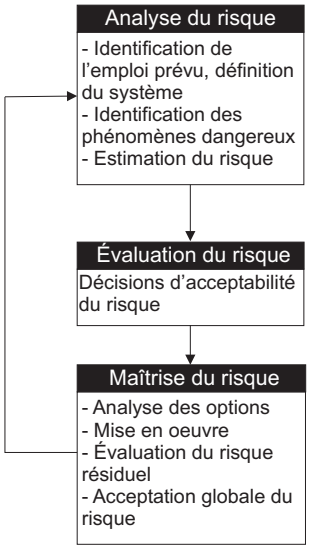

FIG. 1.3 – Activités de la gestion des risques, figure tirée de la norme ISO 14971 (2000)

processus dans les sections 1.4.3, 1.4.4 et 1.4.5, nous abordons la question essentielle des facteurs humains au sein de la gestion du risque.

1.4.2 Facteurs humains et gestion du risque pour des systèmes de la robotique médicale

La présence d'humains dans l'environnement des systèmes de la robotique de service et en particulier de la robotique médicale, en fait des systèmes dits *socio-techniques*. La sécurité de tels systèmes ne dépend pas uniquement de la défaillance de ses composants. De plus, l'humain ne peut être réduit à un simple composant ayant un taux de défaillance. Il peut, par exemple, stopper le système en cas de défaillance et finir une opération chirurgicale de manière classique. Son rôle dans l'utilisation du système est une composante particulière qu'il convient d'analyser. Ainsi, aujourd'hui, la sécurité est généralement obtenue par une analyse des facteurs de risque principalement dus aux défaillances, mais aussi par une analyse des *facteurs humains*.

Dans le milieu industriel, les facteurs humains sont pris en compte au sein du processus de développement. Mais bien que ce domaine cristallise de nombreuses recherches, il est encore difficile de trouver dans les processus de développement des robots médicaux, des éléments y faisant référence. La différence de langage entre ingénieurs et spécialistes du domaine cognitif est sans doute une des raisons de ce problème. On peut noter, par exemple, l'abandon d'un projet de norme, la CEI 300-3-8, mentionnée dans la IEC 60300-3-9 (1995), qui devait traiter de l'appréciation de la fiabilité humaine.

Dans le domaine de la robotique beaucoup d'études portent sur des aspects très techniques liés aux interactions entre humains et robots du point de vue des ordres envoyés au robot. Par exemple, différents modes de téléopération comme le retour d'effort ou la commande par

la voix sont étudiés. Dans le domaine médical, les facteurs humains concernaient principalement l'étude de l'erreur humaine (Felciano, 1995), sujet largement abordé par les sciences cognitives (Reason, 1990). Dans le contexte de la robotique médicale, nous nous intéresserons essentiellement aux facteurs humains qui concernent l'utilisation d'un système de robotique médicale. Nous n'aborderons pas les facteurs humains concernant la fabrication du système, ni ceux dévolus exclusivement au domaine médical comme l'erreur de diagnostic.

Plusieurs approches existent pour exprimer les interactions entre l'analyse de la sécurité (correspondant à la gestion du risque) et l'analyse des facteurs humains. Laprie et al. (1995, p65-66) classent les moyens pour la sûreté de fonctionnement des systèmes socio-techniques suivant quatre activités. La sécurité, en tant qu'attribut de la sûreté de fonctionnement, est ainsi augmentée par ces moyens. Les aspects de la sécurité liés aux facteurs humains utilisent ici la notion d'erreur humaine pour la classification de ces activités :

- La *prévention des erreurs humaines* a trait principalement à la recherche d'une répartition des tâches entre humains et technologie qui permette de confier aux humains des fonctions et des tâches homogènes et cohérentes.
- La *prévision des erreurs humaines* concerne les méthodes pour estimer la présence, la création et les conséquences des erreurs humaines.
- L'*élimination des causes d'erreurs humaines* concerne la réduction des causes d'erreurs en mettant l'humain en situation réelle ou en simulation.
- La *tolérance aux erreurs humaines* concerne l'introduction de mécanismes permettant au système socio-technique de continuer sa fonction en dépit des erreurs.

Cette classification permet, grâce à la terminologie utilisée, de donner une vision claire des moyens disponibles concernant les facteurs humains. Mais elle n'indique aucun enchaînement d'étapes et aucune interaction avec l'analyse des risques.

Daouk et Leveson (2001) exposent les étapes de la fabrication d'un système en mettant en parallèle les étapes et les interactions entre trois domaines : le processus de développement, l'analyse de la sécurité et l'analyse des facteurs humains. Leurs travaux font apparaître des redondances ou des activités communes à l'analyse de la sécurité et à l'analyse des facteurs humains mais conservent la séparation entre ces deux domaines. De plus, il est intéressant de noter que dans un ouvrage de référence sur la sécurité des systèmes comme celui de Leveson (1995), la notion de facteurs humains n'est pas utilisée, et que seule l'erreur humaine est abordée.

Dans un rapport de la Food and Drug Administration (2000), les facteurs humains concernent l'identification et la maîtrise des dangers relatifs à l'utilisation des dispositifs médicaux, et sont à ce titre intégrés dans la gestion du risque. Nous reprenons cette proposition, afin de l'appliquer à la robotique médicale, au sein des étapes de la gestion du risque. Cette démarche se retrouve dans un rapport de la HSE (2001b), où il est proposé d'inclure les facteurs humains à la norme IEC 61508 (2001) dédiée aux dispositifs électriques, électroniques et programmables relatifs à la sécurité. Ce rapport reprend étape par étape le cycle de vie de la sécurité vu par cette norme et propose l'inclusion d'activités propres aux facteurs humains.

1.4.3 Analyse du risque

Première étape de la gestion du risque, l'analyse du risque constitue le cœur du processus. Dans la terminologie des normes précédentes (EN 1441, 1997), l'analyse du risque était même parfois considérée comme le processus entier.

1.4.3.1 Analyse du risque et processus de développement

Après une synthèse des aspects normatifs dans les domaines ferroviaire et aéronautique, Papadopoulos et McDermid (1998) proposent d'utiliser un processus d'évaluation de la sécurité (Safety Assessement Process) qui prend comme premier point d'entrée une analyse des risques. Ce processus est parallèle au processus de développement avec lequel des informations sont continuellement échangées. Il est évident qu'une analyse de la sécurité dépend du développement du système mais impose aussi certaines modifications (conception, utilisation, etc.). Leveson (1995) attribue ainsi deux finalités majeures à une analyse du risque : la participation au processus de développement et la production d'informations en vue de la certification. Le cadre légal actuel renverse cette situation, puisque ce sont les règles de la certification qui imposent la présence d'une analyse du risque dans le processus de développement. Cela implique notamment que ce processus comme tout processus actuel de développement soit itératif, continu et incrémental et n'intervienne donc pas seulement en début de fabrication ou à des étapes fixes du développement.

1.4.3.2 Définition du système et de l'utilisation prévue

Le premier point de l'analyse du risque concerne la description de l'emploi prévu du dispositif. Cette étape a pour but de fournir une base pour l'identification des phénomènes dangereux. D'un point de vue générique il est conseillé dans cette étape de fournir les éléments suivants :

- une description générale,
- une définition des frontières du système, et une description de ses interfaces,
- une définition de l'environnement,
- une description des flux d'énergie, de matières et d'informations aux travers des limites,
- une définition des fonctions couvertes par l'analyse du risque.

La norme ISO 14971 (2000) fait aussi apparaître des éléments faisant référence aux facteurs humains (sans les nommer) pour cette activité. Il est par exemple conseillé de « décrire l'environnement d'utilisation du dispositif, de définir qui se charge de l'installation du dispositif, et si le patient peut contrôler le dispositif médical ou influer sur son utilisation ». Dans ces considérations on met en valeur, d'une part, l'importance de la description de l'utilisation en prenant en compte les interactions avec les humains, et d'autre part, la nécessité d'effectuer une allocation des tâches entre les différents humains et le dispositif. Cette distribution de la charge de travail se fait en fonction des performances humaines. Cette norme aborde ce der-

nier point en évoquant l'importance des facteurs tels que « l'utilisateur prévu, ses capacités physiques et mentales, ses compétences et sa formation ». L'analyse de la tâche et l'allocation de la charge de travail sont des activités que l'on trouve dans la littérature sur les facteurs humains, mais qui n'apparaissent pas explicitement dans les normes dédiées au risque.

Dans le cas des robots médicaux il convient de se poser la question fondamentale : quelles tâches ou parties de tâches le robot et l'humain sont-ils les plus à-même de réaliser ? Alors que pour les robots industriels la tâche s'exécutait entièrement soit par l'humain soit par le robot, le problème se pose différemment en robotique médicale. En effet, pour une même tâche, le spécialiste et le robot peuvent collaborer de différentes façons ; le lecteur pourra se référer au classement des robots médicaux en fonction du type d'interaction avec le spécialiste, présenté par Troccaz (1999) et Cinquin (1993). Pour le développement de ces nouvelles applications, l'aspect d'analyse de la tâche avec prise en compte du comportement humain est devenu une étape complexe. Les codes du domaine médical et la complexité des tâches telle qu'une opération chirurgicale rendent ce travail délicat. Aujourd'hui c'est un travail qui ne peut se faire que par une collaboration étroite entre ingénieurs et spécialistes du domaine médical.

1.4.3.3 Identification des phénomènes dangereux

L'étape d'identification des phénomènes dangereux connus et prévisibles est basée sur la définition du système. De nombreuses activités peuvent être utiles à l'identification des dangers dans des systèmes technologiques comme les robots médicaux telles que (liste non exhaustive) :

- la consultation des bases de données sur les expériences, les rapports sur les accidents et les incidents,
- l'utilisation de listes comme celle de la norme ISO 14971 (2000),
- l'examen des sources et des transferts d'énergie,
- l'estimation des matières dangereuses (substances, rayons, systèmes de pression, etc.),
- la consultation des analyses de risque d'autres dispositifs,
- les sessions de *brainstorming*, consultations d'experts,
- l'examen des interactions entre le patient, le robot et le spécialiste,
- l'utilisation de techniques d'analyse.

Les techniques utilisées pour l'identification permettent soit de partir de phénomènes dangereux, et d'en déduire des situations et des événements dommageables (on parle alors de techniques *forward*), soit de partir des dommages et des événements dommageables et de retrouver les situations puis les phénomènes dangereux (analyses *backward*).

Parmi les points essentiels de cette activité, il convient de noter l'importance de l'erreur humaine en tant que phénomène dangereux, ayant des causes, des effets, et des interactions avec d'autres défaillances, et qu'il est possible de détecter et de réduire. Il est aujourd'hui devenu évident que ce thème est un des enjeux majeurs de la gestion du risque. Des exemples comme l'avionique où l'erreur humaine est responsable de 70% des accidents, ou comme

l'accident de Chernobyl, démontrent la nécessité d'intégrer cette problématique. Au sein de ce domaine d'étude un aspect important concerne les problèmes d'utilisation des interfaces homme-machine. Un des cas classiques illustrant cet aspect dans le domaine médical, est celui du Therac-25, analysé par Leveson (1995, Annexe A). De 1985 à 1987, cette machine de traitement de tumeurs par laser, contrôlée par ordinateur, a provoqué des dommages sur 6 personnes allant parfois jusqu'au décès. Certains des accidents ont été provoqués par des séquences non prévues de manipulation par les utilisateurs. Ces derniers n'ont forcé ou désactivé aucune fonctionnalité, et les modifications pour bloquer les erreurs ont été introduites par la suite. L'étude de l'erreur humaine est un domaine à la frontière des sciences cognitives (Reason, 1990; Leplat, 1985), et difficilement intégrée par les concepteurs. Comme exposé en 1.4.3.3, la complexité de cette discipline a souvent mené à l'utilisation de *checklist* et de règles de conception. Cependant comme le soulignent Wright et al. (1994) ces moyens ne sont pas suffisants pour de nombreux systèmes et plus particulièrement pour les systèmes innovants. Face à cette problématique, les ingénieurs ont donc développé des heuristiques et des techniques comme THERP (Technique for Human Error Rate Prediction, développée par Swain au Sandia National Laboratories) permettant d'aider les concepteurs à adapter la tâche à l'humain.

1.4.3.4 Estimation du risque

Cette étape correspond au calcul du risque. La norme ISO 14971 (2000) indique que le risque pour chaque *phénomène dangereux* doit être calculé. Or par définition (cf. partie 1.2.2), le risque est calculé sur la base du *dommage*, et non par les phénomènes dangereux qui sont à son origine. L'estimation ne peut donc se faire directement, et le calcul est effectué en combinant les estimations pour chaque phénomène et chaque situation. C'est sans doute une des raisons qui explique pourquoi les normes précédentes proposaient une définition du risque en fonction du danger et non du dommage comme aujourd'hui.

L'estimation consiste donc à identifier la gravité des dommages et la probabilité des phénomènes dangereux conduisant à ce dommage. Le risque global est ensuite calculé à partir de toutes ces données. Pour cette estimation, il est possible d'utiliser des normes publiées sur l'application médicale considérée, des données techniques scientifiques, des données de terrain à partir de robots médicaux déjà en service (données encore très rares), d'essais et d'évidences cliniques, l'opinion des experts, ou des systèmes extérieurs d'évaluation de la qualité.

La principale difficulté provient du calcul de la probabilité d'un dommage. Toute étude débute par une estimation qualitative et lorsque cela est possible, elle est complétée par une étude quantitative. Dans les systèmes complexes comme les robots médicaux (mélangeant des aspects mécaniques, électroniques, informatiques et humains), il est naturel de se poser la question de la faisabilité et de la valeur d'une estimation quantitative. Dans d'autres domaines, tel que l'avionique, certains résultats ont été critiqués soit pour la complexité de leurs calculs,

soit du fait de résultats peu exploitables (des probabilités inférieures à 10^-7) ou encore du manque de confiance que l'on attachait à ces résultats.

La majeure partie de l'analyse du risque couvre l'aspect défaillance des composants du système. Ces défaillances, en tant que phénomènes dangereux, peuvent mener à des situations dangereuses dont on étudie la probabilité et la gravité du dommage induit. Un aspect fondamental dans cette étape est la différence entre les analyses couvrant le système (matériels et logiciels étant pris comme des composants) et les analyses propres au logiciel. Il est en effet impossible d'obtenir des données quantitatives ou qualitatives sur des probabilités de défaillance des composants logiciels qui soient utilisables pour un autre projet. Cet aspect qui n'est pas abordé au niveau des normes médicales, est pourtant déjà largement abordé dans des domaines comme l'aéronautique. La méthode conseillée pour pallier cette méconnaissance, est de donner au sous-système logiciel un niveau d'intégrité (ou SIL, Software Integrity Level) correspondant à un niveau de criticité des fonctions que le sous-système doit réaliser. Ainsi, d'après la norme logicielle aéronautique DO178B/ED-12 Revision B (1992), un logiciel dont un comportement anormal contribuerait à la défaillance d'une fonction du système, et qui conduirait à une défaillance catastrophique pour l'avion, est de niveau *A* le plus élevé (pour un effet nul, le niveau *E* est attribué). Il ne s'agit pas de l'affectation d'un niveau de risque, car ce niveau ne peut être réduit, mais d'un niveau qui impliquera que le fabricant applique des méthodes de développement plus ou moins contraignantes. Ainsi, plutôt que d'analyser le logiciel lui même en tant que produit, cette norme propose de contraindre la production de ce logiciel. Cette philosophie, propre au logiciel, se retrouve dans de nombreux domaines, et notamment dans une norme générique sur les dispositifs électriques programmables, la IEC 61508 (2001). Ainsi, pour le logiciel, les étapes suivantes ne peuvent s'appliquer directement, sauf si l'on considère le logiciel comme un composant au même titre qu'un composant électronique.

1.4.4 Évaluation du risque

L'activité d'évaluation du risque concerne les prises de décisions d'acceptabilité du risque évoquée précédemment (cf. 1.2.2.3). Une première étape de l'évaluation consiste à isoler les dangers dont le risque estimé n'est pas suffisamment faible. Le tableau de la figure 1.1 permet de choisir dans un premier temps les risques que l'on accepte en raison de leur faible niveau. Par exemple, il est possible de décider que seuls les dangers induisant un risque de niveau T ou L ne nécessiteront pas de réduction. En l'état actuel des choses, la détermination d'un niveau de risque acceptable est réalisée en utilisant les niveaux de risque acceptable de dispositifs médicaux déjà en service et ceux d'actes médicaux réalisés sans l'intervention d'un robot.

La deuxième étape consiste en l'étude des risques de niveau trop élevé, mais dont la réduction n'est pas réalisable. Dans cette situation, il convient de réduire le risque au niveau aussi faible que raisonnablement praticable. Ce concept, intitulé ALARP (*As Low As Reasonably Practicable*), exprime la nécessité de réduire au maximum le risque dans la limite des

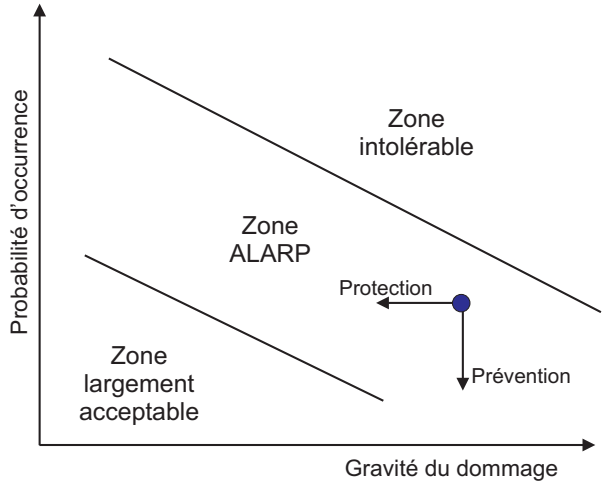

FIG. 1.4 – Exemple de diagramme des niveaux d'acceptabilité du risque et de la zone ALARP

moyens techniques et économiques disponibles (cf. figure 1.4). La réduction du risque peut être réalisée en utilisant deux techniques illustrées sur cette figure. Tout d'abord, la probabilité d'occurrence du dommage peut être réduite, ce qui est noté comme la *prévention*. Puis, la gravité du dommage peut être réduite grâce à des techniques de *protection*.

Une fois le risque ramené en zone ALARP, la problématique risque/bénéfice se pose. La figure 1.5 illustre ce principe pour la zone dite *tolérable* qui n'est qu'une vision différente de la zone ALARP de la figure 1.4.

Cette balance entre risques et bénéfices, est sans doute la notion la plus controversée du processus de gestion des risques, puisqu'elle se base sur des concepts liés aux valeurs de la société concernée. En effet, comme cela a été présenté en partie 1.2.2.3, le risque est accepté dans un contexte donné, basé sur des valeurs courantes de notre société. Ce qui n'est pas explicite dans ces normes, est qu'un risque est acceptable d'une part parce qu'il se situe dans une limite comparable avec d'autres systèmes, mais surtout parce que la société est prête à l'accepter au regard des bénéfices qu'elle peut en tirer. Ainsi, même après avoir déterminé qu'un risque n'était pas acceptable, et qu'il est impossible de le réduire pour des raisons techniques ou économiques, il est toujours possible de décider que l'on prendra le risque en raison du contexte. Le niveau du risque acceptable est alors modifié (et non le niveau du risque lui-même). Pour exprimer le fait qu'un risque non acceptable au départ, puisse être pris par la suite, la notion de risque tolérable est parfois utilisée. Dans la figure 1.5, on retrouve cette notion de tolérance lorsqu'une décision doit être prise. Ce principe est plus largement abordé dans un rapport de la HSE (2001a).

L'exemple de l'hépatite B illustre parfaitement la problématique de la *balance* entre risques et bénéfices. En effet, suite à une campagne de vaccination contre l'hépatite B en France (vaccin ayant fait l'objet d'un processus de gestion des risques), plusieurs patients se sont retournés contre des fabricants de ce vaccin pour des cas d'effets secondaires (et en 2002, contre l'État français). L'ensemble des patients affectés par ses effets a donc dénoncé le niveau de risque

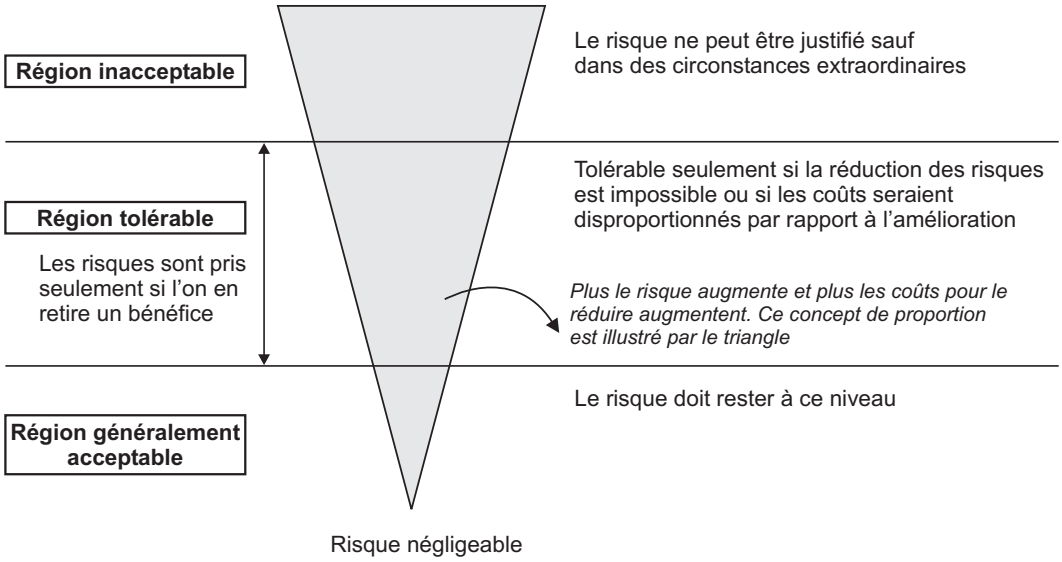

FIG. 1.5 – Illustration de l'acceptabilité du risque en fonction du coût, figure tirée de HSE (1999)

du vaccin en le jugeant inacceptable. L'État, représenté par l'Agence Française de Sécurité Sanitaire des Produits de Santé (l'AFSSAPS), a alors décidé en septembre 1998, « d'interrompre la campagne de vaccination collective en milieu scolaire et de compléter l'évaluation du risque en initiant des études et de réévaluer régulièrement les données disponibles » (Communiqué de presse du 6 mars 2000, Ministère de l'emploi et de la solidarité, Secrétariat d'état à la Santé et à l'Action Sociale[6]).

Dans le domaine des dispositifs médicaux - dont font partie les robots médicaux - la problématique intègre d'autres éléments. Ces robots qui assistent aujourd'hui les spécialistes réalisent des tâches qui existaient auparavant. Ainsi l'acceptabilité du risque peut se mesurer en comparant l'acte médical utilisant le robot, avec l'acte sans le robot. Une étude sur plusieurs années présentée par Bargar et al. (1998), compare avec des métriques du domaine médical les résultats des opérations du remplacement de la hanche par Robodoc avec ceux d'une équipe donnée. Les performances du robot sont ainsi évaluées et les risques encourus sont validés par la mesure des dommages sur les patients (en l'occurrence identiques). De plus, les ingénieurs qui ont développé le premier système de Robodoc, ont choisi de prendre le risque d'adapter un robot industriel pour l'application médicale. En effet, la rigidité, la vitesse et l'espace de travail du robot industriel adapté, constituaient des sources potentielles de dommage, et induisaient ainsi un risque. Une autre comparaison entre un robot chirurgical et une équipe médicale classique est présentée par Reichenspurner et al. (2000). De la même manière, les performances sont évaluées et les dommages sont exposés en terme de temps de récupération, de complications post-opératoires, et de mortalité (nulle dans les deux cas sur plusieurs centaines d'opérations). Pour les futures applications robotiques qui permettront d'effectuer

[6]Document disponible sur http ://www.sante.gouv.fr/htm/pointsur/vaccins/31_000306.htm

des tâches jusqu'alors impossibles, la problématique de l'acceptabilité sera plus complexe. En effet, l'absence de comparaison avec des opérations uniquement réalisées par l'humain, empêche de fixer les règles de risque acceptable. Le processus sera alors le même que pour des nouvelles opérations chirurgicales ; c'est le corps médical et la société qui prendront la décision de faire les essais cliniques et d'accepter un risque non nul.

1.4.5 Maîtrise du risque

La maîtrise du risque concerne les étapes au cours desquelles sont sélectionnés les moyens pour réduire les risques. Du point de vue de la sécurité, il est courant de ne parler que de réduction du risque. Selon la définition du risque (cf. 1.2.2), il est possible soit de réduire la gravité du dommage, on parle alors de mécanismes de *protection*, soit de réduire la probabilité d'occurrence du dommage, ce qui correspond aux actions de *prévention*. Ce concept est illustré sur la figure 1.4.

Cette étape interagit avec le processus de développement et les moyens choisis ne doivent pas dégrader les spécifications du système. Un exemple classique est le temps d'exécution d'un logiciel qui peut considérablement augmenter par l'introduction de procédures dédiées à la sécurité et donc conduire au débordement des échéances temporelles critiques pour un système temps réel. Un des enjeux de la maîtrise du risque est donc de garantir un respect des fonctions du système tout en proposant des mesures pour la réduction du risque.

La maîtrise du risque concerne principalement les activités suivantes :

- Modification des interfaces : les interfaces humains-machines sont modifiées afin de maîtriser le risque. Elles concernent les interfaces humain-ordinateurs, les boîtiers de commande des robots mais aussi certaines parties du robot lui-même qui interagissent avec le patient ou les spécialistes. Il existe de nombreux travaux dédiés exclusivement à ce domaine qui relève de différentes disciplines, et l'attitude générale des ingénieurs de développement consiste à suivre des guides sous forme de règles de conception. La partie 1.6.2.4 aborde cette problématique en se restreignant aux robots médicaux.
- Modification de l'utilisation : l'allocation des tâches est modifiée ainsi que les procédures d'utilisation, les formations ou la rédaction des différents guides accompagnant le dispositif.
- Modification de la conception : certaines solutions pour réduire le risque nécessitent de modifier la conception même du dispositif. On peut d'ores et déjà citer les techniques de tolérance aux fautes fournissant des mécanismes de protection, ou les guides de développement basés sur le principe de prévention des risques de défaillance induites par des fautes de conception (Geffroy et Motet, 2002). Une synthèse de l'ensemble de ces moyens techniques pour la robotique médicale est présentée dans la partie 1.6.

Dans les sections suivantes sont présentées des techniques pour mettre en œuvre l'analyse et la maîtrise du risque. Nous ne présenterons pas de techniques d'évaluation du risque car il est très difficile de trouver dans la littérature et plus particulièrement dans le domaine médical

des travaux sur ce sujet. L'ensemble des concepts d'évaluation du risque sont fortement liés aux valeurs de la société, et en font un domaine où les techniques n'ont pas encore leur place.

1.5 Techniques utilisées pour l'analyse du risque

Les moyens présentés en section 1.4, fournissent une approche globale pour appréhender la gestion du risque. La mise en œuvre de ces moyens peut se faire par l'utilisation de techniques spécifiques à chacune des activités de la gestion du risque. Ainsi, pour l'analyse du risque, il est souvent fait mention de méthodes analytiques qui permettent au concepteur de comprendre comment les phénomènes dangereux, les situations dangereuses et les événements dommageables peuvent mener à des accidents, et d'estimer les probabilités afin d'annuler ou de réduire les effets.

Comme nous l'avons vu précédemment, l'analyse du risque concerne trois points : définition du domaine d'application, identification du phénomène dangereux (et/ou de la situation dangereuse) et estimation du risque. Ces trois volets correspondent en général à trois activités que l'on exécute de manière itérative. Il n'existe pas de technique couvrant globalement la totalité de ces aspects et, la plupart du temps, plusieurs techniques sont utilisées pour chacun d'eux. Le but de la multiplication de ces techniques est aussi d'atteindre un certain niveau d'exhaustivité d'analyse (qui tout comme la sécurité est relatif !).

Les techniques les plus utilisées en robotique industrielle sont présentées par Dhillon et Anude (1993). Parmi ces techniques, Dhillon et Fashandi (1997) retiennent que l'AMDEC[7] (Analyse des Modes de Défaillance et de leurs Effets et de leur Criticité) et l'analyse des arbres de fautes[8] semblent être les plus appropriées pour les systèmes robotiques. Bien que les exemples d'utilisation de ces méthodes (Khodabandehloo, 1996; Walker et Cavallero, 1996) soient orientés vers des robots industriels, elles sont largement applicables aux robots médicaux (Guiochet et al., 2001; Guiochet et Vilchis, 2002; Urbain et Guiochet, 2001). Ces deux techniques souvent utilisées conjointement, sont aussi préconisées pour l'analyse du risque dans le domaine médical (ISO 14971, 2000). Ces méthodes sont particulièrement adaptées aux dispositifs médicaux car il est possible d'inclure certains aspects liés aux facteurs humains comme l'erreur humaine, et aussi de mélanger les différents domaines que sont la mécanique, l'électronique, et l'informatique, présents dans un système robotique.

La figure 1.7 illustre l'utilisation d'un tableau pour l'AMDEC en ne prenant en compte qu'un des modes de défaillance pour le processeur du contrôleur de robot télé-échographe TER (Vilchis et al., 2001a). Il existe en effet d'autres modes de défaillance de ce composant comme la coupure (toutes les sorties nulles) qui peut être due à une chute de tension, ou encore le cas où les sorties deviennent aléatoires du fait d'un dépassement de mémoire ou d'un pointeur *fou* du programme. Le mode de défaillance analysé sur cet exemple est un

[7] Ou FMECA, *Failure Mode, Effects and Critically Analysis*
[8] Ou FTA, *Fault Tree Analysis*

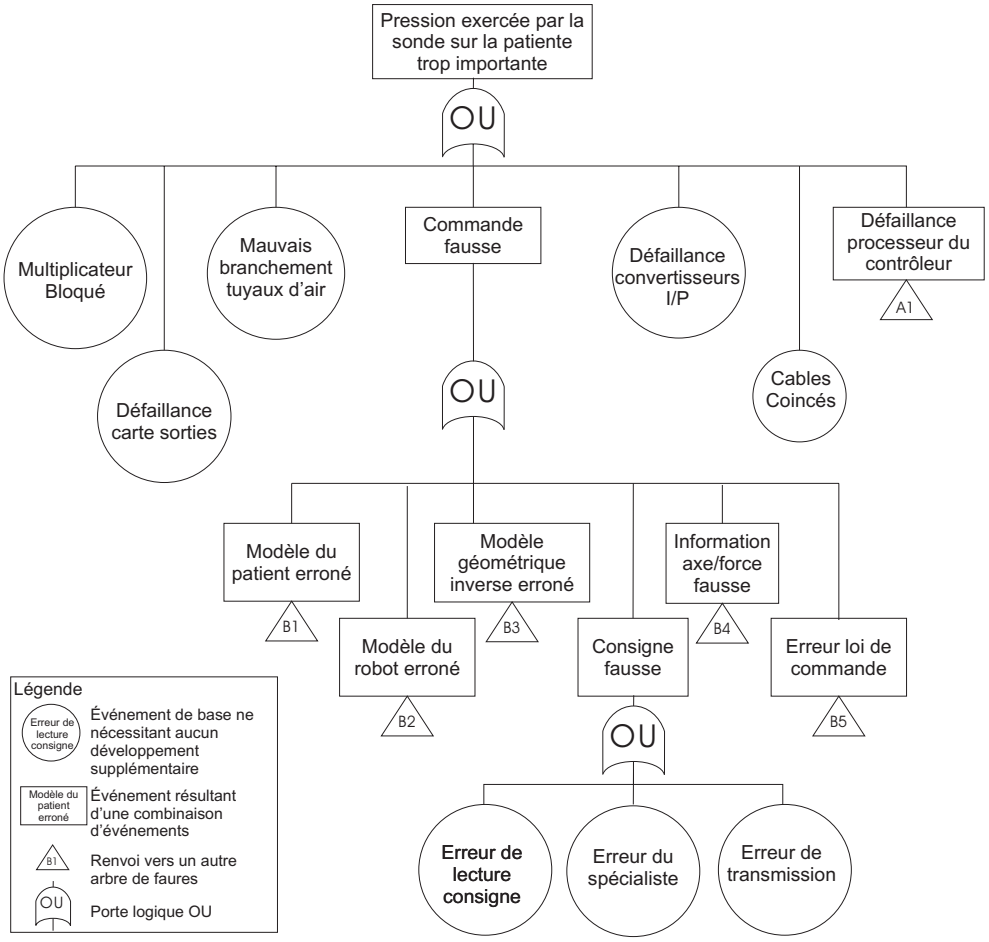

FIG. 1.6 – Exemple d'arbre de fautes

Composant	Modes de défaillance	Cause	A. Effet local B. Effet sur le système	Evaluation du risque			A. Moyens de détection possibles B. Solutions
				Occurrence	Sévérité	Risque	
Processeur du contrôleur de robot	Figé	Interblocage du programme ou du système d'exploitation	A. Envoie commande constante B. Mouvement bloqué en un point	F	1	H	A. Système externe type watchdog B. Réinitialisation et Alerte utilisateur

FIG. 1.7 – Exemple de tableau de l'AMDEC

processeur bloqué dans un état quelconque et ne donnant plus signe d'activité ; on utilise alors le terme de *figé*. La cause principale identifiée est l'interblocage entre tâches, ou une mauvaise gestion des ressources du système d'exploitation. On exprime également les effets, ainsi qu'une estimation du risque suivant les niveaux présentés dans le tableau de la figure 1.1. La solution envisagée pour maîtriser ce risque est l'utilisation d'un système de protection du type *watchdog* ou chien de garde présenté ultérieurement.

La figure 1.6 représente un arbre de fautes pour l'événement racine « *Pression exercée trop importante* » dans le cadre d'un robot pour l'échographie. Certaines particularités du système, comme l'utilisation de nouveaux actionneurs, les muscles artificiels (Tondu et Lopez, 2000) fonctionnant avec de l'air comprimé, introduisent des événements dommageables liés à l'utilisation d'air (mauvais branchements des tuyaux d'air, défaillance des convertisseurs Intensité / Pression, etc.).

Pour chaque technique d'analyse du risque, les solutions proposées pour réduire les risques, présentées par exemple dans la dernière colonne du tableau de l'AMDEC, seront par la suite sélectionnées s'il s'avère nécessaire et réalisable de les implanter. Ceci correspond à l'étape d'évaluation du risque. Parmi ces solutions se trouvent un grand nombre de moyens techniques, dont nous proposons une synthèse dans la section suivante.

1.6 Techniques utilisées pour la maîtrise du risque

Les nouvelles applications de la robotique ont souvent été réalisées à partir de robots industriels, notamment pour les robots médicaux comme le robot échographe Hippocrate (Degoulange et al., 1998) ou le robot chirurgical Robodoc (Cain et al., 1993). Dans ce contexte les solutions techniques permettant d'augmenter la sécurité des robots ont été conservées voire améliorées. Les spécificités des tâches de la robotique médicale ont cependant amené les concepteurs à des solutions propres à chaque application (robotique de réhabilitation, robotique chirurgicale, etc.). Sur la base de ce constat, Davies (1993) présente des solutions technologiques et des principes sur lesquels peuvent se baser les concepteurs de tels systèmes. De la même manière, cette partie présente les principales solutions matérielles et logicielles qui peuvent être utilisées lors de l'étape de maîtrise du risque au sein du processus de gestion du risque. Ces solutions sont issues des exigences non fonctionnelles du système et de l'analyse du risque. Elles découlent donc d'un processus d'analyse et ne peuvent être directement implantées comme garanties de sécurité. Par exemple, afin de réduire le risque de choc entre un robot et un humain, Ikuta et Ishii (2003) introduisent un indice de danger calculé par le rapport entre la force possible produite par le robot et la force maximale que tolère un humain. En se basant sur l'identification et la quantification de ce danger, ils propose alors des solutions de conception et de contrôle permettant de réduire ce danger équivalentes à celle présentée ici. Les solutions technologiques présentées dans cette section sont donc des patrons (ou *patterns* en anglais) de sécurité, puisqu'ils représentent des solutions connues à des problèmes

particuliers dans un contexte[9]. Nous proposons de les utiliser une fois que les risques ont été estimés et évalués. Les autres moyens pour la maîtrise du risque comme la maintenance, les modifications de conception, les procédures d'utilisation, la formation ou la documentation, ne sont pas évoqués car il est difficile d'en effectuer une synthèse et encore plus un catalogue.

1.6.1 Conception matérielle

1.6.1.1 Architectures mécaniques et matériaux utilisés

L'architecture du robot dépend des exigences fonctionnelles mais aussi de certaines exigences de sécurité spécifiques au milieu où le robot évolue. La notion de *sécurité intrinsèque* est parfois utilisée lorsque la structure même du robot possède la particularité de ne pas présenter de phénomènes dangereux (pas de bords tranchants, la vitesse des mouvements ainsi que l'amplitude sont limités, etc.).

Ainsi, à l'opposé d'HIPPOCRATE, un bras manipulateur pour l'échographie (Degoulange et al., 1998), le projet TER (Vilchis et al., 2001a,b) propose une structure dite parallèle restreignant l'amplitude, la force et la vitesse des mouvements. Cette structure permet notamment en cas de coupure d'alimentation en air, de n'exercer plus aucune pression sur la patiente (la notion d'arrêt d'urgence est exposée plus en détail dans la partie 1.6.1.2). Il est évident que c'est en majorité l'analyse de la tâche (incluant l'analyse du geste médico-chirurgical pour les robots médicaux) qui guide les choix d'architecture, et dans beaucoup de cas c'est une architecture série comme les bras manipulateurs, qui a été choisie. Parmi les raisons de ces choix on peut citer le fait que beaucoup de robots médicaux ont été des robots industriels adaptés et modifiés, et aussi le fait qu'il est plus difficile de commander des structures parallèles. Il est possible d'imaginer que les applications de demain seront libérées de ces contraintes, et que certains robots gagneront à changer d'architecture. Les robots médicaux sont construits aujourd'hui pour intervenir sur de très petits espaces de travail, et des choix spécifiques au niveau de l'architecture contribuent à respecter cette contrainte. Ainsi, pour une tâche très spécialisée, il est conseillé de réduire le nombre d'articulations. Le robot neuro-chirurgical Minerva (Glauser et al., 1993) possède seulement trois articulations (une translation et deux rotations). A l'opposé un robot multitâche, travaillant dans un milieu hostile, possédera un plus grand nombre de degrés de liberté, éventuellement redondants. En complément de cette restriction, certains robots sont équipés de butées mécaniques très contraignantes (la rotation de Minerva est limitée à 30 degrés). De plus, certains robots sont entourés de matériaux viscoélastiques comme celui de Morita et Sugano (1997b). La gravité du choc est alors diminuée par l'absorption de l'énergie cinétique ; le système enregistre le niveau du choc et adapte alors sa réaction.

[9]Cette définition de patron n'est pas universelle, et le lecteur pourra se reporter au rapport de Manhes (1998) pour une synthèse sur ce sujet.

1.6.1.2 Sécurité autour des actionneurs

Précision des mouvements articulaires

Il est important de noter que la précision du mouvement articulaire est un critère de performance établi dans les exigences fonctionnelles, et qu'il n'est pris en compte dans une analyse de la sécurité que si la dégradation de cette performance consititue un phénomène dangereux pour l'utilisateur. Dans une analyse des risques, c'est principalement cette défaillance qui est traitée. Il existe aujourd'hui un ensemble d'actionneurs (principalement des moteurs électriques AC, DC, et pas à pas), qui ont des modes et des taux de défaillances connus et généralement fournis par les fabricants. La solution la plus commune à ces défaillances est un programme de maintenance préventive. Cette dernière est définie comme l'ensemble des actions périodiques qui interviennent avant l'occurrence des défaillances. Elle correspond aux procédures de détections, inspections, entretiens, calibrages, tests et ajustages (Worthington et Burns, 1988). En robotique médicale, le choix des concepteurs s'est principalement porté sur les moteurs pas à pas, plus précis que les autres moteurs. En dehors des modes de défaillances des actionneurs, un des phénomènes qui dégrade la précision au niveau de l'articulation est le *backlash* (Klafter et al., 1989) (contre-mouvement dû au jeu des articulations sur un changement de sens). Ce mode est réduit par le principe de transmission harmonique[10] (roues en forme d'ellipses). Dans le cadre du robot neuro-chirurgical Minerva, l'équipe de recherche a mis au point une solution dérivée de ce principe (Glauser et al., 1993).

Arrêt d'urgence

Les mécanismes d'arrêt d'urgence présents sur les robots industriels sont utilisés sous la même forme pour la robotique médicale. Ils constituent des dispositifs de protection permettant de maîtrise l'énergie du système. Le principe d'interrupteur *homme-mort* ou DMS[11], où les mouvements du robot ne sont autorisés que si l'opérateur maintient un contact, est implanté sur des robots médicaux comme Acrobot (robot chirurgical orthopédique présenté par Davies et al. (1997)) sous forme de *gâchette*. Il peut aussi être sous forme de pédale comme sur le robot échographe Hippocrate (Degoulange et al., 1998). Comme pour les robots industriels, l'appellation *arrêt d'urgence* n'est valide que si le circuit est dit *figé* (INRS, 1998), ce qui veut dire entièrement matériel, sans aucun traitement informatique. Ces systèmes sont généralement doublés par des boutons poussoirs directement reliés aux sources d'énergie.

Une fois l'arrêt d'urgence activé, différents systèmes de blocage des articulations peuvent être intégrés. Wadegaonkar, Sunnapwar, et Modak (1996) utilisent des systèmes de freins débrayables électromagnétiques que l'on trouve sur des robots industriels commercialisés, comme le Mitsubishi MoveMaster (bras manipulateur à 6 axes), couplés à des moteurs électriques. Il existe aussi des systèmes de freins utilisés par Ng et Tan (1996), qui s'enclenchent

[10] ou *harmonic drive*, produit d'une division de Quincy Technologies, Inc., USA.

[11] Dead Man Switch

lors des coupures d'énergie. Le robot Wendy (Morita, Iwata, et Sugano, 1999) possède un système fonctionnant sans logiciel qui permet d'enclencher des freins électromagnétiques si le bras dépasse une vitesse limite. Delnondedieu (1997) présente le mécanisme de plusieurs systèmes, notamment le concept de Transmission Continûment Variable utilisé par Cobot[12], et le système de roue libre utilisé par le robot Padyc (Troccaz et Delnondedieu, 1996). Ce système est mis en œuvre par un ensemble de billes qui se coincent dans des cavités, bloquant alors seulement dans un sens la rotation d'une bague extérieure. Prévu à la base pour une restriction de l'espace de travail du robot (le geste chirurgical est alors contraint), il est possible de l'utiliser pour un blocage en cas d'arrêt d'urgence.

Il ne convient cependant pas toujours d'utiliser des systèmes de blocage des articulations. En effet en cas de panne, l'arrêt peut ne pas être systématique, il pourrait même s'avérer dangereux dans certaines circonstances. Certains sites robotisés utilisent ainsi des systèmes hydrauliques secondaires pour des robots manipulant des charges très lourdes. En cas de défaillance cette installation prend le relais et redescend doucement le bras chargé sur la plateforme robotique. Cette utilisation reste tout de même rare et concerne le milieu industriel. Dans le cas des robots médicaux, c'est une problématique complexe qui dépend entièrement de l'application. Le choix du type d'arrêt d'urgence est guidé par les états dans lesquels on peut placer le système. Le cas le plus classique est celui où toute énergie est retirée lors de l'enclenchement de l'arrêt d'urgence. Pour un robot chirurgical, une des solution est de le figer (par exemple avec des freins électromagnétiques) dans sa position et de permettre au spécialiste de venir le retirer (en ayant un mode débrayé par exemple) et de terminer l'opération sans le robot. C'est l'humain qui doit intervenir pour cette situation extrême, il ne doit donc pas exister de mouvements de recul du robot suite à un traitement informatique. Il existe certaines applications où le robot se place dans un état dit de *pause* suite à une défaillance détectée par le logiciel. Cet état ne peut être comparé à un arrêt d'urgence et il est très important de faire la distinction. Dans le cas du robot télé-échographe TER, deux choix de conception ont permis d'adopter un arrêt d'urgence sans blocage ou frein. La sonde, maintenue par quatre câbles, repose sur le ventre de la patiente, et ces câbles sont actionnés par des muscles artificiels de McKibben (Tondu et Lopez, 2000) qui se rétractent lorsque la pression d'air augmente. Ils permettent ainsi de déplacer l'anneau où se trouve la sonde. Le fait de couper toute alimentation d'air, détend les quatre muscles, et la pression sur le ventre de la patiente devient nulle. Dans ce cas particulier l'arrêt d'urgence est très simple, et place directement le système dans un état sûr. Les corrélations qui existent entre les choix d'architecture, le choix des actionneurs et la sécurité, sont très importantes et encore à l'étude du fait de la difficulté de trouver de nouveaux actionneurs et de nouvelles architectures. C'est un thème qu'il conviendrait d'introduire au niveau de la sécurité.

[12]http ://othello.mech.nwu.edu/ peshkin/cobot/index.html

Compliance passive

Les robots industriels sont en général lourds, puissants et programmés pour effectuer une tâche sans obstacle. Ces caractéristiques conduisent à identifier l'énergie cinétique comme étant un de principaux phénomènes dangereux. Ce phénomène est inacceptable dans le cadre de la robotique de service qui exige fluidité, légèreté et souplesse. Cette dernière caractéristique, proche de l'humain, est un des enjeux les plus importants en robotique de service. Le contrôle d'un mouvement sous de fortes contraintes (comme un contact permanent avec l'homme) a amené les fabricants à développer des contrôleurs de robot combinant un contrôle en position et en force. La *compliance active*[13] est ce type de contrôle par l'utilisation d'algorithmes implantés dans le contrôleur, alors que la stratégie de la *compliance passive*, celle qui nous intéresse ici, est définie par « l'utilisation de structures matérielles montées sur le poignet du robot » (Wu, 1988). Il est cependant possible d'étendre cette définition à l'ensemble des articulations d'un robot. De tels mécanismes apportent une souplesse au poignet ou à d'autres articulations du robot. L'utilisation de poignets compliants permet de se rapprocher de la souplesse de l'homme et marque une première étape en ce sens. Le MIA (Mechanical Impedance Adjuster) présenté dans l'article de Morita et Sugano (1997a), se compose principalement d'un ajusteur d'impédance (afin de régler la souplesse) et d'un entraînement d'articulation (pour transmettre le mouvement). La compliance consiste à régler un ressort directement joint à l'axe de l'articulation.

La compliance passive peut se faire aujourd'hui sans structure matérielle montée sur le poignet. En effet, il est possible d'obtenir une compliance naturelle de l'actionneur. Les muscles artificiels décrits par Tondu et Lopez (2000) offrent une compliance très comparable aux articulations humaines. Ils contribuent ainsi à la sécurité intrinsèque du robot. Ce sont des actionneurs pneumatiques qui se rétractent lorsque la pression augmente. Ils ont été utilisés pour le développement du robot télé-échographe TER (Vilchis et al., 2001a) mais aussi dans le cadre de bras manipulateurs comme le robot ISAC[14] pour la réhabilitation des personnes âgées ou handicapées.

1.6.1.3 Sécurité par la redondance des composants

La redondance structurelle consiste à multiplier les composants matériels du robot comme les actionneurs, les capteurs, les cartes contrôleur, etc. Les données produites par ces composants peuvent ensuite être comparées et certains composants écartés en raison d'une défaillance détectée. Cette technique de protection, utilisée dans les systèmes à sécurité critique comme l'avionique est aujourd'hui employée en robotique (Rodrigues, 1993; Baerveldt, 1992a). Par exemple, le robot chirurgical Robodoc possède pour chaque articulation deux codeurs incrémentaux (données de position), et les données sont comparées à tout instant par

[13] *Compliance* est un anglicisme que l'on trouve parfois traduit par *complaisance*, ou plus simplement par *souplesse*

[14] Intelligent Soft Arm Control, http ://shogun.vuse.vanderbilt.edu/CIS/IRL/Projects/isaac.html

deux processeurs (Cain et al., 1993). La redondance structurelle est à la base une technique pour augmenter la fiabilité : on prolonge la fonction du système même si un des composants est défaillant. Mais dans le cas où la défaillance du composant est critique en terme de sécurité, la redondance devient alors une technique de sécurité.

Il existe un autre type de redondance, la redondance fonctionnelle, qui permet d'effectuer une même fonction mais avec une conception différente. Ainsi, il est possible de retrouver une donnée de vitesse avec des capteurs de vitesse mais aussi depuis des données de position et d'accélération du mouvement. La redondance peut aussi s'effectuer à un niveau beaucoup plus élevé comme l'ensemble d'un contrôleur de robot. Laible et al. (2001) présentent une application d'un patron de sécurité, l'utilisation d'un *double canal*[15], à un robot chirurgical. L'information depuis l'utilisateur transite par deux canaux symétriques mais de conception différente. Le terme de canal exprime le fait que les données passent par un ensemble de composants où elles sont traitées. Ainsi les valeurs de la position du robot sont comparées tout au long du calcul (en utilisant une mémoire partagée entre les deux canaux), avant d'être envoyées aux moteurs. Tout comme la redondance structurelle, cette technique est en général limitée par les coûts, l'espace, la consommation d'énergie et la complexité du système.

1.6.1.4 Sécurité par la détection de contacts avec l'humain

Le principal risque pour la robotique industrielle est la présence de l'humain dans la zone de travail. L'INRS (1998) présente l'ensemble des moyens de détection possible de l'humain dans la zone de travail, à proximité et au contact. Ces technologies (cellules photoélectriques, champs magnétiques, planchers, détecteurs volumétriques, antennes, peaux sensibles , etc.) peuvent être utilisées pour la robotique de service afin de modifier le comportement du robot. Mais pour beaucoup d'applications de robotique de service, la tâche s'effectue en collaboration avec l'humain dans la zone de travail. En dehors des dangers liés au domaine médical (nature biologique, liés à l'environnement, etc.) et de ceux qui ne présentent pas de détection possible, on peut identifier le danger principal qui est un déplacement du robot provoquant un dommage humain. La source est donc d'origine physique et il faut, pour détecter ce danger, des capteurs de force suffisamment précis. Au niveau de la structure même du robot il est possible d'utiliser des matériaux couvrant le robot et permettant de mesurer la force de contact ou la pression exercée en tout point (Yamada et al., 1996). Les capteurs d'effort permettent aussi de détecter une pression trop importante sur le patient et de modifier l'état du robot. Ainsi, en cas de dépassement d'une force maximale mesurée sur l'effecteur, le robot échographe Hippocrate (Degoulange et al., 1998), se débraye et le spécialiste peut déplacer manuellement le bras pour le mettre dans une position non dangereuse. De nombreux systèmes robotiques utilisent aujourd'hui le retour d'effort en tant que variable de rétroaction en associant à l'effecteur ou aux articulations des capteurs de force. Bien que le retour d'effort soit une technique étudiée depuis plusieurs années, elle est encore peu présente en robotique médicale, du fait notam-

[15] *Dual channel pattern* présenté par Douglass (1998)

ment que le calcul de cette commande qualifiée d'*hybride* est complexe et que les données des capteurs de force sont difficiles à interpréter en termes de dommages sur l'homme. Dans le cas d'un robot chirurgical, il est très difficile lors d'une opération de mesurer l'ensemble des efforts sur l'outil chirurgical. Les solutions que nous venons de présenter cherchent à mettre en place des moyens de réduction des risques de dommages induits par la force ou l'énergie cinétique. Rappelons que nous avons également proposé des moyens de réduction de ces dommages basés sur la souplesse (compliance) des actionneurs. Pour ces raisons, la génération de mouvements compliants sans recours à un calcul de force apparaît comme une solution d'avenir pour un geste médical plus sûr. Par ailleurs, notons que l'imagerie médicale, utilisée jusqu'alors à des fins de diagnostic, de localisation et en suivi postopératoire est aujourd'hui entièrement intégrée et utilisée lors de l'opération avec un robot médical (Smith-Guerin, 2000, p20-25).

1.6.2 Conception logicielle

Les contrôleurs de robots ont aujourd'hui atteint la complexité de nombreux systèmes informatisés. Pour les directives européennes exposées dans la section 1.7, le logiciel est considéré comme un des éléments du dispositif médical, et doit être traité de la même manière que les autres composants. Il n'y a pas encore de norme dédiée au logiciel médical qui serait analogue à la DO178B/ED-12 Revision B (1992) pour le logiciel avionique. La seule recommandation que l'on peut extraire des directives est de suivre un système qualité de type ISO 9000-3 (1997), appliqué au logiciel médical (Cosgriff, 1994). Ces recommandations concernent surtout la manière dont le logiciel doit être conçu. Dans cette partie nous ne traiterons pas cet aspect, nous étudierons les solutions technologiques logicielles permettant la réduction du risque. Ainsi nous ne nous intéresserons pas au vaste domaine des techniques de sûreté de fonctionnement appliquées au logiciel. De plus, en se contraignant au domaine de la robotique de service, il existe tout de même de nombreuses solutions sans cesse en évolution. En conséquence, cette section présente des pointeurs vers les éléments de recherche majeurs permettant de réduire certains risques.

1.6.2.1 Limitations logicielles des performances du robot

Le principe de limitation logicielle de performances est utilisé dans la robotique industrielle. Par exemple, lorsqu'un opérateur doit effectuer une tâche de maintenance ou programmer une nouvelle tâche robotique, la vitesse des mouvements peut être limitée à 20 cm/s. Cette valeur que l'on peut retrouver dans plusieurs études et dans certaines normes robotiques industrielles (comme par exemple la norme américaine ANSI/RIA R15.06-1999) ne convient pas à toutes les applications de la robotique de service. Sa valeur a été fixée en fonction du temps de réponse de l'humain face à une défaillance. Pour des applications de robotique médicale cette valeur doit être adaptée en fonction de l'application et elle se situe évidemment

bien en dessous de 20cm/s. Dans le cadre de la téléopération, le robot suit les mouvements du spécialiste et doit pouvoir bloquer les mouvements trop rapides ou en dehors de la zone de travail. Pour ces applications, le robot ne peut aller plus vite que le spécialiste car il suit les mouvements de l'opérateur, mais pour les robots autonomes comme Robodoc, il est possible d'effectuer un geste plus rapide. Il n'existe aujourd'hui encore aucune information ou aucune norme relative à la vitesse de déplacement d'un robot médical. Il semble pour l'instant difficile de définir une vitesse commune à des opérations aussi différentes qu'une chirurgie invasise ou un examen échographique.

1.6.2.2 Sécurité lors de la mise sous tension

La sécurité attachée aux étapes de mise sous tension et d'arrêt des systèmes robotiques est bien maîtrisée pour les robots industriels. Dans le cas des robots médicaux il est important de spécifier l'étape de préparation du patient lors de la mise en route. D'une part, la séquence de mise en route doit intégrer le placement du patient tout en garantissant la sécurité de celui-ci. Et d'autre part la préparation du patient peut avoir des effets sur la sécurité lors de l'opération. La conception du contrôleur doit prendre en compte cette donnée dès les spécifications. Les étapes essentielles des vérifications à la mise en route d'un robot médical concernent les tests de la mémoire et du processeur par le BIOS, les tests du *checksum* du code exécutable et des fichiers utilisés, les tests interactifs des capteurs et des actionneurs, et les tests des circuits d'arrêt (Laible et al., 2001). Dans le cadre du robot TER, la mise sous tension est effectuée en premier avec tous les tests relatifs aux composants électroniques. Une fois ces tests validés par l'utilisateur, la pression d'alimentation en air des actionneurs est commandée. D'une manière générale, si le système peut être divisé en blocs comportant des alimentations séparées (et d'origines différentes comme l'électricité et l'air), la mise sous alimentation doit se faire de manière séquentielle.

1.6.2.3 Surveillance du système

Les mécanismes de surveillance doivent en général (Leveson, 1995) :

- détecter les problèmes de très bas niveau pouvant induire des situations dangereuses,
- être indépendants des dispositifs qu'ils surveillent,
- augmenter le moins possible la complexité du système,
- demander une maintenance simple.

Largement répandue aujourd'hui, l'utilisation de chiens de garde (*watchdog*) traite les modes de défaillance comme celui de l'exemple du processeur figé (cf. section 1.5). Ce système peut être matériel ou logiciel. Il est en effet possible de concevoir une carte électronique qui ré-initialise le système si ce dernier ne donne plus signe d'activité au bout d'un certain temps. De manière analogue, il est possible de trouver le même concept pour le logiciel. En effet, il existe aujourd'hui des composants logiciels disponibles sur les plates-formes temps réel

(comme *VxWorks*[16]), qui permettent de lancer une tâche de plus haut niveau si le programme semble être bloqué ou dépasse le temps imparti. Dans les deux cas, matériel ou logiciel, on évite un mode de défaillance du contrôleur de robot et, en choisissant la solution matérielle, on s'affranchit des problèmes liés au processeur. C'est un patron de sécurité que l'on peut retrouver dans de nombreux systèmes et que Douglass (1999) modélise avec la notation orientée objet UML (Booch et al., 1999).

En plus de cette surveillance d'activité, certains robots sont équipés de modules de surveillance qui comparent et analysent toutes les entrées/sorties et les états du système. Robodoc possède ainsi deux cartes contrôleurs :

- une carte principale pour les aspects fonctionnels,
- une carte dédiée à la sécurité qui compare les différentes données, analyse les messages, etc.

D'une manière similaire, Baerveldt (1992b) expose un système avec une carte principale, des cartes contrôleurs pour chaque articulation, et une carte sécurité divisée elle-même en deux modules : un module de contrôle de la carte principale, et un module de contrôle des cartes contrôleurs des articulations. Les informations sont donc analysées et comparées par différents modules. En cas de détection de défaillance, le processeur doit appliquer une politique de sécurité qui peut être le recouvrement d'information (on effectue un retour en arrière sur les données sauvegardées), ou même la réinitialisation du système. Un patron de sécurité regroupant l'utilisation d'un chien de garde et d'un système de surveillance de l'exécution est présenté par Douglass (1999), sous l'appellation *Safety Executive Pattern*.

D'une réalisation plus simple, les contrôleurs peuvent comparer et analyser les données provenant des capteurs (comme par exemple avec le principe de redondance analytique présenté par Visinsky et al. (1994)), et les filtrer avec différents algorithmes numériques (un des plus célèbres étant le filtre de Kalman). Le système garantit ainsi une cohérence de ses entrées.

1.6.2.4 Conception des interfaces humain-machine

Bien que cette sous-partie soit placée dans la section *Conception logicielle*, la conception des interfaces humain-machine ne peut être réduite à des moyens logiciels. En effet, depuis l'étude de la forme et la couleur d'un bouton d'arrêt d'urgence jusqu'à la conception d'une interface graphique, les études couvrent des systèmes matériels et logiciels. Cependant, dans le cadre de la robotique de service et plus particulièrement de la robotique médicale, de nombreuses études sur les interfaces concernent des systèmes informatisés.

Les études effectuées dans ce domaine ne peuvent se résumer à un paragraphe, et l'on ne pourrait énoncer ici l'ensemble des solutions techniques adoptées par les systèmes robotiques. C'est un sujet qui relève de différentes disciplines, et qui se trouve au croisement de l'analyse des facteurs humains, des sciences cognitives et de la théorie des systèmes ; le terme *génie cognitif* parfois utilisé, permet de regrouper cette multi-compétence. Ainsi, cette sous-section

[16] Produit de *Wind River Systems, Inc.*, http ://www.windriver.com

présente quelques points importants des problématiques et des travaux réalisés autour des interfaces.

Une sous branche importante de ce domaine est l'étude des interactions humain-ordinateur. Dans le cadre de la robotique médicale, les interfaces fournissent au spécialiste des informations sur l'état du système mais surtout les données essentielles relatives à la tâche (par exemple une image pour une opération chirurgicale ou pour une échographie). En cela, la qualité des informations sur l'état du système délivrées à l'opérateur est une garantie de la bonne exécution de la tâche, et ainsi de la sécurité.

Le spécialiste utilisant un robot médical conserve sa responsabilité de médecin mais devient responsable du contrôle de la tâche, comme l'est un opérateur pour un robot industriel. Plusieurs problèmes se posent alors, et notamment l'analyse des erreurs humaines liées aux interfaces, les choix des messages d'alerte, les informations affichées, etc. L'aspect cognitif de ces études étant complexe et éloigné des sciences de l'ingénieur, le choix est souvent fait de suivre des directives sous forme de listes de règles de conception (Leveson, 1995, p.485-488), encore appelées « bonnes pratiques ». Cependant comme le soulignent Wright et al. (1994), ces moyens sont insuffisants pour de nombreux systèmes et plus particulièrement pour les systèmes innovants. Ils ne donnent pas la solution exacte au problème lié à l'application mais une façon d'orienter la solution, et à ce titre ils ne garantissent pas la sécurité du système. Les interfaces suivent donc, comme pour tout composant du système, un cycle de vie itératif avec notamment une étape d'analyse du risque lié aux facteurs humains.

L'aspect propre à la robotique médicale provient de l'effecteur du robot qui entre en contact avec le patient, et du système permettant au spécialiste de donner des consignes au système robotique. Les études relatives à ces deux aspects concernent principalement les fonctionnalités du système. Par exemple un système de vision en trois dimensions comme le robot da Vinci[17] pour le spécialiste et un retour d'effort de l'effecteur sur le patient sont des fonctionnalités avancées, mais ne concernent pas directement la sécurité. Il est évident qu'un système difficile à piloter pour un spécialiste s'avérera dangereux pour le patient, mais nous considérons que cela fait partie des exigences du système que le concepteur doit suivre. Les recherches très techniques dans ce domaine ne sont pas directement reliées à des exigences de sécurité, c'est lors de l'analyse du risque que le concepteur peut décider s'il a besoin ou non de certaines fonctionnalités au regard de leur apport pour la sécurité.

1.7 Assurance : la certification

La certification a pour but de fournir à un client une garantie qu'un produit est conforme à sa spécification. Dans notre cas, les spécifications de sécurité sont au centre du problème.

[17] Intuitive Surgical Inc., USA, http ://www.intuitivesurgical.com

1.7.1 Certification et risque

Bien qu'une analyse du risque ne puisse suffire à un processus de certification, elle constitue néanmoins le cœur des principales exigences actuelles en terme de certification. Afin d'harmoniser et d'assurer un haut niveau de sécurité, trois directives européennes, 90/385/CEE (1990), 93/42/CEE (1993), et 98/79/CE (1998), couvrent l'ensemble des équipements médicaux. Ces directives définissent un dispositif réglementaire pour la certification (marquage CE obligatoire) et expriment les exigences essentielles de sécurité. Il existe à un niveau plus technique, un ensemble de normes dédiées aux dispositifs médicaux qui concernent des aspects plus applicatifs, et la manière de mettre en œuvre les exigences des directives. Il est cependant important de noter que dès ces premières directives de très haut niveau conceptuel, l'accent est mis sur l'analyse du risque. La directive 93/42/CEE (1993) l'exprime ainsi dans la première des exigences essentielles :

> *« Les dispositifs médicaux doivent être conçus et fabriqués de telle manière que leur utilisation ne compromette pas l'état clinique et la sécurité des patients ni la sécurité et la santé des utilisateurs,[...], étant entendu que les risques éventuels liés à leur utilisation constituent des risques acceptables au regard du bienfait apporté au client ».*

Comme pour la définition de la sécurité, il ressort de cette directive une valeur relative de la prise de risque faisant référence à la notion d'acceptabilité. Ainsi, l'importance des étapes d'analyse et d'évaluation du risque apparaît clairement pour tout dispositif médical. De plus, lors d'un processus de certification, un organisme notifié procédant à l'examen de conception (le G-MED en France) doit pouvoir accéder aux données et aux résultats de l'analyse du risque.

1.7.2 Classification des dispositifs médicaux

Le processus de certification fait appel à plusieurs étapes entièrement dépendantes de la classe à laquelle appartient le dispositif. La classification se fait en fonction du type d'interaction avec le patient. On peut résumer ces classes à (93/42/CEE, 1993) :

- classe I : dispositifs non invasifs,
- classe IIa : dispositifs actifs échangeant de l'énergie,
- classe IIb : dispositifs actifs échangeant de l'énergie potentiellement dangereuse,
- classe III : dispositifs implantables ou dispositifs invasifs à long terme.

La plupart des robots médicaux appartiennent à la catégorie IIa et IIb. La classe III concerne les dispositifs en rapport avec le cœur ou le système nerveux, qui pour la plupart sont implantés dans le corps humain (comme les *pacemaker*). Les robots chirurgicaux comme Robodoc appartiennent à la classe IIb ainsi que les robots non invasifs mais utilisant une énergie ionisante (un robot échographe par exemple). Cette classification exprime un niveau d'interaction avec le patient sans préciser explicitement le niveau de risque correspondant. La notion de

défaillance n'est pas abordée dans cette classification. Il est intéressant de noter que dans un domaine comme l'avionique, la norme logicielle DO178B/ED-12 Revision B (1992) fait référence à l'*effet de la défaillance* d'un élément pour la classification (catastrophique, dangereux, majeur, mineur, aucun effet). Cette référence fait défaut dans les systèmes médicaux où tout est centré sur l'humain et la proximité du dispositif, sans attribuer directement un niveau de risque à chaque classe. En effet, une défaillance d'un dispositif de la classe III peut avoir un effet moins grave qu'une défaillance d'un dispositif de la classe IIb. Les notions de défaillance et d'effet ne sont abordées que dans les normes sur la gestion du risque. La différence d'approche entre l'avionique et le médical tient au fait que ce dernier domaine aborde depuis peu la notion de système, et de ce fait ne possède encore que des normes de haut niveau. En devenant plus techniques, les normes du domaine médical devraient permettre d'exprimer les raisons d'une telle classification. En avionique, où ces notions sont abordées depuis longtemps, les normes se concentrent sur les composants du système, et suivent une approche beaucoup plus technique. Ainsi, le logiciel, comme composant, a une norme dédiée tout comme la technologie électronique. On peut noter que dans le médical, bien que la complexité des logiciels augmente, il n'y a toujours pas de norme dédiée au logiciel.

1.7.3 Processus de certification

Dans le cadre d'une étude de la sécurité, et en vue de la certification, il est obligatoire de classer le dispositif comme présenté dans la section précédente. En fonction de la classe du dispositif, la procédure de preuve de conformité devient plus contraignante. En effet, pour un dispositif de classe I, seule une auto déclaration du fabricant est nécessaire. Pour des dispositifs comme les robots médicaux (classe IIa et IIb), il est exigé un examen complet de conception. Cela consiste à fournir à l'organisme de certification, une trace (un dossier) contenant l'ensemble des informations sur le processus qualité qui a été appliqué (en général de type ISO 9000), les informations sur la gestion des risques, et les informations sur la conception même du dispositif. Cette procédure, qui met l'accent sur le processus de fabrication plutôt que sur le produit lui même, est commune à de nombreux domaines. Ce qui apparaît néanmoins dans le domaine médical c'est une absence de normes ou de guides relatifs à la production du dossier de certification. Dans le domaine des transports, on retrouve cette notion dans la rédaction des *safety cases*, qui ne trouvent pas leur équivalent dans le médical. Il existe tout de même des normes pour guider le fabricant dans l'application du processus qualité, comme par exemple la norme NF EN 46003 (1999) qui exprime les particularité de l'utilisation de l'ISO 9003 à la fabrication d'un dispositif médical. On retrouve alors des éléments comme le *dossier de gestion des risques* et d'autres dossiers relatifs au processus. Les relations entre l'utilisation de ces normes et la rédaction du document pour la certification, restent néanmoins floues et peu explicites.

1.8 Conclusion et problématique

Les technologies ont aujourd'hui atteint un tel niveau de développement qu'il est désormais possible de transférer au robot un ensemble de responsabilités et de fonctions jusqu'alors réservées exclusivement aux spécialistes humains. Le domaine de la robotique de service et plus particulièrement de la robotique médicale est un exemple de cette évolution. Dans ce cadre, se pose la problématique de la confiance accordée à ces systèmes technologiques, et notamment celui de la sécurité.

Afin de décomposer la notion de sécurité, ce chapitre a repris les notions de base comme le dommage, le risque, et le danger et a cherché à faire le lien entre les normes internationales et la recherche en robotique médicale. Nous avons montré comment il était possible aujourd'hui d'arriver à un ensemble de termes stables, non ambigus et également applicables à d'autres domaines technologiques.

Certaines relations non explicites dans les normes induisent encore quelques imprécisions notamment pour ce qui est des moyens d'appréhender la sécurité. Nous avons choisi de décomposer la gestion du risque en trois étapes, l'analyse du risque, l'évaluation du risque et la maîtrise du risque. Au sein du processus de gestion du risque subsistent deux points essentiels en cours de développement. Tout d'abord l'inclusion des facteurs humains dans les différentes activités n'est pas explicite, et nous avons montré comment il était possible de prendre en compte ce concept à plusieurs niveaux du processus de gestion des risques. Le second point concerne l'estimation du risque pour le logiciel. Il est en effet impossible d'estimer la probabilité d'occurrence de défaillance d'un logiciel, et cela est en partie traité par l'introduction des SIL (*Software Integrity Level*) au sein de différentes normes. Cette démarche est actuellement absente des normes dédiées aux dispositifs médicaux ou aux robots de service.

Afin de mettre en application les activités de la gestion du risque, des techniques sont utilisées mais ne couvrent que certains aspects du processus. Ainsi, l'analyse du risque est en partie traitée par l'utilisation de techniques comme l'AMDEC ou les arbres de fautes. Ces techniques largement utilisées dans d'autres domaines sont aujourd'hui applicables à des dispositifs comme les robots de service. Les principales techniques pour la réduction du risque, ont ensuite été présentées, en se concentrant sur deux aspects : les moyens matériels et logiciels. La séparation entre matériel et logiciel a permis de structurer cette partie, mais il est évident qu'un grand nombre de patrons de sécurité nécessitent l'utilisation combinée de ces deux technologies. Beaucoup de techniques utilisées dans la robotique industrielle, mais aussi dans l'avionique ou le nucléaire, sont utilisables pour la robotique médicale. Les particularités des applications médicales nécessitent tout de même des investigations qui n'existaient pas jusqu'ici, en particulier sur de nouvelles architectures, de nouveaux actionneurs, et sur l'intégration de la mesure des efforts appliqués sur le patient. Ces thèmes de recherche ont une incidence directe sur la sécurité, qui est la contrainte principale pour le développement de nouvelles applications de robots médicaux. Dans ce cadre de recherche, le chapitre 2 présente

une analyse d'un actionneur atypique, le muscle artificiel, et évalue les risques et les bénéfices induits par l'utilisation de cet actionneur.

Afin de valider qu'une conception a bien intégré les différents concepts relatif à la sécurité, la société propose un système d'assurance par le biais de la certification. Dans le cas des dispositifs médicaux, c'est un ensemble de directives qui définit le cadre légal de la certification. La section 1.7 a montré comment cette réglementation intègre les aspects liés au risque, présentés dans les sections précédentes. Comme pour des démarches qualité de type ISO9000, nous avons montré comment la certification des dispositifs médicaux dont font partie les robots médicaux, se basent plus sur la manière dont le système est conçu que sur l'évaluation du système lui-même. L'observation des informations de la conception ne peut se faire que si la documentation a été correctement réalisée, et permet notamment la *traçabilité* des choix de conception. La traçabilité définie d'une manière générique dans la norme ISO 8402 comme « *l'aptitude à retrouver l'historique, l'utilisation ou la localisation d'un produit ou d'un processus de délivrance d'un service au moyen d'identifications enregistrées* », peut s'appliquer à un développement d'un dispositif technologique comme un robot de service.

À partir de ces différentes conclusions, le travail présenté dans cette thèse contribue à apporter une réponse à certains des besoins évoqués ci-dessus. Ainsi, la démarche proposée tentera de fournir une solution aux différentes exigences présentées ci-dessous :

- utiliser une terminologie stable et non ambiguë pour la sécurité (ce qui est fait dans ce chapitre) ;
- permettre une approche système multi-domaine ;
- intégrer les facteurs humains relatifs à la sécurité, c'est à dire proposer une démarche « centrée humain » ;
- intégrer l'analyse du risque du logiciel ;
- et favoriser la traçabilité des choix de conception pour répondre aux exigences de la certification.

Outre ces différentes problématiques, un effort doit être fait pour utiliser des techniques connues des ingénieurs, ou facilement compréhensibles. Le chapitre 4 présente cette démarche, et valide son utilisation par un cas concret dans le chapitre 5.

Chapitre 2

Risques liés à l'utilisation des muscles artificiels de McKibben

2.1 Introduction

Les robots industriels effectuent aujourd'hui des tâches complexes, rapides et extrêmement précises. Ces robots ont inondé les industries de l'électronique et de l'automobile, et l'on assiste à l'explosion du nombre de ces dispositifs au sein des pays industrialisés. À titre d'exemple, le Japon comptait en 1999 plus de 400 000 robots industriels, soit un robot pour quatre ouvriers (Karlsson, 1999). Les actionneurs utilisés pour ces robots sont étudiés depuis de nombreuses années et satisfont pleinement les exigences du domaine industriel. Cependant, comme nous l'avons montré au premier chapitre, l'avènement d'une robotique de service à côté de la robotique industrielle, soulève des problèmes qui n'existaient pas ou avaient été évités jusqu'ici. En particulier, le fait de travailler au contact d'un environnement non maîtrisé, comme un environnement humain, ne permet plus d'effectuer des tâches à la manière des tâches industrielles. L'exigence de la sécurité rend ce problème encore plus complexe. Une approche utilisée aujourd'hui, consiste à concevoir des systèmes comportant de nombreux capteurs extéroceptifs, et notamment des capteurs de force, et de contrôler les mouvements du robot par des commandes hybrides, comme par exemple une commande position/force. Beaucoup d'applications de la robotique de service utilisent des actionneurs classiques de la robotique industrielle tels que (Hannaford et Winters, 1990) :

- les moteurs à courant continu,
- les moteurs à induction à courant alternatif,
- les moteurs pas à pas,
- les actionneurs hydrauliques cylindriques,
- et les actionneurs pneumatiques cylindriques.

La plupart des actionneurs utilisés dans des domaines de pointe comme la robotique médicale sont des moteurs pas à pas en raison de leur précision et de leur faible couple. Cependant, les contrôleurs permettant de réaliser des tâches de service, et nécessitant de nombreux capteurs

extéroceptifs, sont complexes, difficiles à mettre en œuvre, et encore sujets à de nombreuses études théoriques. De plus, le poids, la taille, et la rigidité des actionneurs induisent des phénomènes dangereux pour des applications en contact avec l'homme. Une alternative à ces problèmes est l'utilisation d'actionneurs petits, légers et avant tout compliants. Les muscles artificiels de McKibben, inventés dans les années 50 par le physicien Joseph L. McKibben (Schulte, 1961), sont étudiés au sein du LESIA depuis plusieurs années (Tondu et Lopez, 2000), et semblent répondre à ces exigences. Depuis leur invention jusqu'à leur industrialisation[1], les muscles ont été sujets à de nombreuses modifications (E.P.W., 1984; Inoue, 1988; Immega, 1986; Greenhill, 1993) et utilisés dans plusieurs projets de recherche de robotique de service (Brigeston Corporation, 1987; Brigeston Corporation and Taicubo Engeneering, 1993; Noritsu et al., 1996; Pack et al., 1997). Cependant ces muscles sont encore à l'état d'évaluation, et aucun robot commercialisé ne les utilise. Alors que toutes les premières études concernaient la validation de la commande des muscles et la faisabilité d'applications robotiques comportant des muscles, des travaux actuels sont initiés sur l'évaluation de certaines performances comme la fatigue et l'usure des muscles (Klute et Hannaford, 1998). Ceci prouve que les muscles sont aujourd'hui envisagés par la communauté scientifique pour des applications hors laboratoire.

Dans le cadre d'applications sûres (au sens de la sécurité), nous proposons d'effectuer ici une analyse du risque lié à l'utilisation des muscles artificiels. Il ne s'agit pas de faire une analyse des performances, comme par exemple les précisions atteintes, car nous nous limiterons au domaine de la sécurité. Ainsi, à titre d'exemple, une dégradation de la précision d'un actionneur peut être considérée comme un phénomène dangereux et donc sujet à une analyse, alors que la propriété inhérente d'un actionneur à être moins précis qu'un autre par sa construction même ne nous intéressera pas ici. En d'autres termes, il ne s'agit pas de démontrer ici que les muscles artificiels sont utilisables pour certaines applications et pas pour d'autres, car cela relève d'une analyse des besoins fonctionnels, qui doit être faite par tout concepteur désirant utiliser un actionneur. Ce que nous présentons ici est une analyse des risques induits par l'emploi des muscles artificiels, qui peut être utilisée en complément des documents décrivant le fonctionnement du muscle. Pour cela, ce chapitre suit les étapes de l'analyse du risque et la maîtrise du risque. L'évaluation du risque ne sera pas abordée en tant que section car elle repose sur des concepts fortement liés à l'application robotique. Comme nous souhaitons aborder le muscle dans une approche générique, nous n'aborderons pas cette activité.

Tout d'abord l'analyse du risque consiste à définir le système (le muscle artificiel) et ses caractéristiques principales, puis à décrire son utilisation. Une utilisation des muscles est présentée sur un bras anthropomorphe sept axes développé au LESIA. La seconde étape de l'analyse du risque concerne l'identification des situations et des phénomènes dangereux, en esti-

[1]Par exemple le constructeur Allemand FESTO, www.festo.com, propose une gamme d'actionneurs du type muscles artificiels

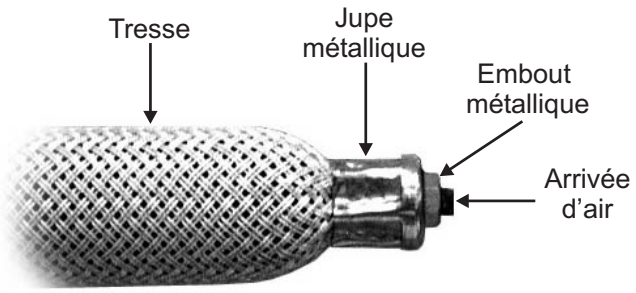

FIG. 2.1 – Photo de l'extrémité d'un muscle artificiel

mant les risques associés. Une troisième section détaille les moyens possibles de maîtrise des risques liés au muscle artificiel. Enfin, une dernière partie conclut en effectuant une comparaison des bénéfices en terme de sécurité par rapport aux autres actionneurs.

2.2 Description du muscle artificiel

Cette section décrit les principales caractéristiques du muscle artificiel en terme de conception et de performances. L'accent est mis sur les données concernant les énergies échangées, et fournit les derniers résultats sur les muscles fabriqués au LESIA. Pour certains détails de la conception, du principe de fonctionnement, ou des modèles mathématiques du muscle, le lecteur pourra se reporter à l'article de Tondu et Lopez (2000). Nous donnons ici une synthèse de certains de ces éléments.

2.2.1 Description générale

2.2.1.1 Fabrication du muscle

Les muscles artificiels fabriqués actuellement au laboratoire consistent en un tube de caoutchouc, entouré d'une tresse faite de fibres d'un textile très résistant, et de deux bouts de tube métallique (jupe métallique) permettant de sertir les extrémités. Une photo de l'extrémité correspondante est présentée sur la figure 2.1, et la vue schématique correspondante est donnée figure 2.2.

À l'opposé d'un pneu qui contient une chambre à air dans un volume fixe mais dont la pression interne augmente, la tresse permet au tube de caoutchouc de moduler son gonflement et donc sa forme. Le tressage suivant un réseau pantographe flexible permet de convertir les forces exercées sur la circonférence en une force mécanique de contraction axiale. Ainsi, lorsque la pression dans le tube de caoutchouc augmente, le muscle se rétracte. Ce principe a mené à l'identification de différents paramètres :

– α_0 est l'angle de tresse, qui correspond à l'angle d'une fibre de la tresse par rapport à l'axe du muscle, en considérant le muscle au repos (non gonflé). De ce paramètre dépend la conversion de l'énergie de pression de circonférence en force axiale.

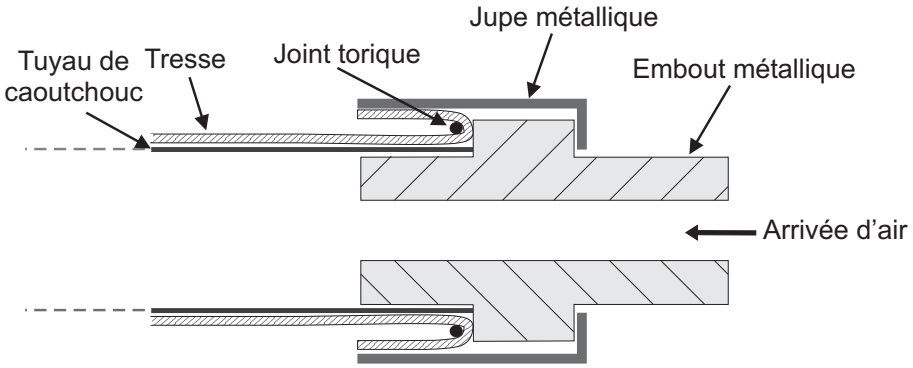

FIG. 2.2 – Coupe d'une extrémité d'un muscle artificiel avant le sertissage de la jupe métallique

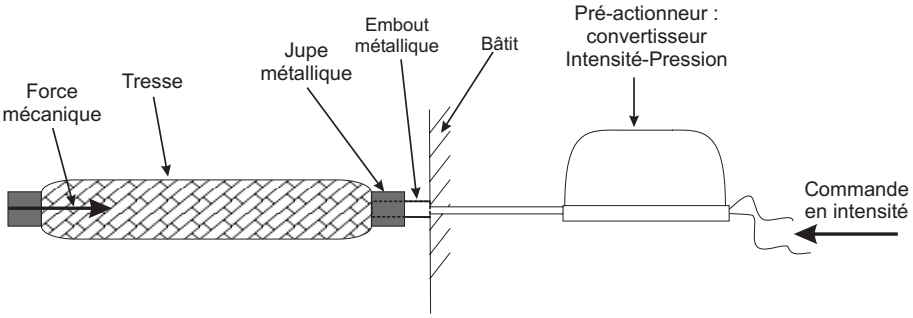

FIG. 2.3 – Branchements entre muscle artificiel et son pré-actionneur

- l_0 définie comme la longueur active initiale du muscle. La partie active est celle qui se rétracte lors de la mise sous pression, elle correspond donc à la longueur de la tresse entre les deux anneaux de métal.
- r_0 correspond au rayon du tube de caoutchouc, sachant qu'il doit être identique au rayon de la tresse pour de meilleures performances.

De plus l étant la longueur considérée à un instant t, on définit le taux de contraction par $\varepsilon = (l_0 - l)/l_0$. Les trois paramètres présentés ci-dessus, ont une influence directe sur les performances des muscles, la section suivante présente quelques résultats sur les forces dynamique et statique.

Pour les applications de robotique de service, on pourrait ajouter les paramètres que sont le volume maximal d'occupation d'un muscle, ainsi que son poids. Nous verrons ultérieurement que ces caractéristiques sont très différentes des celles des autres actionneurs.

2.2.1.2 Utilisation du muscle

Dans son utilisation la plus courante, le muscle est commandé par un pré-actionneur qui permet de convertir une tension ou une intensité, en pression comme présenté sur la figure 2.3. Il est important de souligner toute l'importance de ce pré-actionneur, notamment pour l'estimation des puissances instantanées du muscle données ci-après. En effet les performances

du pré-actionneur influent directement sur les performances du muscle, comme par exemple le temps de réponse, et donc la vitesse de déplacement. Nous utilisons au sein du LESIA des convertisseurs intensité/pression de la société Samson[2]. Ces préactionneurs prennent en entrée une intensité comprise entre 4 mA et 20 mA, et régulent la sortie en pression entre 0 et 5 bars. Sanchez (2000) propose dans sa thèse une modélisation complète de ce pré-actionneur permettant d'identifier la totalité de la boucle de rétroaction.

2.2.2 Évaluations des efforts en statique et dynamique

2.2.2.1 Force statique

Le lecteur peut se référer à l'article de Tondu et Lopez (2000) pour une expression des modèles mathématiques des forces statiques et dynamiques du muscle. Les auteurs exposent différentes conclusions ayant trait à la force statique :

- la force statique est globalement proportionnelle à l'aire de la section d'un muscle ($\pi * r_0^2$),
- la force statique est globalement indépendante de la longueur initiale l_0,
- la force statique est globalement proportionnelle à la pression de commande P,
- la force maximale est inversement proportionnelle à l'angle de tresse α_0,
- la force statique est proportionnelle au taux de contraction avec un facteur proportionnel à la pression de commande P.

De la même manière on présente une synthèse des conclusions relatives à la force dynamique :

- la contraction du muscle est naturellement diminuée par une friction cinétique non-linéaire inhérente à la tresse,
- la réponse varie entre 0,1 et 1 seconde selon le volume du muscle.

D'une manière générale, la gamme des efforts exercés par les muscles que nous utilisons au laboratoire, peut monter jusqu'à 400 daN[3]. Un graphique représentant les forces statiques que les muscles peuvent déployer est fourni sur la figure 2.4. Ce graphique donne pour chaque muscle, la force qu'il déploie à pression maximale (5 bars), et pour des longueurs de muscles fixées grâce à un tire-fort. Le choix de notation de ces muscles correspond à l'utilisation que l'on en fait au laboratoire pour le bras anthropomorphe sept axes où six types de muscle ont été utilisés. Les caractéristiques pour chacun des muscles sont :

- Muscle type 1 : $l_0 = 230mm$, $\alpha_0 = 17,1°$, $r_0 = 12\ mm$, poids=236 g ;
- Muscle type 2 : $l_0 = 196mm$, $\alpha_0 = 17,1°$, $r_0 = 12mm$, poids=223 g ;
- Muscle type 3 : $l_0 = 190mm$, $\alpha_0 = 19,8°$, $r_0 = 12mm$, poids=220 g ;
- Muscle type 4 : $l_0 = 193mm$, $\alpha_0 = 18,9°$, $r_0 = 8,5mm$, poids=86 g ;
- Muscle type 5 : $l_0 = 141mm$, $\alpha_0 = 20,1°$, $r_0 = 8,5mm$, poids=81 g ;
- Muscle type 6 : $l_0 = 118mm$, $\alpha_0 = 23°$, $r_0 = 7\ mm$, poids=46 g.

[2]http ://www.samson.de

[3]Une force de un décanewton (daN) équivaut approximativement à une masse de un kilogramme

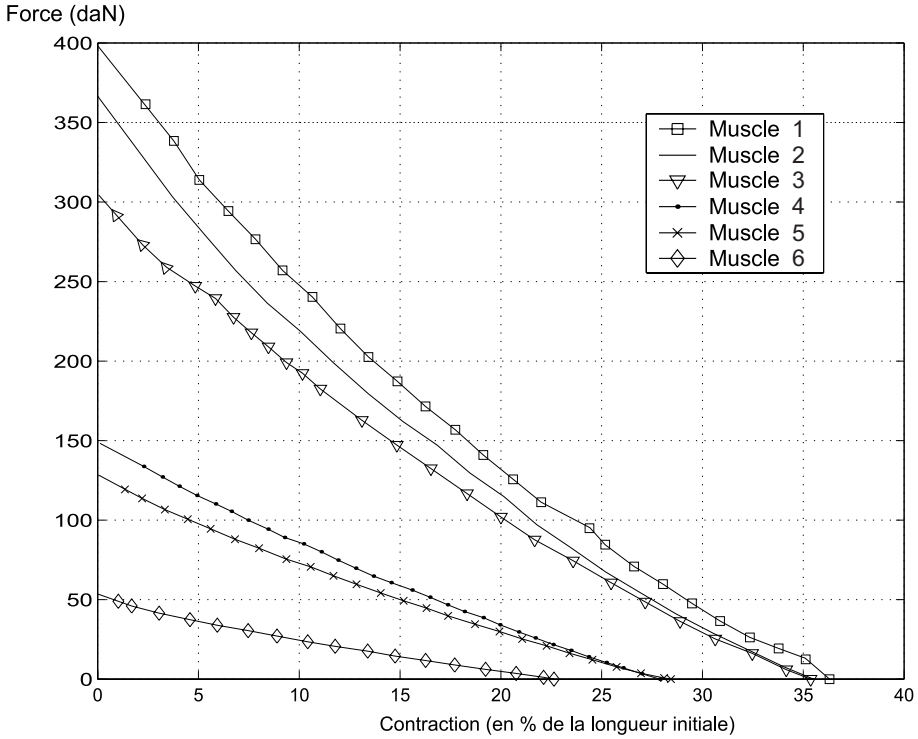

FIG. 2.4 – Force statique de plusieurs muscles à 5 bars

Pour des rayons de tube plus importants on peut imaginer que les efforts puissent atteindre des valeurs entre 400-500 daN. Ceci nous amène à conclure que ces actionneurs sont extrêmement puissants par rapport à leur poids et à leur volume d'occupation, notamment par rapport aux moteurs.

2.2.2.2 Force dynamique

Le protocole d'expérimentation utilisé depuis plusieurs années au laboratoire (Tondu et Lopez, 1997) consiste en une charge soulevée par un muscle grâce à une poulie. Le site expérimental que nous avons amélioré est présenté sur la figure 2.5. La force déployée en dynamique par ces actionneurs est représentée sur la figure 2.6, pour une masse de 15 kg, et un muscle de type 6. La masse à soulever est à l'origine posée sur un socle, et la force est donc nulle au départ. L'envoi d'un échelon de 5 bars permet alors d'effectuer une mesure de la force dynamique maximale.

À partir de ces mesures, on déduit la vitesse de déplacement, puis la puissance définie par le produit entre la force dynamique et la vitesse de déplacement. Cette puissance identifiée en dynamique est représentée sur la figure 2.7 où l'échelle est modifiée pour la lisibilité de la courbe. La valeur réelle de la puissance maximum pour cette expérimentation est de 700 daN.cm/s, soit de 70 Watts. On obtient alors un rapport puissance sur masse de 0.3. Cependant malgré la valeur élevée de ce rapport, il convient de noter que ces tests ont été effectués en dynamique sur une impulsion de pression, et que les rapports des moteurs pas

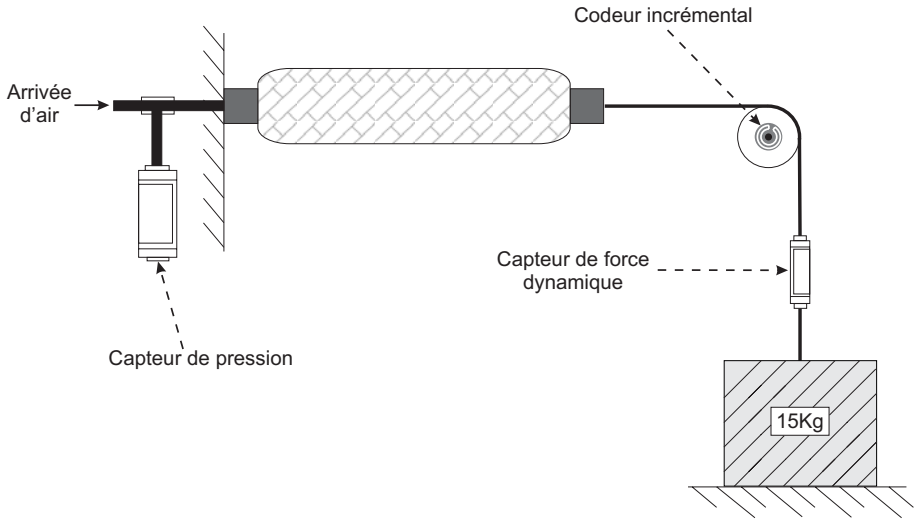

FIG. 2.5 – Site expérimental pour les tests d'un muscle en dynamique

à pas par exemple, qui sont de l'ordre de 0.1 (Hannaford et Winters, 1990), ont été calculés à vitesse constante de déplacement. Ce test est évidemment très difficile à effectuer pour les muscles artificiels, dont le débattement est limité (à l'opposé d'un moteur qui peut tourner à vitesse constante indéfiniment).

De plus, les forces dynamiques atteintes par les plus petits muscles (ceux du type 5) sont du même ordre de grandeur que celles des plus gros muscles (type 1 et 2). La figure 2.8 montre les forces dynamiques d'un muscle de type 1 et d'un muscle de type 6, pour une charge soulevée de 10 kg. Il est important de noter que la force dynamique, proportionnelle à la charge soulevée, reste très élevée, même pour les plus petits muscles que l'on utilise dans notre laboratoire.

2.2.3 Compliance

Le terme de *compliance*, utilisée dans ce chapitre fait référence à la *compliance passive* présentée dans la section 1.6.1.2. Cette propriété, parfois appelée *complaisance*, est calculée comme l'inverse de la raideur. Elle donne une mesure sur la souplesse naturelle du muscle. Cette caractéristique constitue l'intérêt principal de ces actionneurs. Elle est une conséquence directe du composant de type ressort du muscle artificiel. Il n'existe en effet aucun autre actionneur capable de donner cette souplesse naturelle, que ce soit parmi les moteurs électrique ou les autres actionneurs pneumatiques.

2.2.4 Actionnement d'une articulation

Dans beaucoup d'applications robotiques, le muscle est utilisé dans un montage comportant deux muscles antagonistes permettant d'effectuer une rotation d'un axe suivant le principe présenté sur la figure 2.9. Les différents modèles mathématiques de l'actionnement sont pré-

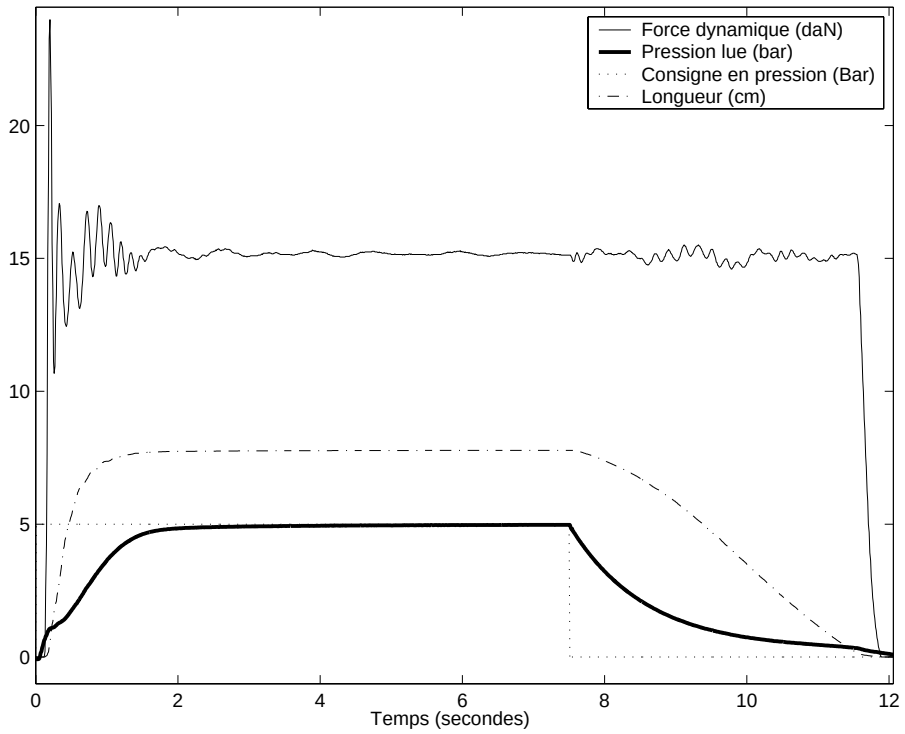

FIG. 2.6 – Réponse en longueur et force dynamique d'un muscle à un échelon pour une charge de 15 kg

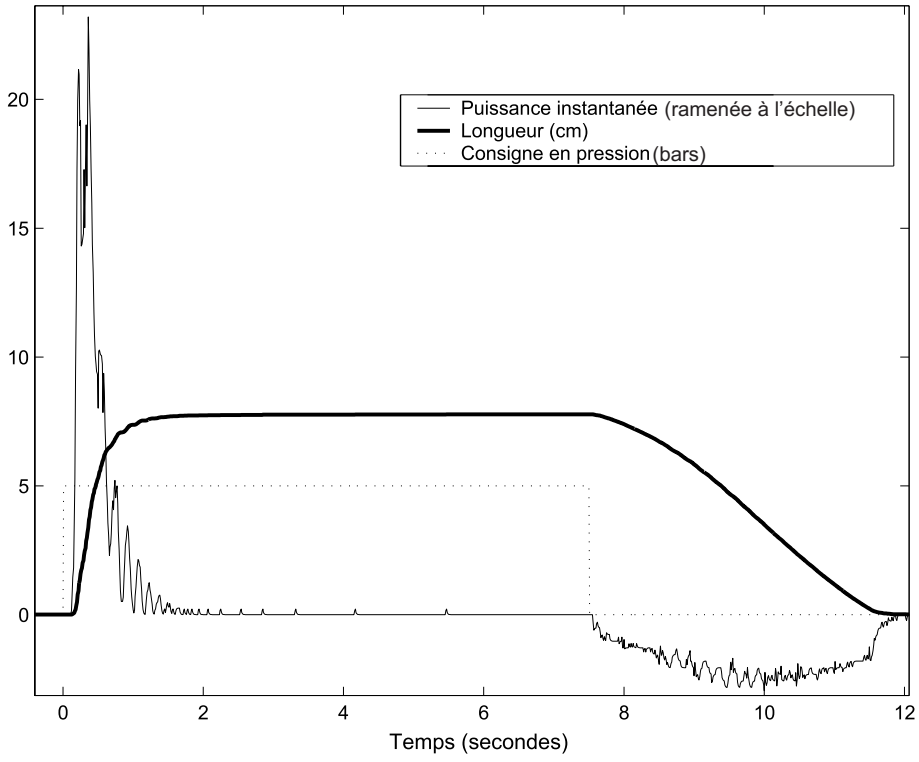

FIG. 2.7 – Puissance instantanée en réponse à échelon pour une charge de 15 kg

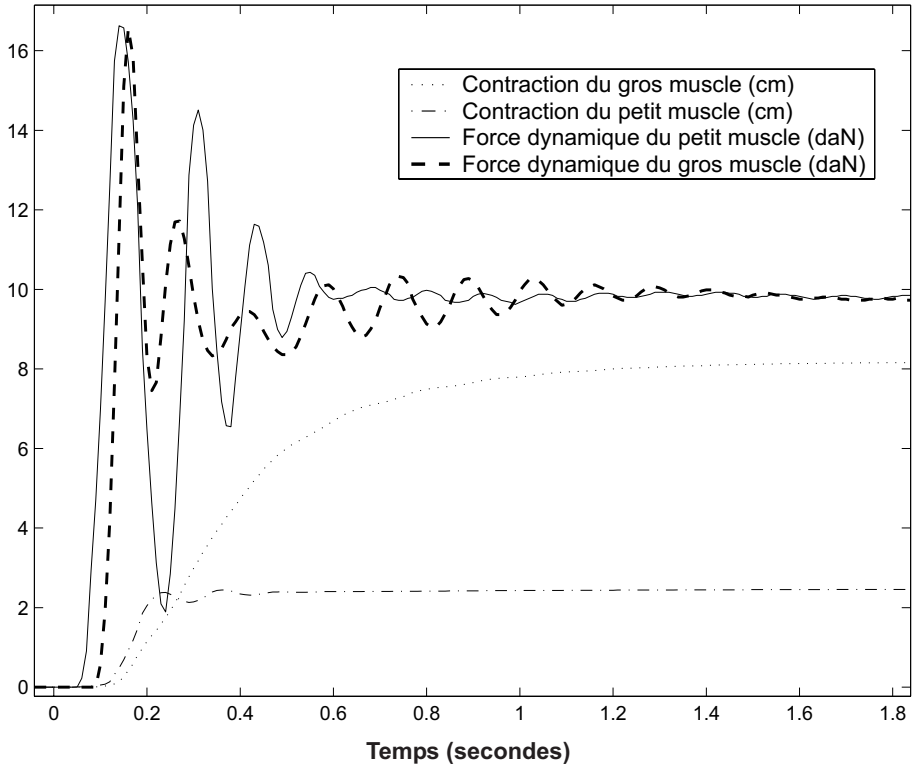

FIG. 2.8 – Forces dynamiques de deux muscles pour une masse soulevée de 10 kg

sentés dans l'article de Tondu et Lopez (2000). Cependant, l'apparition des robots parallèles, et notamment pour des tâches de service, a rendu possibles d'autres types d'actionnement. Le muscle peut par exemple être utilisé dans sa longueur comme pour le robot télé-échographe TER présenté au chapitre 5.

2.2.5 Contrôle des muscles artificiels

Les travaux de recherche sur la commande des robots manipulateurs actionnés par des muscles artificiels de McKibben menés au laboratoire, ont démontré que plusieurs techniques d'asservissement apportent des performances de précision satisfaisantes pour des applications de service. Les travaux de Boitier (1999) et de Carroll et al. (1997) démontrent que la commande PID[4] avec terme d'anticipation[5] et la commande à Structure Variable sont utilisables pour un robot SCARA[6] à deux DDL (Degrés De Liberté). D'autres commandes comme la commande floue (Vial, 1997), la commande plate (Sanchez, 2000), ou la commande par réseaux de neurones (Eskiizmirliler et al., 2002), ont aussi apporté des résultats satisfaisants en terme de précision.

[4]Régulateur comportant des calculs à partir de l'erreur entre la position désirée et la position réelle de type Proportionnel, Intégrateur et Dérivée

[5]Sur la base de modèles de réponse connus des axes du robot, ce terme vient anticiper la réponse de l'axe commandé

[6]Les robots de ce type effectuent leurs mouvements dans un plan

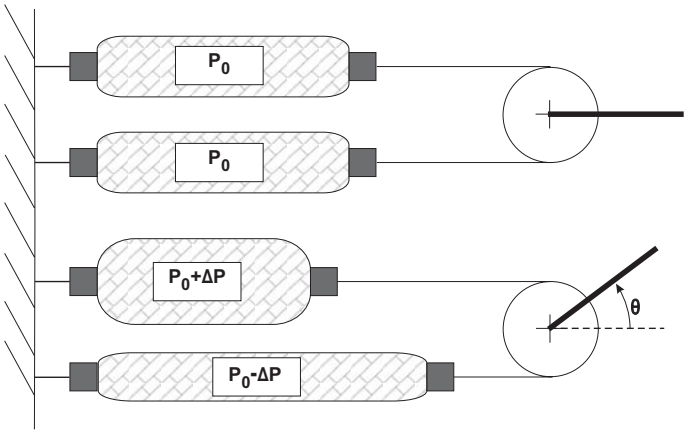

FIG. 2.9 – Principe de l'actionnement d'une articulation au moyen de deux muscles antagonistes

Bien que les résultats de ces différents travaux sur l'asservissement en position des muscles soient satisfaisants pour les applications développées au sein du laboratoire, ils comportent tout de même certains points négatifs. Tout d'abord en terme de précision, les résultats sur un muscle sont très satisfaisants car on obtient des précisions inférieures au millimètre, mais lorsque l'on passe à une structure comportant plusieurs muscles, comme une articulation, la non-linéarité propre à chaque muscle mène à des calculs complexes et lourds pour les calculateurs. Certaines solutions s'avèrent alors difficilement utilisables en raison des contraintes temps réel. Cependant, pour certaines applications ne nécessitant pas une précision inférieure au millimètre, ou dont la structure rend difficile la commande (comme des structures parallèles), il est possible d'utiliser une simple commande en boucle ouverte grâce à la stabilité naturelle du muscle en boucle ouverte.

Cette souplesse naturelle du muscle, rend les applications robotiques travaillant dans l'espace, fortement dépendantes de la gravité. En effet, pour un robot de type SCARA dont les mouvements se font dans un plan, la force de gravité sera toujours orthogonale aux axes du robot. Mais pour un robot comme un bras articulé à plusieurs axes, les efforts exercés sur une articulation dépendront de la configuration des axes du robot. Par exemple, pour un bras anthropomorphe tendu à l'horizontale, l'effort que devra fournir une épaule sera moins important si l'on plie le coude. On déplace alors le centre de gravité vers l'épaule. Cette problématique doit encore être étudiée car cela peut générer des asservissements complexes.

2.2.6 Validation sur un bras sept axes anthropomorphe

Le LESIA[7] a développé avec le laboratoire de mécanique de l'INSA de Toulouse, un bras anthropomorphe sept axes actionné exclusivement par des muscles artificiels (Tondu et al., 2003). Ces 7 axes correspondent à 7 liaisons de rotation comme présenté sur la figure 2.10. La finalité était une validation expérimentale de l'actionnement grâce aux muscles artificiels

[7]Laboratoire d'Étude des Systèmes Informatiques et Automatiques

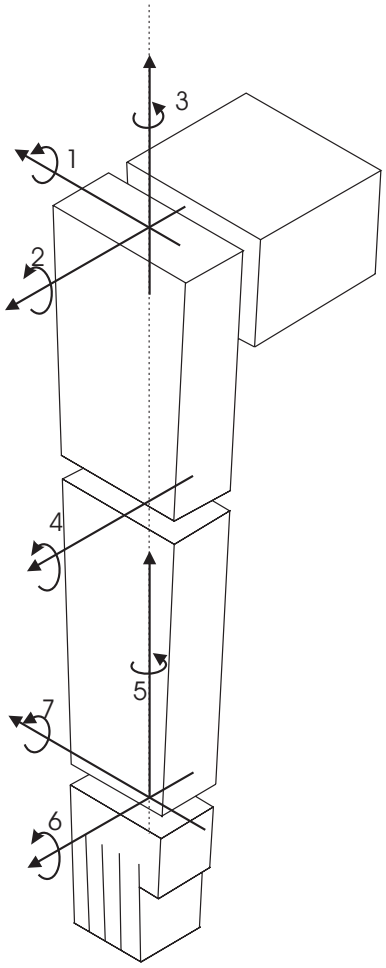

FIG. 2.10 – Sens de rotation des 7 axes du bras anthropomorphe

sur une structure robotique complexe. La structure mécanique reproduit les sept articulations du bras humain, en se limitant à une taille humaine. Des photos de ce bras sont données figures 2.11, 2.12 et 2.13.

Le contrôleur de ce robot a été développé en UML, et implanté en C++ sur le noyau temps réel VxWorks. Une commande en boucle ouverte et un simple proportionnel ont permis de valider cette technologie et d'obtenir des premiers résultats expérimentaux. Ce bras permet notamment d'effectuer des tâches de service en téléopération comme la pose d'un verre représenté sur la figure 2.14. Le fait d'être en téléopération a réduit la complexité de la loi de commande possible pour contrôler les mouvements de ce bras. Mais cela nous a permis de valider la faisabilité d'une telle structure, et de prochaines études automatiques devraient être réalisées au sein du laboratoire. Pour réaliser la téléopération, l'opérateur utilise deux joysticks et un palonnier qui lui permettent de contrôler chaque articulation du robot. Si l'apprentissage s'avère complexe pour l'opérateur, nous avons pu réaliser et valider des tâches comme la pose d'un verre, le suivi d'une surface et la rencontre d'obstacles, et ainsi valider l'apport très important de la compliance pour la réalisation et la sécurité de ces tâches complexes.

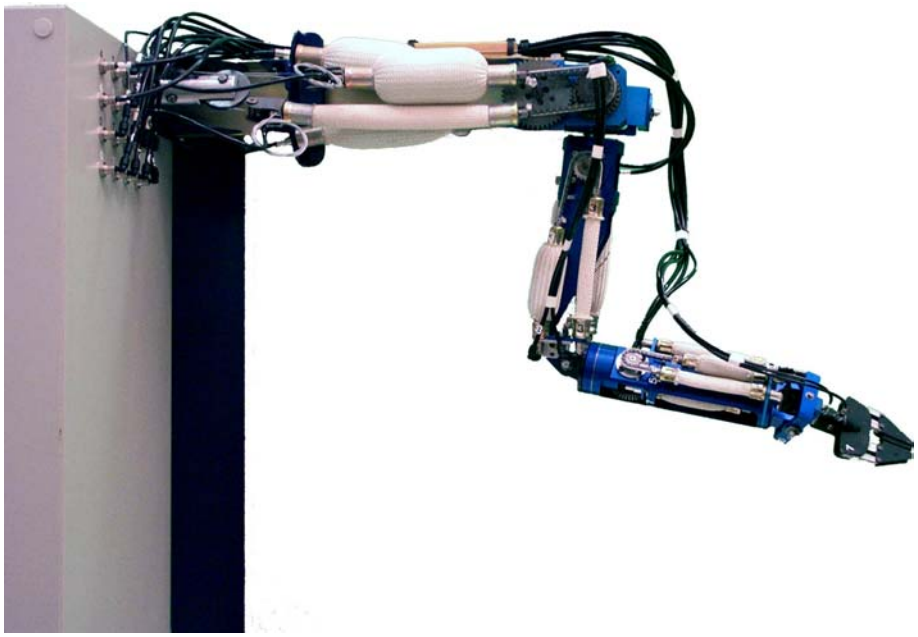

FIG. 2.11 – Vue de dos du bras anthropomorphe 7 axes

FIG. 2.12 – Vue de face du bras anthropomorphe 7 axes

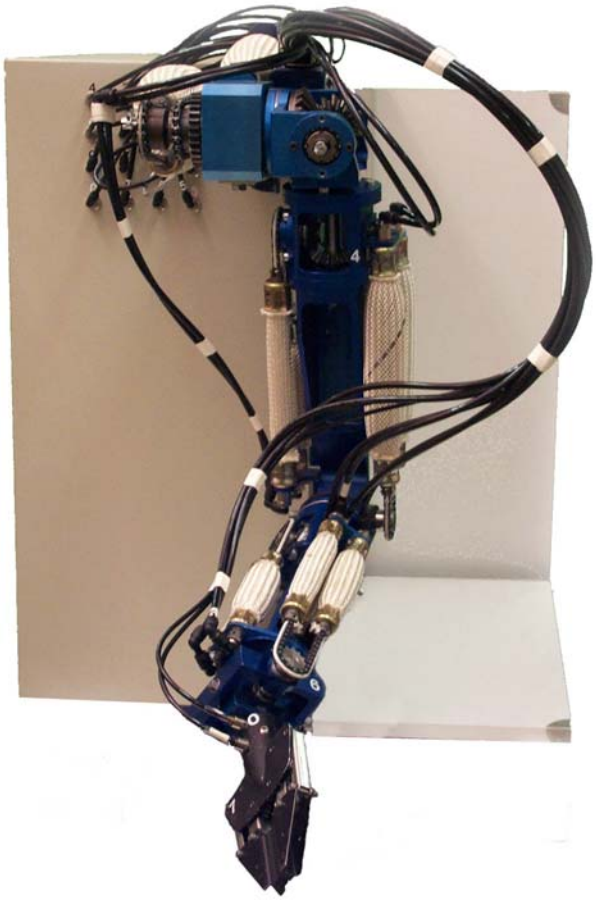

FIG. 2.13 – Vue de profil du bras anthropomorphe 7 axes

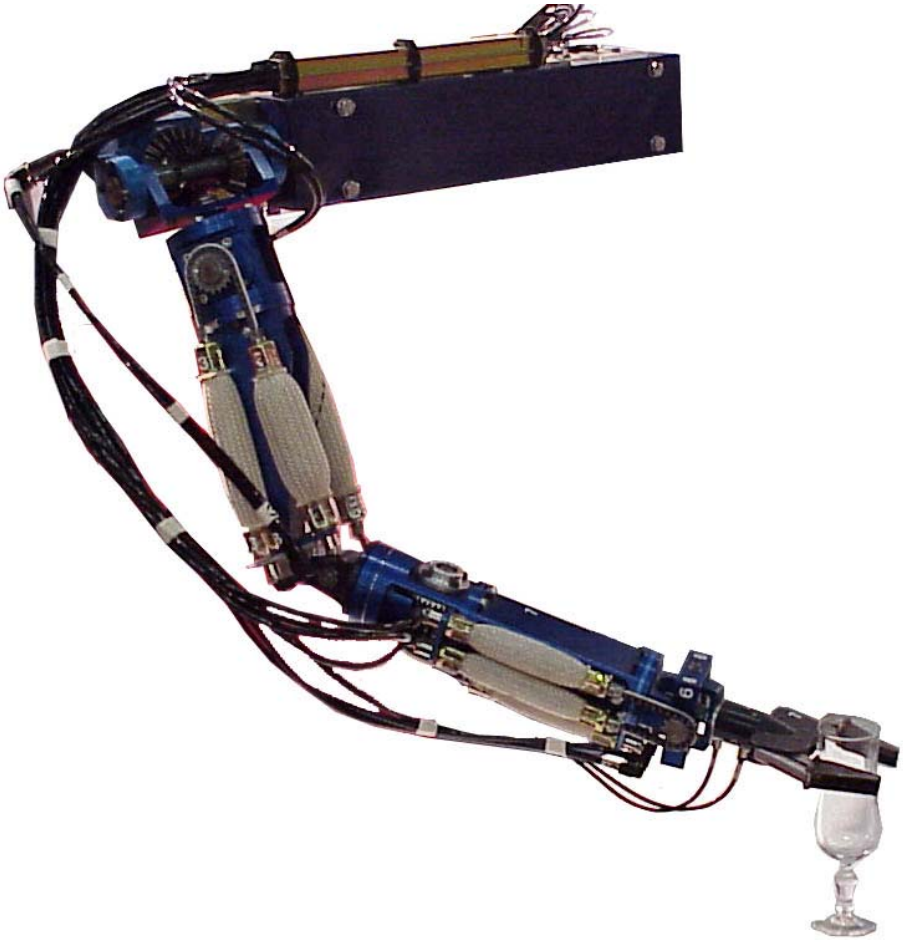

FIG. 2.14 – Manipulation d'un verre par le bras anthropomorphe 7 axes

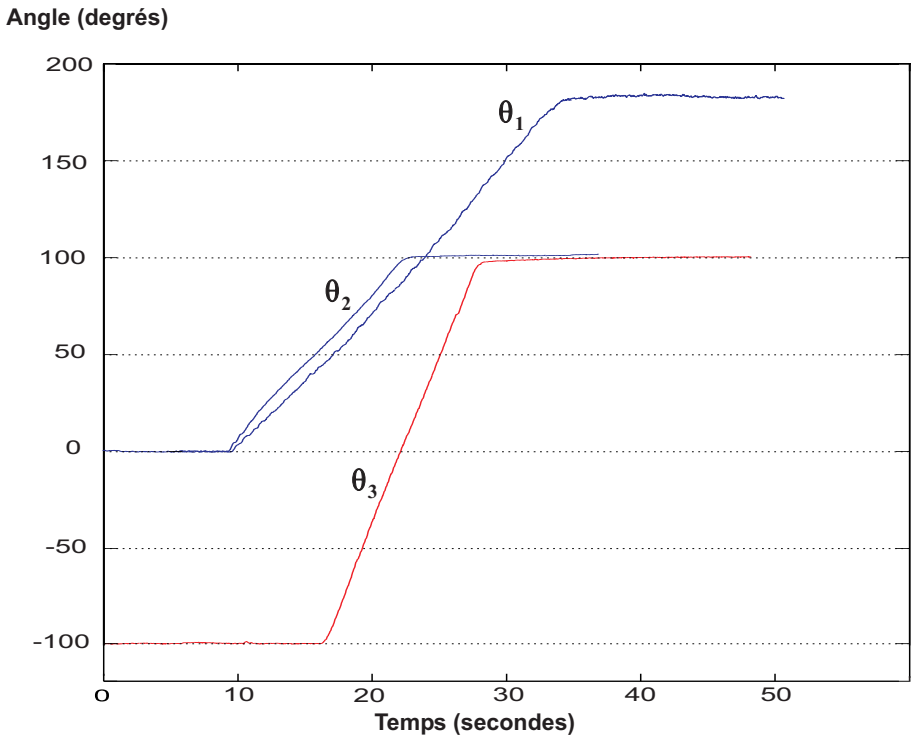

FIG. 2.15 – Débattements des axes 1 à 3 du bras anthropomorphe 7 axes

D'un point de vue mécanique, les débattements que nous avons obtenus sont proches de ceux du bras humain et sont représentés sur les figures 2.15 et 2.16.

2.3 Identification des dangers et estimation du risque

À partir des caractéristiques du muscle artificiel, on analyse ici les dangers induits par l'utilisation de ces muscles. En référence au premier chapitre, la notion de *danger* que l'on emploie ici inclut, comme cela est présenté dans la section 1.3.6, les notions de *phénomènes dangereux*, *situations dangereuses*, et *événements dommageables*. En premier lieu nous analyserons les phénomènes dangereux que sont les défaillances du muscle. Et pour cela nous étudierons les modes de défaillance des composants du muscle, pour en déduire les modes de défaillance du muscle lui-même. Il est évident qu'un muscle artificiel ne peut constituer à lui seul une source de dommage et c'est le contexte d'utilisation qui permet de déterminer les dangers induits par le muscle artificiel. En conséquence, nous analyserons les situations dangereuses les plus communes et que l'on peut identifier de manière générale sans se référer à une application robotique particulière.

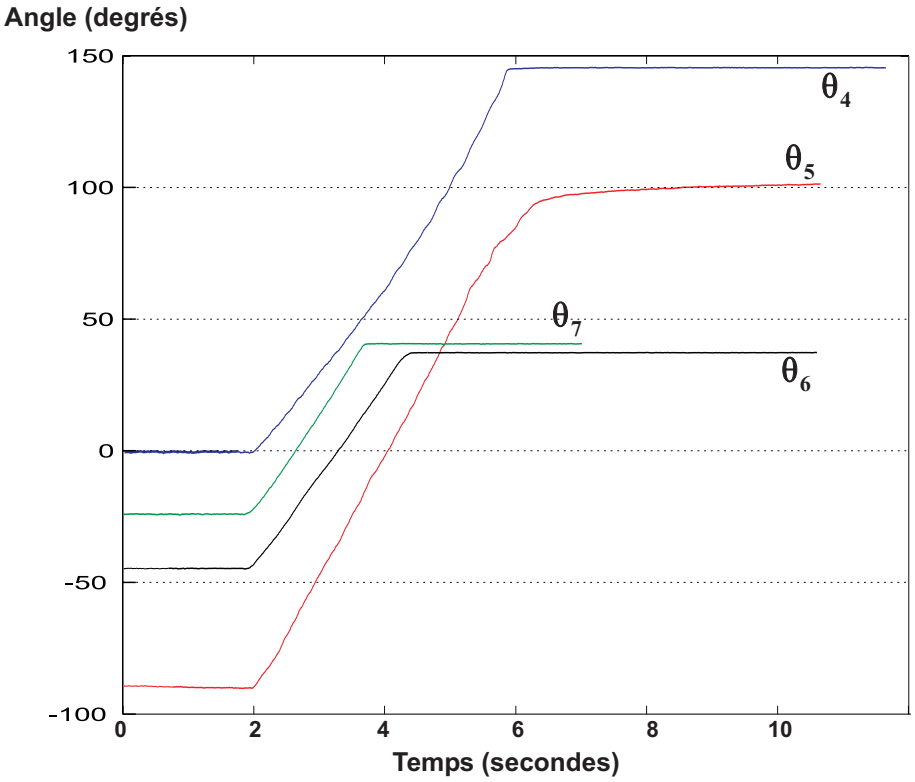

FIG. 2.16 – Débattements des axes 4 à 7 du bras anthropomorphe 7 axes

2.3.1 Phénomènes dangereux de type défaillances

Pour cette section nous utilisons la technique de l'AMDEC présentée au chapitre 1. Par souci de clarté, ne sont présentées ici que les analyses des composants les plus importants : la tresse et son système de maintien, et le tube de caoutchouc.

2.3.1.1 Défaillances de la tresse et de son système de maintien

Ces composants dont un schéma a été présenté sur la figure figure 2.2 conduisent à l'identification de deux modes de défaillance principaux :

– la rupture totale de la tresse,
– et le désengagement de la tresse hors de la zone de pincement formée entre l'embout métallique et la jupe métallique.

Certains autres modes de défaillances comme l'étirement de la tresse ou la déformation du système de maintien ne sont pas envisagés car ils sont physiquement impossibles.

La rupture totale de la tresse est une défaillance qui n'a pas encore été observée dans la pratique, et cela pour la simple raison que cette tresse est formée de plusieurs centaines de fibres d'un textile très résistant. Bien que des essais réels n'aient pas été effectués, on peut penser que cette tresse puisse résister à plusieurs tonnes d'étirement. Dans le cadre de la robotique de service, les efforts n'atteignent jamais des valeurs aussi importantes. On peut donc

Composant	**Mode de défaillance**	**Causes**	**Effets**	Risque			**Moyens de détection possibles (en ligne)** a. Mode défaillance b. Effets	**Solutions possibles** : a. Prévention b. Protection c. Autres actions d. Remarques
				Gravité	Occurrence	Risque		
Tresse	Rupture tresse	a. Mauvaise conception b. Mauvaise utilisation	a. Force mécanique transmise nulle b. Flot d'air c. Projections	1	I	I	b. Capteur de pression	a. Définir règles d'utilisation b. Entourer les muscles d'un matériau de protection
Système de maintien	Désengagement de la tresse	c. Mauvaise conception d. Mauvaise utilisation	d. Force mécanique transmise nulle e. Flot d'air f. Projections	1	P	H	b. Capteur de pression	a. Définir règles d'utilisation Pincement de la jupe métallique plus important b. Entourer les muscles d'un matériau de protection

FIG. 2.17 – Analyse des modes de défaillance de la tresse et du système de maintien

estimer la probabilité d'occurrence d'une telle défaillance comme *invraisemblable* suivant l'échelle qualitative présentée à la section 1.4.3.4.

Le désengagement de la tresse hors de la zone de pincement est un phénomène plus probable. Sur l'ensemble des manipulations des muscles artificiels au sein du LESIA depuis une dizaine d'années, un seul cas de désengagement total a été observé. Il n'a provoqué aucun dommage grave, et l'on peut qualifier l'événement d'incident. Puis, quelques cas d'un désengagement progressif sur une année sans qu'il y ait désengagement total ont été observés suite à une modification de la conception. C'est pour cette raison que nous attribuons le niveau de *probable* à ce mode de défaillance.

L'effet direct de ces modes de défaillances est en premier lieu la rupture de la chaîne cinétique, produisant une force nulle aux deux extrémités du muscle, et qui peut produire un dommage sur le système en fonction de l'architecture du robot choisie. Cependant, il convient de prendre en compte les effets pouvant produire des dommages directement vers les autres composants du système et sur son environnement, et notamment les acteurs humains se trouvant à proximité. Tout d'abord, une rupture totale peut entraîner des projections d'éléments comme les pièces métalliques composants le muscle, puis le volume d'air n'étant plus contraint, cela mène à un flot d'air incontrôlé délivré par le préactionneur.

Le système le plus simple pour détecter ces modes de défaillance est l'introduction de capteurs de pression en amont du muscle, reliés soit à un système de coupure de la pression, soit à un contrôleur capable de prendre une décision. Les moyens de prévention concernent alors la fabrication du muscle, et sont abordés ultérieurement dans la section relative à la maîtrise du risque.

L'ensemble de ces éléments peut être résumé dans un tableau de type AMDEC, présenté sur la figure 2.17. Sur ce tableau sont présentés des solutions envisageables sur lesquelles nous reviendrons dans la section 2.4. Cependant, il est déjà possible d'identifier deux causes principales pour ces modes de défaillance qui sont une mauvaise conception (tresse fragile ou abîmée), et une mauvaise utilisation (proximité d'objet tranchant, frottements, etc.), et d'en déduire des moyens de prévention et de protection que nous détaillerons dans la section 2.4.

Composant	**Mode de défaillance**	**Causes**	**Effets**	Risque			**Moyens de détection possibles (en ligne)** a. Mode défaillance b. Effets	**Solutions possibles** : a. Prévention b. Protection c. Autres actions d. Remarques
				Gravité	Occurrence	Risque		
Tube de aoutchouc	Crevaison	a. Mauvaise conception b. Mauvaise utilisation	a. Chute de la pression et dérive de la contraction vers zéro b. Flot d'air c. Bruit		F		b. Capteur de pression, capteur de longueur	a. Règles d'utilisation b. Protection autour des muscles

FIG. 2.18 – Analyse du mode de défaillance du tube de caoutchouc

2.3.1.2 Défaillance du tube de caoutchouc

Il n'existe qu'un mode défaillance de ce composant, mais comportant plusieurs degrés. Il s'agit d'une fissure dans le tube, plus communément appelée crevaison. Cette fissure peut être de différentes tailles, et donner lieu à une crevaison importante ou à une simple fuite. Dans tous les cas, les causes de cette défaillance sont soit liées à une mauvaise conception (fragilité du caoutchouc), soit à une mauvaise utilisation (hors limites, intensive, etc.). L'AMDEC de ce mode de défaillance est donné figure 2.18. Il convient de noter qu'une des causes de la crevaison est l'usure naturelle du tube. Elle est incluse dans la cause *Mauvaise utilisation*, puisque que l'on considère alors que l'utilisation d'un muscle usagé ou trop vieux est une erreur de l'opérateur.

Pour ce mode de défaillance, il est important de souligner que l'effet n'est pas comme précédemment, une force mécanique nulle, car le muscle s'étirera au maximum jusqu'à sa longueur à pression nulle. L'effet est donc une dérive de la longueur du muscle, qui peut être traduit comme une diminution de la force transmise vers une valeur fixe. Par exemple, dans le cas d'une masse suspendue à un muscle, la masse ne sera pas livrée à la seule force de son poids, car la tresse du muscle délivrera au minimum à elle seule une force de réaction permettant de maintenir la masse. Ceci est fondamental pour la suite de l'analyse. Les effets en terme de longueur et de pression sont illustrés sur les figures 2.19 et 2.20, dans les mêmes conditions de test que pour l'identification de la force dynamique, avec une masse de 15 kg, et un des plus gros muscles que l'on utilise ($l_0 = 230mm$, $\alpha_0 = 17,1°$, $r_0 = 12mm$). À partir du troisième créneau on peut observer le phénomène de crevaison qui n'existait pas jusqu'alors. D'autres tests ont montré que ce mode de défaillance se comportait globalement toujours de la même manière, c'est à dire en une dérive progressive de la longueur et de la pression. Ainsi, sur ces diagrammes, on peut observer, pour une commande en créneau 0-5 bars, une chute de pression à 1,75 bar en quatre paliers, et une chute de contraction de 72% en plusieurs dizaines de secondes. Ceci mène à considérer cette défaillance, comme *douce* : le muscle dérive lentement vers une valeur fixe de longueur, puis y reste. Les effets sur le système robotique utilisant le muscle sont alors directement dépendants de l'architecture choisie. Cet aspect est développé dans la section sur la maîtrise du risque (cf. section 2.4). Les autres effets à prendre en compte sont le flot d'air continu s'échappant par la fissure et le bruit associé. Ces effets peuvent notamment être une source de dommage sur l'environnement (par exemple

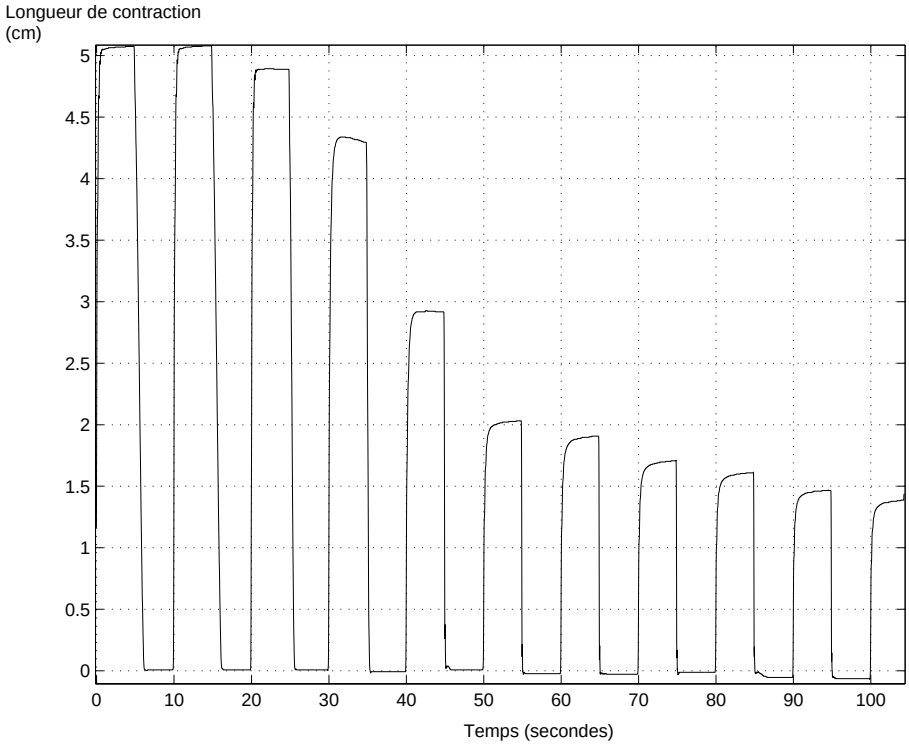

FIG. 2.19 – Réponse en longueur à une crevaison

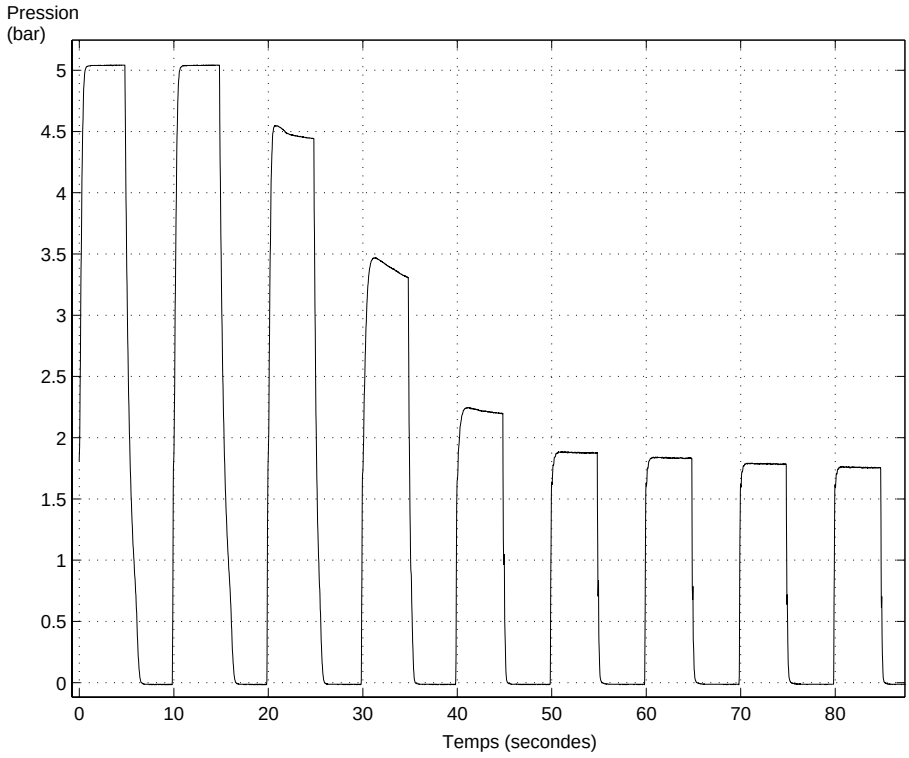

FIG. 2.20 – Réponse en pression à une crevaison

dans un milieu aseptisé), voire psychologique (peur, stress de la part des humains à proximité du muscle).

Il est alors difficile d'estimer la gravité du dommage que peut générer cette défaillance, car elle dépend évidemment de l'application robotique. Cependant, il est possible d'estimer qualitativement la fréquence d'occurrence de ce mode de défaillance à *fréquent*, car on la mesure aujourd'hui, sur les différentes plates-formes du LESIA (robotiques ou non), à plusieurs crevaisons par an. Une estimation quantitative théorique de la durée de vie des muscles a été abordée de manière mathématique par une équipe de l'université de Washington (Klute et Hannaford, 1998), mais elle concerne principalement l'utilisation de modèles mathématiques de durée de vie du caoutchouc. Or, au sein du muscle artificiel, les frottements de la tresse sur le tube, non mesurables en terme d'effets, sont difficilement intégrables à ce calcul. De plus, une estimation expérimentale comme nous l'avons faite, doit se limiter à un type de muscle, une charge donnée, mais surtout un type de commande (triangle, sinusoïde, créneau, ou constante), et les résultats sont encore difficilement exploitables. En conclusion, la crevaison est aujourd'hui encore le mode de défaillance le plus commun. Les solutions possibles pour maîtriser le risque associé à cette défaillance sont basées sur l'expertise du LESIA au travers des différents projets utilisant les muscles. Ces solutions sont développées dans la section 2.4.

2.3.2 Situations dangereuses

Contrairement à la section précédente, l'identification de situations dangereuses ne peut se faire que lorsque l'on connaît le système global où sont intégrés les muscles. Cependant, nous avons identifié deux situations dangereuses génériques.

Ainsi, dans le cas d'un bras robot, une première situation dangereuse induite par la gravité peut être identifiée. En effet, comme présenté dans la section 2.2.5, la compliance des muscles artificiels peut rendre certaines architectures fortement dépendantes de la gravité. Il est alors possible de passer par certaines configurations où le poids du robot lui-même entraîne un écart de mouvement, et ceci est fortement augmenté si le robot comporte une charge. Les effets de tels déplacements, lors d'une tâche de service, peuvent être critiques pour la sécurité, et les moyens de maîtriser ce risque résident principalement dans le contrôle des mouvements comme cela est présenté dans la section 2.4.

La deuxième situation dangereuse générique que l'on peut identifier est la contrainte d'un muscle dans une position, puis la libération instantanée. Par exemple, lors d'un contact entre un obstacle et un bras robot actionné par des muscles artificiels, le choc sera fortement réduit grâce à la compliance du muscle. Cependant, lors du relâchement de la contrainte, si la commande n'a pas été adaptée suffisamment rapidement, il est possible d'obtenir un mouvement de relâchement dont la vitesse peut engendrer une énergie cinétique dangereuse. C'est donc la souplesse elle-même qui peut encore une fois être source potentielle de dommage. Cet

événement ne se limite pas qu'aux bras robots, et peut concerner toute architecture articulée (jambes, robots parallèles, etc.).

2.4 Solutions possibles pour la maîtrise du risque

Cette section reprend les principaux moyens de maîtrise des risques abordés dans la section d'analyse du risque. Il est d'effectuer de la prévention en réduisant la probabilité d'apparition du danger, et donc du dommage. Mais on peut aussi utiliser des techniques de protection pour réduire la gravité d'un dommage.

2.4.1 Moyens de prévention

La prévention passe d'abord par la conception du muscle lui-même. Le système de maintien de la tresse par une jupe métallique est réalisé par un système de presse hydraulique permettant d'effectuer un sertissage homogène et aussi puissant que l'on désire (ou du moins jusqu'à la résistance des pièces de métal). Le matériau du tube de caoutchouc qui est utilisé aujourd'hui est du simple latex, qui fournit une durée de vie nettement supérieure au butyle ou aux autres matériaux utilisés précédemment. Ces données n'ont pas encore été chiffrées, mais le latex semble au moins dix fois plus endurant que les anciens tubes. Ce latex est néanmoins sensible aux agressions de l'environnement comme la lumière, la température et l'humidité. Pour l'utilisation des muscles, il est donc préférable de prévoir des emplacements protégés contre ces agressions. De plus, la durée de vie limitée du tube impose un programme de maintenance préventive adapté à l'application robotique concernée, et à la sollicitation des muscles[8]. Cependant, les résultats actuels de durée de vie de ces muscles sont encore assez variables et il est extrêmement difficile de proposer des règles de remplacement de ces muscles.

Les moyens de prévention de situations dangereuses sont encore à l'état de recherche. Ils concernent en effet, la commande de ces muscles, pouvant intégrer des situations dangereuses comme le contact avec un obstacle, le blocage dans une configuration donnée puis le relâchement brutal, ainsi que la gestion de la force de gravité. Ces aspects ont encore à être développés et concernent principalement les contrôleurs informatiques.

2.4.2 Moyens de protection

Des systèmes de muscles redondants comme le sont les fibres d'un muscle humain, ou des dispositifs externes aux muscles que nous n'aborderons pas ici (revêtement visco-élastique, freins sur articulations, etc.), pourraient permettre au robot de réagir en cas de défaillance d'un

[8] Sur l'utilisation même des muscles, le type de commande joue fortement sur la durée de vie de ces muscles. Par exemple des fronts de pression sont à éviter

muscle, et protéger les biens ou les personnes. Cependant un des principaux intérêts du muscle est son faible poids, et ces moyens nécessitent des montages plus lourds et volumineux.

Des systèmes de protection autour des muscles, permettent de protéger l'environnement extérieur des projections ou du flot d'air en cas de crevaison ou de rupture d'un muscle, tout en protégeant les muscles des agressions extérieures (pointes, frottements). De tels systèmes doivent néanmoins rendre possible la maintenance des muscles comme par exemple le remplacement.

En dernier point, nous voulons souligner ici toute l'importance de l'architecture du robot face aux effets des défaillances des muscles. Comme nous l'avons présenté, la principale défaillance d'un muscle étant la crevaison, le muscle s'allonge jusqu'à la longueur initiale. Dans ce cas, une architecture permettant un geste de retrait de l'effort appliqué par le robot sur l'environnement en cas d'allongement d'un muscle, permet d'obtenir une sécurité intrinsèque.

2.5 Synthèse et conclusion

Les principales caractéristiques du muscle artificiel ayant une incidence sur la sécurité ont été présentées à travers les différentes sections de ce paragraphe. Tout d'abord, la fabrication du muscle consiste en l'assemblage de matériaux résistants et limités en nombre[9] dont les deux principaux éléments sont la tresse et le tube de caoutchouc. Ce principe simple de fonctionnement réduit le nombre de défaillances possibles propres à cet actionneur. En comparaison, les modes de défaillance des autres actionneurs sont en général :

- dérive de la position,
- saut de position ou pic d'accélération,
- position ou vitesse aléatoire,
- position figée,
- flot d'un type d'énergie non prévue (choc électrique, décharge électrostatique, etc.).

Les modes de défaillance les plus dangereux sont sans doute le pic d'accélération et la sortie aléatoire. Dans le cas du muscle artificiel, les puissances atteintes sont très importantes (jusqu'à 400 kg pour un muscle de 23 cm !), et de tels modes de défaillance pourraient engendrer des dommages importants. Cependant nous avons montré qu'il n'existe qu'une défaillance probable, la crevaison, dont les effets sont une diminution « douce » de la longueur de contraction, et un flot d'air vers l'environnement extérieur. Dans ce cadre, pour de nombreuses applications robotiques cette défaillance pourra avoir un effet négligeable suivant l'architecture choisie. Par exemple, dans le cas d'un bras robot, si un muscle d'une articulation se perce, l'effet sera un mouvement lent vers une position fixe dépendant de la configuration dans laquelle se trouvait le bras. De plus nous avons montré qu'un ensemble de moyens de prévention concernant la fabrication, l'utilisation, et la maintenance, permettent de réduire la probabilité d'occurrence de la crevaison.

[9] Notons que cet aspect en fait un des actionneurs le moins coûteux du marché

Le risque le plus important est sans doute l'occurrence de situations dangereuses, induites par la force de gravité et les perturbations extérieures. Les effets peuvent être des mouvements brusques et non maîtrisés pouvant induire des dommages importants. Ce risque, peut être réduit par le contrôle des mouvements, mais cet aspect très complexe doit être encore développé. Un des axes de recherche est le contrôle de la raideur d'une articulation comme celle de la figure 2.9, en augmentant la pression de base P_0 des deux muscles antagonistes. Un autre axe de recherche est la découverte de nouvelles architectures robotiques permettant de limiter les effets de la gravité. Ainsi, il est possible de créer des architectures, comme celle du robot médical parallèle TER présenté dans le chapitre 5, pratiquement indépendantes de la gravité. De plus, de par son architecture, le robot TER limite les effets des défaillances des muscles, en réduisant la pression sur le patient en cas de crevaison d'un muscle.

Les muscles artificiels sont donc une technique nouvelle de la robotique, permettant de réduire les risques suivant l'architecture du robot. Un muscle seul est actionneur intrinsèquement sûr, car son principal mode de défaillance, la crevaison, a un effet « doux ». Cependant, si les muscles artificiels apportent une garantie supplémentaire de sécurité, il est évident qu'une commande erronée de pression les rend dangereux. Or dans un système robotique, cette commande résulte de l'interaction de nombreux éléments. Et la maîtrise de la sécurité ne peut donc venir que de la maîtrise de l'ensemble de ces éléments, soit la maîtrise de la sécurité du système. Pour cela, les chapitres suivants proposent une démarche système (chapitre 4 et 5), permettant d'analyser les risques d'un système de robotique de service, en se basant notamment sur un langage de modélisation présenté dans le chapitre 3.

Chapitre 3

La modélisation orientée objet avec UML

3.1 Introduction

Au chapitre 2 nous avons établi que les muscles artificiels constituaient une technique permettant de réduire le risque généralement associé aux actionneurs. Cependant, nous avons vu que le risque principal d'une application de robotique de service, résidait dans la complexité du système global. La maîtrise de la sécurité ne peut se réduire à l'utilisation de techniques particulières comme les muscles artificiels, mais doit intégrer une gestion des risques comme nous l'avons présenté au premier chapitre. Or, les différentes activités de la gestion du risque doivent être effectuées avant que la fabrication ait eu lieu. Pour cela les concepteurs s'appuient sur des modèles, représentation abstraite d'un problème et de sa solution, permettant de décrire de manière précise et non ambiguë l'architecture du système.

Dans de nombreux projets, l'expression textuelle, est la technique principale pour décrire l'ensemble des éléments. Cette méthode est cependant très limitée puisqu'elle ne favorise pas les relations inter-domaines (où les langages sont différents), et la qualité de la description est entièrement dépendante du savoir-faire du concepteur. L'alternative à cette démarche est l'utilisation de langages particuliers, et idéalement de langages formels. Pour mener une analyse, différents critères de sélection de ces langages peuvent être utilisés. Il est avant tout important de se baser sur un langage existant, ou familier du fabricant. La facilité de compréhension et d'application du langage permet de créer rapidement des documents et de mieux communiquer les informations. Il existe de nombreux langages qui permettent aujourd'hui de modéliser la conception d'un système. Mais dans le cadre d'une étude sur le risque, il convient d'intégrer toutes les composantes du système dans son environnement, et d'adopter une vue plus globale. On parle aujourd'hui de vue « système » (par opposition à système informatique ou électronique, etc.). Un des aspects importants, est la représentation graphique qui facilite la lecture et la compréhension rapide et qui permet de communiquer avec des personnes non spécialistes. Pour ces différentes raisons, le langage UML (*Unified Modeling Language*), semble convenir à l'expression d'un modèle en vue d'une gestion du risque. De plus ce langage est devenu un standard dans le domaine des technologies et de nombreux industriels l'utilisent.

Une étude de conception d'un contrôleur de robot basée sur UML, a été effectuée au sein du laboratoire lors d'une thèse précédente (Carroll, 1999), et au regard de la complexité et des particularités du contrôleur de robot, la conclusion est que ce langage a permis de modéliser l'ensemble du contrôleur. De plus, une étude de comparaison (Caroll et al., 1998) a permis de montrer l'apport de la conception orientée objet par rapport à une approche fonctionnelle.

Avant d'aborder la problématique de l'analyse du risque couplée avec UML au chapitre 4, nous présenterons dans la section 3.2 une synthèse des concepts importants de ce langage et des notions que nous utiliserons par la suite. Puis, la section 3.3 présente une rapide description de quelques travaux sur les interactions entre la modélisation orientée objet et la sûreté de fonctionnement de systèmes à sécurité critique, et permet de positionner nos travaux dans ce cadre de recherche.

3.2 UML : notation pour le développement orienté objet

Bien avant l'apparition des techniques permettant de modéliser des systèmes entiers incluant des acteurs humains, des systèmes externes, et des composants informatiques, mécaniques, ou électroniques, les informaticiens avaient travaillé sur des techniques de modélisation permettant de décrire et de réaliser les logiciels. Ainsi, ce que l'on appelle aujourd'hui la théorie des systèmes doit beaucoup au développement de l'informatique, et c'est pour cette raison que de nombreux concepts présentés ici font référence à ce domaine.

3.2.1 Le paradigme objet

3.2.1.1 La complexité des systèmes

La complexité grandissante des systèmes informatiques et l'explosion de leurs domaines d'applications, ont rendu le développement lent, chaotique, induisant des risques de défaillances dangereuses ou très coûteuses. La maintenance de tels systèmes s'est alors révélée extrêmement complexe. La situation fut décrite à la fin des années 70 comme la « crise du logiciel ». Pour pallier cette complexité, une solution fût de développer des vues simples des systèmes, en cachant certaines informations. De plus la technique du « diviser pour mieux régner » fut appliquée, et dans cette logique deux approches sont aujourd'hui possibles. L'une concerne la vision du système comme un ensemble de fonctions et de sous-fonctions permettant de délivrer le service voulu, l'autre s'appuie sur une structure composée d'entités collaborant entre elles afin de fournir le service attendu, on parle alors de développement orienté objet.

Cette deuxième approche de la programmation est apparue après la programmation fonctionnelle, mais a intégré beaucoup de techniques validées au fur et à mesure des expériences de la programmation fonctionnelle. Les principales caractéristiques de la programmation objet sont l'encapsulation, l'héritage et le polymorphisme. Ces trois techniques, largement décrites dans les ouvrages sur l'objet (Jacobson, 1992; Booch, 1994; Rumbaugh et al., 1991),

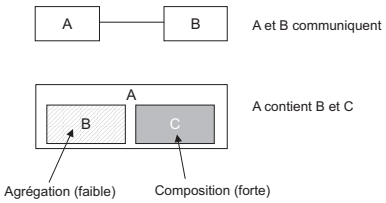

FIG. 3.1 – Interactions entre les objets

permettent d'apporter à un système une vision plus claire, une séparation entre entités plus importante, et ont amené les développeurs à introduire les notions de réutilisabilité, modifiabilité et traçabilité. Le débat entre programmation fonctionnelle et programmation objet n'est toujours pas terminé aujourd'hui. Des systèmes très complexes, validés expérimentalement utilisent encore une approche fonctionnelle et il est difficile pour les industriels de faire la transition vers l'objet. Beaucoup sont encore sceptiques quant à l'utilisation de l'objet dans certains domaines comme les systèmes à sécurité critique[1].

3.2.1.2 La notion d'objet

Les objets du monde informatique sont des entités modélisant des entités physiques du monde réel (un capteur, un moteur, etc.) et des entités conceptuelles (un générateur de trajectoire, un contrôleur d'axe, etc.). Ces objets donnent une représentation simplifiée du monde réel, mais en communiquant entre eux, ils réalisent le service demandé par l'utilisateur. Chaque objet possède une identité unique, une partie visible (publique) pour les autres objets et une partie cachée (privée). On représente les objets comme des rectangles, communiquant par des liens de communication comme représenté en figure 3.1. Sur cette figure apparaît une deuxième relation entre objets qui est la contenance. Nous verrons ultérieurement comment UML représente ces relations.

3.2.1.3 Les classes d'objet

Des objets distincts ayant une structure et un comportement commun, peuvent être regroupés en *classes* d'objets. Cela permet de spécifier pour un ensemble d'objets, comme par exemple plusieurs capteurs de position, les aspects statiques et dynamiques, puis de décrire le nombre et l'identité des objets qui vont être instanciés à partir de cette classe. La figure 3.2 illustre le cas où l'on instanciera deux objets capteurs de position, *CapteurPosition1* et *CapteurPosition2*, à partir de la classe *CapteurPosition*.

[1]Voir les discussions sur le *Safety Critical Forum* à l'adresse web : http ://www.cs.york.ac.uk/hise/hise4/frames11.html

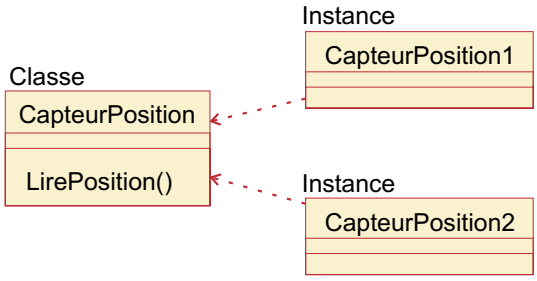

FIG. 3.2 – Exemple de deux instances de la classe CapteurPosition

3.2.2 Le langage de modélisation UML

3.2.2.1 Pourquoi modéliser ?

De la même manière qu'à l'origine de la programmation orientée objet, le besoin d'accéder plus facilement aux informations de la conception a conduit à la modélisation. Modéliser un système informatique sert à donner des angles de vue différents à la manière d'un plan d'architecte. Par cet ensemble de vues, le modèle permet de mieux comprendre le problème et sa solution. Les différents intervenants de la conception peuvent alors communiquer sur la base d'une documentation avant que la fabrication (ou l'intégration) ait eu lieu. Cela permet notamment d'exprimer les résultats de chaque étape du processus de développement et ainsi de mieux détecter les erreurs de développement comme les omissions, les incompréhensions, les inversions, etc. Un des derniers points que l'on peut noter est la génération automatique de code depuis les modèles, ce qui produit un gain de temps et une fiabilité plus importante si les modèles ont été validés. Ce point est encore à l'étude, particulièrement pour la modélisation en UML, qui comporte beaucoup d'imprécisions et de choix laissés à l'utilisateur (ce qui en fait un langage semi-formel et non formel).

D'une manière générale, la modélisation est un des moyens les plus importants pour la prévention des fautes de développements, et aujourd'hui, indépendamment du langage utilisé, tous les projets industriels de grande envergure utilisent cette technique.

3.2.2.2 Bref historique de UML

UML (*Unified Modeling Language*) est un langage pour la représentation des constructions des systèmes complexes. Ce langage a été proposé suite à une requête (RFP, *Request For Proposal*) lancée par l'OMG (*Object Management Group*) concernant une méthode orientée objet standard. Grady Booch, Jim Rumbaugh et Ivar Jacobson proposèrent ce langage (et non une méthode comme cela était demandé) en unifiant une partie de leurs travaux respectifs et en intégrant d'autres travaux orientés objet ou pouvant s'appliquer à l'objet comme les Statecharts développés par Harel (1987). La première version validée par l'OMG fut la 1.1 en 1997, la version 1.4 est actuellement disponible (OMG, 2001), et les réflexions pour la version 2.0 sont en cours (voir figure 3.3). Dès 1996, UML devint un enjeu stratégique et de

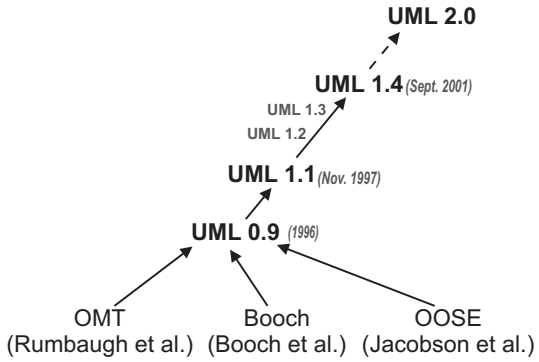

FIG. 3.3 – Évolution des versions du langage UML

nombreux partenaires industriels participèrent à son élaboration. Le succès que connaît aujourd'hui UML, faisant de ce langage un standard industriel, doit beaucoup à la participation des industriels à son développement et à son utilisation, mais prouve aussi que la communauté des informaticiens éprouvait un réel besoin de clarification au sein des différents langages objets qui existaient. Le pari d'UML est aujourd'hui de se stabiliser à une version 2.0, et de résoudre les différentes critiques qui lui ont été faites (voir par exemple les articles de Hitz et Kappel (1998) et Simons et Graham (1999)). Les travaux se portent notamment sur des extensions de ce langage, appelés *profils*, à des domaines spécifiques comme le temps réel, ou les systèmes à sécurité critique.

3.2.2.3 Les diagrammes UML

Le but de cette partie n'est pas de proposer un cours sur la notation UML, largement décrite dans de nombreux ouvrages (Muller, 2000; Booch et al., 1999; Roques et Vallée, 2002), mais de présenter les différents diagrammes qui seront utilisés par la suite. UML possède neuf diagrammes permettant de modéliser les aspects structurels (ou statiques), les aspects dynamiques (changements d'états et réponses aux messages venant des autres objets), et les aspects propres à la représentation des exigences fonctionnelles (diagramme des cas d'utilisation). Le détail de ces diagrammes est fourni en annexe A, nous donnons cependant une courte description pour chacun d'eux :

1. *Les diagrammes des cas d'utilisation* décrivent la façon dont le système sera utilisé. Ils montrent les relations entre les acteurs et les cas d'utilisation du système. Jacobson (1992) définit les acteurs comme les entités interagissant avec le système (les acteurs sont en général des utilisateurs ou des dispositifs extérieurs), et un cas d'utilisation comme une manière spécifique d'utiliser le système.
2. *Les diagrammes de séquence* décrivent une interaction entre plusieurs objets organisée dans le temps. Cette interaction est un ensemble de messages échangés entre les objets pour effectuer une opération ou obtenir un résultat.

3. *Les diagrammes de collaboration* présentent l'interaction organisée autour des objets et de leurs liaisons. À la différence des diagrammes de séquence, ils ne montrent pas le temps comme une dimension séparée. Ces diagrammes, ainsi que les diagrammes de séquence, sont appelés diagrammes d'interaction.
4. *Les diagrammes de classes* expriment de manière générale la structure statique d'un système, en termes de classes et de relations entre ces classes.
5. *Les diagrammes d'objets* sont des instances des diagrammes de classes, utilisés pour présenter un contexte particulier du problème.
6. *Les diagrammes d'états* montrent le comportement des classes. Ils sont basés sur les *statecharts* définis par Harel (1987). Ce sont des automates hiérarchiques, permettant une certaine représentation du parallélisme.
7. *Les diagrammes d'activités* sont des variantes des diagrammes d'états, organisés par rapport aux actions (ou opérations). Ils sont utilisés pour représenter le comportement interne d'une méthode ou d'un cas d'utilisation.
8. *Les diagrammes de composants* sont utilisés pour modéliser les différents composants du système et leurs relations. Ces composants peuvent être des modules (ou unités de compilation ou fichiers), des sous-programmes (ou entités fonctionnelles), des tâches (ou modules destinés à l'implémentation des processus) et des sous-systèmes (ou regroupement de modules, de sous-programmes, de tâches et/ou d'autres sous-systèmes).
9. *Les diagrammes de déploiement* montrent l'organisation des composants matériels et le rattachement du logiciel aux dispositifs matériels (ou *nœuds*).

La particularité de ce langage est la possibilité d'utiliser les diagrammes pour les concepteurs de la manière dont ils le désirent. Si la sémantique est normalisée par l'OMG, l'utilisation et la modification n'en reste pas moins ouverte. Cette caractéristique, et la variété des diagrammes a conduit à des utilisations très variées d'UML, et dans de nombreux domaines, allant de la modélisation de systèmes organisationnels à la modélisation de systèmes temps réel. Bien que ce langage ait été créé à la base pour des systèmes informatiques, son évolution spectaculaire fait de lui un langage *système* au sens le plus large du terme.

3.2.2.4 Le *métamodèle*

Un métamodèle est un modèle qui décrit d'autres modèles. Ainsi, les différents concepts du langage UML ont été eux-mêmes modélisés avec UML comme le montre la figure 3.4. Cette modélisation récursive, qui constitue le métamodèle, décrit de manière semi-formelle les éléments de modélisation ainsi que la syntaxe et la sémantique de la notation qui permet de les manipuler. Le métamodèle devient, entre autres, l'outil vérificateur qui facilite l'identification des incohérences, notamment par l'utilisation de règles de bonne modélisation[2], exprimées dans la version 1.4 en langage OCL (*Object Constraint Language*). Dans les versions à venir

[2]qui peut être une traduction de *well-formedness rules*

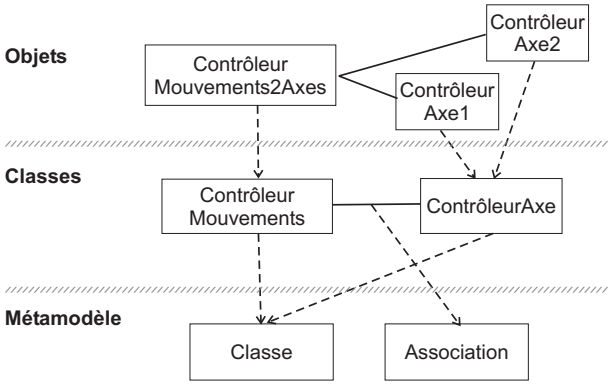

FIG. 3.4 – Objets, classes et métamodèle

d'UML, la spécification d'UML est séparée à ce jour en deux documents complémentaires. Tout d'abord l'*infrastructure* (OMG, 2003b) définit les concepts fondamentaux du langage même et se place en tant que métamodèle ou « méta-métamodèle »[3]. Puis la *superstructure* (OMG, 2003a) se place à un niveau utilisateur en se basant sur les concepts de l'*infrastructure*.

3.2.3 Processus de développement et UML

Bien que la demande lancée par l'OMG concernait une méthode de développement orienté objet, UML n'a été introduit qu'en tant que langage. Il n'existe pas aujourd'hui de méthode de développement standard propre à UML, et l'on peut trouver suivant les domaines d'applications des méthodes différentes (et sans doute autant que de développeurs !). Ainsi, malgré un effort de standardisation autour du Processus Unifié (Jacobson et al., 1999), commercialisé par la société Rational[4] sous le nom de RUP (Rational Unified Process, présenté par Kruchten (1999)), on peut trouver des méthodes comme COMET de Gomaa (2000) ou ROPES de Douglass (1999), adaptées aux systèmes temps réel. Bien que ces méthodes aient une approche différente de l'enchaînement des activités du processus de développement, et utilisent de manière différente les diagrammes UML, il existe néanmoins des règles communes que l'on retrouve dans ces méthodes. Selon les auteurs de UML, le processus utilisant cette notation doit être :

- itératif et incrémental,
- centré sur l'architecture,
- piloté par les cas d'utilisation.

Comme pour la création d'UML, ces principes dérivent de nombreux travaux sur le domaine, et le concept est de les réunir pour se diriger vers un standard.

[3] Il est possible de remonter de niveau conceptuel un grand nombre de fois, et pour simplifier le terme de métamodèle suffit

[4] http ://www.rational.com

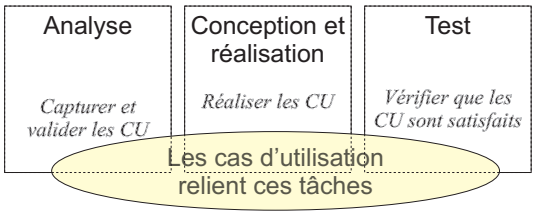

FIG. 3.5 – Développement piloté par les cas d'utilisation (CU)

3.2.3.1 Piloté par les cas d'utilisation

Les cas d'utilisation, développés par Jacobson (1992), permettent d'identifier et de modéliser les besoins fonctionnels lors de l'analyse. Dans un développement utilisant UML, ils permettent de relier les différentes activités comme le montre la figure 3.5. Le fait de s'appuyer sur ces cas d'utilisation, impose aux concepteurs pour chaque étape du développement de faire le lien avec les exigences. Et en effet, depuis l'analyse jusqu'aux tests, chaque équipe de développement, ou chaque concepteur, peut réaliser ou tester un ou plusieurs cas d'utilisation.

3.2.3.2 Itératif et incrémental

Un processus itératif consiste à intégrer le fait que les exigences évoluent au cours d'un projet, en développant des prototypes. La plupart des méthodes s'appuient sur le modèle en spirale de Boehm (1988) qui consiste à réaliser des prototypes en enrichissant chaque version de nouvelles exigences. La représentation de ce modèle de processus sous forme de spirale, permet de définir des cadrans d'activités communs à chaque itération. Par exemple, pour chaque prototype, il est possible de définir des étapes d'analyse, conception, implémentation et de test. Les premières itérations permettent d'expérimenter et de valider des technologies, de planifier la suite du développement, et surtout de définir le noyau d'architecture du système. Chaque itération permet de réaliser une partie des fonctionnalités du système. On parle alors de processus incrémental.

Le point important concerne alors le choix des exigences que l'on souhaite développer, et par extension à UML, le choix des cas d'utilisation à réaliser. Il convient donc d'établir des priorités entre cas d'utilisation. Certains lient cette problématique à la notion de *risque* (Gomaa, 2000; Douglass, 1999) et utilisent le terme d'*analyse des risques* pour classer les cas d'utilisation. L'emploi de ce terme est ici relatif aux dommages liés à la gestion de projet, c'est-à-dire les performances du système réalisé, les délais, les coûts, etc. Cette analyse des risques est donc en fait une évaluation des différentes contraintes qui pèsent sur le concepteur, et qui l'amèneront à choisir les cas d'utilisation qu'il réalisera en premier lieu. L'étude que nous réalisons dans ce mémoire, ne concerne pas ce type de risques, puisque les seuls dommages que nous traiterons concernent la sécurité des biens et des personnes. Cependant la terminologie et les concepts fondamentaux sont communs à ces différents domaines.

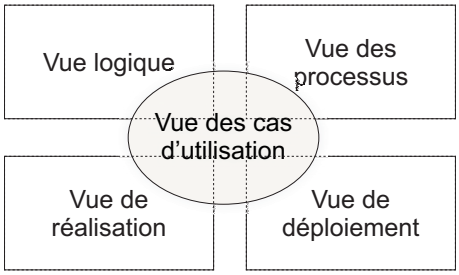

FIG. 3.6 – Le modèle d'architecture à 4+1 vues, figure tirée de Kruchten (1999)

3.2.3.3 Centré sur l'architecture

La recherche de la forme générale du système doit être faite dès le début du développement et va ensuite être complétée tout au long des différentes étapes. Une approche systématique consiste à rechercher une architecture :

- adaptée aux changements,
- pour et avec la réutilisation,
- compréhensible intuitivement,
- satisfaisant les cas d'utilisation.

La notion d'architecture est très diffuse et parfois mal appréhendée. Elle consiste néanmoins, pour un système logiciel, à définir la façon dont le système est construit. Ainsi elle est souvent représentée par des packages et des sous-systèmes interagissant entre eux pour fournir le service demandé. Cependant la vision de l'architecture ne peut se réduire à ces représentations. Par analogie, un architecte peut s'il le désire, consulter les plans d'une maison suivant différentes vues, ayant chacune une granularité (niveau de détail) différente. De la même manière, le système logiciel peut être représenté suivant cinq vues représentées sur la figure 3.6 :

- *La vue logique* décrit les aspects statiques et dynamiques du système en termes de classes et d'objets. Elle se concentre sur l'abstraction et l'encapsulation et met en jeu des objets et des classes, ainsi que des collaborations et des interactions autour de ces objets. Cette vue est composée de diagrammes de classes, d'objets, de séquence, de collaboration et d'états.
- *La vue de réalisation* se préoccupe de l'organisation des modules dans l'environnement de développement. Elle montre l'allocation des classes dans les modules, et l'allocation des modules dans les sous-systèmes. Les sous-systèmes sont eux-mêmes organisés en niveaux hiérarchiques comportant des interfaces bien définies. Cette vue traite les modules, les sous-programmes, les sous-systèmes et les tâches, dans des diagrammes de composants.
- *La vue des processus* représente la décomposition en flots d'exécution (processus, fils d'exécution et tâches), la synchronisation entre flots et l'allocation des objets et des classes au sein des différents flots. Elle prend toute son importance dans des environne-

ments multitâches. Elle fait apparaître les tâches, les fils d'exécution, les processus et les interactions, dans des diagrammes de séquence, de collaboration et d'états.

– *La vue de déploiement* décrit les ressources matérielles et l'implantation du logiciel dans ces ressources. Elle prend toute son importance lorsque le système est distribué. Cette vue se concentre sur les nœuds, les modules et les programmes principaux, dans des diagrammes de déploiement.
– *La vue des cas d'utilisation* unifie les quatre vues précédentes. Les cas d'utilisation permettent d'identifier les interfaces critiques, et forcent les concepteurs à se concentrer sur les exigences fonctionnelles. Elle rend compte des acteurs, des cas d'utilisation, des classes et des collaborations à l'aide de diagrammes des cas d'utilisation et de diagrammes de séquence ou de collaboration.

Il est important de noter que ce concept n'est pas une proposition de méthode, et que ces vues sont complétées tout au long du processus itératif et incrémental même si certaines de ces vues sont plus en rapport avec certaines étapes du processus de développement.

Bien que cette vision ait été proposée dès la création d'UML, il est cependant possible de trouver d'autres représentations. Gomaa (2000) propose notamment de faire la distinction entre la représentation du comportement dynamique (vies possibles des objets) et celle du comportement interactif (flots de contrôle entre les objets). Douglass (1999) utilise pour décrire les vues de la conception les notions de *conception architecturale* (structures physiques, tâches, composants, etc.), de *conception mécanistique*[5] (ensemble des classes d'objets collaborant), et de *conception détaillée* (chaque classe est raffinée en terme d'attributs et d'opérations). La conclusion que l'on peut tirer de ces différentes approches est que UML, "pouvant s'appliquer à n'importe quelle méthode" selon ses auteurs (Booch et al., 1999), impose tout de même par l'ensemble de ses diagrammes un cadre d'exploration du système et de son architecture qui tend à se standardiser.

3.3 Travaux actuels sur le développement orienté objet et la sûreté de fonctionnement

Bien que largement étudiée dans le domaine de la recherche, l'intégration du développement orienté objet dans des projets industriels de systèmes à sécurité critique est encore à l'état d'évaluation. Afin d'effectuer un état de l'art des études sur les rapports entre l'orienté objet et la sécurité, nous avons considéré globalement la sûreté de fonctionnement (la sécurité étant un des attributs de la sûreté de fonctionnement). Il est en effet beaucoup plus rare de trouver des études sur les liens entre le risque et ce langage. Il existe cependant des travaux comme ceux de Tah et Carr (2001) et de Gabbar et al. (2001), où les auteurs utilisent la notation UML pour modéliser le processus de gestion du risque (acteurs et activités de la gestion

[5]Traduction de *Mechanistic Design*

du risque), de la même manière que cela était fait pour le processus de développement avec UP (*Unified Process*) (Jacobson et al., 1999)) et plus récemment SPEM (*Software Process Engineering Metamodel*[6]). Il est important de noter que ni UP ni SPEM incluent les notions relatives au risque. Cependant les travaux actuels concernant la modélisation de l'activité de gestion du risque au sein du processus de développement se placent à un niveau d'abstraction très élevé et ont pour vocation d'aider les concepteurs à s'organiser au sein de l'entreprise. Cela ne concerne donc pas l'activité de gestion du risque elle-même, ce qui nous intéresse ici. De plus les particularités du domaine de la robotique de service et la nouveauté du domaine font qu'il est encore impossible aujourd'hui d'avoir le recul nécessaire pour effectuer une telle analyse. Pour ces raisons, nous nous limiterons donc aux études plus techniques concernant la sûreté de fonctionnement.

En suivant la classification de Laprie (1992), il est possible de classer les moyens de la sûreté de fonctionnement suivant quatre concepts résumés ci-dessous :

- *la prévention des fautes*, ou comment empêcher l'occurrence ou l'introduction de fautes, qui relève de l'ingénierie système (application de méthodes de développement, utilisation de langages formels ou semi-formels comme UML, etc.) ;
- *la prévision des fautes* ou comment estimer la présence, la création et les conséquences des fautes des composants du système, en utilisant des techniques d'analyse quantitatives et qualitatives. C'est principalement cette activité qui est couverte pas l'analyse du risque ;
- *l'élimination des fautes*, ou comment réduire le nombre et la gravité des fautes, par exemple par des techniques de vérification (test fonctionnel, structurel, etc.) ;
- et *la tolérance aux fautes*, ou comment fournir un service à même de remplir la fonction du système en dépit de fautes, qui concerne par exemple le traitement d'erreur par le système (détection, diagnostic et recouvrement).

Cette terminologie permet d'effectuer une classification des aspects liés à la sûreté de fonctionnement. Cependant les analyses couvrent en général simultanément plusieurs de ces aspects. Par exemple, lorsque l'on effectue une analyse du risque comme présentée au chapitre 1, une partie importante concerne la prévision des fautes par l'utilisation de techniques comme l'AMDEC, ou les arbres de fautes. Le fait de baser cette analyse sur une modélisation semi-formelle en UML contribue à la prévention des fautes. Puis, les résultats de l'analyse du risque peut mener à des choix de mécanismes de tolérance aux fautes comme par exemple la redondance, qu'il conviendra de tester et de valider en utilisant des techniques d'élimination des fautes. Ceci montre bien que cette séparation entre les différents moyens est d'un niveau d'abstraction élevé et que, dans la pratique, les études couvrent plusieurs aspects. Il est important de noter que le but de cette section n'est pas d'effectuer un état de l'art exhaustif sur la sûreté de fonctionnement en général et l'orienté objet, mais de se concentrer sur les aspects

[6] http ://www.omg.org/technology/documents/formal/spem.htm

concernant (ou pouvant concerner) la sécurité et l'utilisation de langages de modélisation orientés objet.

3.3.1 Prévention des fautes

L'utilisation de langages de modélisation, contribue à la prévention des fautes. En effet, la qualité du formalisme utilisé aide le développeur à réaliser des modèles clairs et non ambigus. Ainsi, en UML, la représentation des modèles dynamiques repose sur de nombreuses informations présentes dans les modèles statiques (nom des méthodes, des attributs, des classes, etc.), et il est alors aisé d'utiliser des outils simples de vérification de la cohérence de ces informations communes. De plus, un langage de modélisation comme UML contribue à la tracabilité qui est une des exigences fondamentales de la certification (notamment pour les systèmes à sécurité critique). En effet, l'utilisation de cette notation et des mêmes diagrammes pour les différentes étapes du processus de développement, permet de suivre le cycle de vie de chaque exigence ou de chaque composant.

L'utilisation d'UML en tant que langage semi-formel contribue à la prévention des fautes, mais comporte certaines limites et notamment au niveau de l'expression de certaines contraintes. Ainsi Bitsch (2002) propose d'utiliser UML pour les exigences fonctionnelles et TimeLogic pour les contraintes de temps (exigences non fonctionnelles). De manière similaire, Cichocki et Górski (2000) utilisent l'orienté objet puis Z pour exprimer les contraintes de sécurité d'un système. Bondavalli et al. (2001) s'appuient sur des extensions d'UML (les *tagged values*, les *stéréotypes*, et les *contraintes* définies dans la norme de l'OMG (2001)) permettant de prendre en compte un certain nombre d'exigences liées à la sûreté de fonctionnement. L'utilisation de ces techniques combinées, permet en premier lieu de représenter de façon formelle ou semi-formelle les contraintes non-fonctionnelles, puis de les dériver pour la conception ou la validation des modèles (ce dernier point concerne l'élimination des fautes présentée dans la section 3.3.4).

La modélisation des exigences de sûreté de fonctionnement n'était pas traitée dans les premières versions d'UML. Les exigences étaient décrites de manière textuelle et généralement en accompagnement des cas d'utilisation. Cependant il existe aujourd'hui plusieurs travaux concernant la représentation de ces contraintes. La notion de QoS, *Quality of Service*, semble acceptée aujourd'hui dans de nombreux domaines pour regrouper des caractéristiques d'un service, telles que les performances et les attributs de la sûreté de fonctionnement (fiabilité, sécurité, etc.). Dans une réponse à un appel de l'OMG sur les problèmes temps réel, Selic et al. (2000) ont présenté plusieurs moyens de modélisation des contraintes de type temps réel en utilisant la notion de QoS, ce qui a donné lieu aujourd'hui à un profil UML (OMG, 2002b). De la même manière Douglass (1999) intègre ces notations pour exprimer des contraintes de performances, de sécurité et de fiabilité. La notion de QoS, principalement utilisée à l'origine pour les systèmes distribués, a fait l'objet d'un RFP de l'OMG (2002a), pour une application au langage UML.

L'aspect prévention des fautes concerne aussi la manière dont le langage UML sera utilisé. Il existe en effet un certain nombre de travaux abordant cet aspect, qui pourrait correspondre à une analyse des risques liés à l'utilisation d'UML. Plus généralement, l'approche objet est analysée en terme de certification, et certaines caractéristiques de l'objet jugées dangereuses sont contraintes ou interdites. Rierson (1999) a rédigé un rapport pour la FAA (Federal Aviation Administration) où sont recensés les problèmes engendrés par les concepts objets. À partir de ce travail, l'AVSI (Aerospace Vehicule Systems Institute) a rédigé un guide (AVSI, 2002) sur l'utilisation de l'orienté objet en vue d'une nouvelle version de la norme avionique DO178B/ED-12 Revision B (1992).

Si certaines constructions sont dangereuses, d'autres en revanche ont été validées sur de nombreuses applications. Ces constructions génériques ont alors été assimilées à des patrons (*patterns* en anglais) pouvant être utilisés pour d'autres applications. Les patrons sont des solutions à des problèmes précis dans des contextes bien définis. Manhes (1998) présente une synthèse des différentes traductions que l'on peut trouver pour cette notion. Ces patrons sont en général présentés sous forme de catalogues. Il existe un certain nombre de patrons modélisés en UML dans l'ouvrage de Gamma et al. (1995). Douglass (1998) propose un série de patrons propres aux applications temps réel à sécurité critique les plus fréquents. Dans un rapport présentant une réponse pour un RFI (Request For Information) lancé par l'OMG, Daugherty et al. (2002) proposent un ensemble de patrons de spécification et de conception permettant de justifier de la qualité d'un logiciel en vue de la certification. Les catalogues de patrons présentés dans les références précédentes concernent plus généralement l'analyse ou la conception. Cependant, il est possible de trouver des "règles" de programmation pour les langages objet, comme le présente Binkley (1995) pour le langage de programmation C++, qui constituent des patrons d'implantation.

3.3.2 Prévision des fautes

Cette activité doit permettre d'anticiper les fautes par des techniques d'identification, et d'évaluation de leur effets. De nombreux travaux ont été effectués sur la notion de composant, qui est proche de celle d'objet, en considérant que ces composants ont tous des entrées et des sorties (par analogie avec les composants électroniques). Yacoub et al. (2000) identifient les composants critiques d'un système logiciel en calculant pour chaque composant un facteur de risque évalué en fonction de la complexité (avec des métriques définis) et de la gravité d'une défaillance de ce composant. Cela permet ensuite de se concentrer sur certains composants jugés critiques au détriment d'autres moins important pour la sécurité. De la même manière Papadopoulos et Maruhn (2001), se basent sur des modèles de composants et utilisent la technique d'analyse HAZOP (Hazard Operability) pour générer automatiquement par la suite des arbres de fautes. Ces aspects rejoignent la notion de SIL (Safety Integrity Level), utilisée dans les normes IEC 61508 (2001), mais sont difficilement applicables à un système modélisé en

UML. En effet la notion d'objet (qui peut devenir par la suite un composant) se différencie trop de celle de composant (Booch et al., 1999) pour utiliser efficacement une démarche similaire.

Plus proche de l'objet et d'UML, Górski et Nowicki (1997) définissent des « attributs critiques » d'objets du système, et se concentrent sur les effets des variations néfastes de ces attributs. Ces travaux ont notamment mené les auteurs aux notions de sous-systèmes critiques (un sous-système est très proche de la notion de composant) et à l'identification de comportements dangereux sur la base des diagrammes d'états (Nowicki et Górski, 1998). La démarche est identique aux précédentes, et consiste donc à identifier des parties du système susceptibles de créer des dommages. Cependant les rapports avec les objets mêmes du système sont encore difficiles à évaluer.

Parmi les méthodes analytiques permettant la prévision des fautes, l'AMDEC (cf. chapitre 1) est sans doute une des plus utilisées lors des analyses fonctionnelles. Elle peut être néanmoins appliquée à des composants logiciels et à leurs liens comme le font Yacoub et al. (2000). Cette approche est similaire à l'étude des composants électroniques et ne prend pas en compte les concepts importants de l'approche objet. Bitsch (2002) propose d'utiliser cette technique en analysant les méthodes des objets du système à la manière d'une analyse de fonctions, et identifie ainsi les répercussions sur le système. Dans un cas d'étude sur la conception d'une voiture, Johannessen et al. (2001) utilisent les cas d'utilisation d'UML pour exprimer les besoins et effectuent une AMDEC sur ces diagrammes. Le lien avec UML est cependant réduit car les cas d'utilisation identifiés dans cet article, comme « Stabilité lors du freinage », correspondent à des besoins non fonctionnels et ne peuvent donc pas être dérivés vers des objets.

Parmi les autres techniques d'analyse, il convient de citer l'analyse des arbres de fautes, qui est en général couplée avec une AMDEC. Górski et al. (1996) montrent comment ils se basent sur les arbres de fautes pour exprimer des classes de fautes d'un modèle. Le rapport avec les objets du système reste complexe, et l'étude des arbres semble être faite en parallèle sans véritable interaction avec les modèles UML. Pai et Dugan (2002) présentent une méthode et des algorithmes permettant de générer automatiquement des arbres de fautes à partir de modèles UML. Les arbres de fautes qui sont générés par cette méthode sont en fait des modèles de fiabilité, qui permettent de décrire en parallèle des modèles UML, comment une défaillance peut survenir. Bondavalli et al. (2001) ont construit de la même manière des modèles de fiabilité (IM pour *Intermediate Model*) basés sur les diagrammes UML, dérivés ensuite en *Timed Petri Nets* (Bondavalli et al., 1999b; Majzik et Bondavalli, 2000; Bondavalli et al., 1999a). Ces techniques permettent avant tout de fournir des outils aux ingénieurs souhaitant utiliser UML, en palliant certains manques d'UML comme l'exécution des modèles. Cependant, les techniques que l'on vient de présenter s'appuient fortement sur la notion de composant qui est différente de celle d'objet ou de classe. De plus, elles requièrent la connaissance d'autres langages ou techniques en complément à UML ce qui complique le développement.

3.3.3 Tolérance aux fautes

Les études liées à la tolérance aux fautes concernent principalement des analyses d'architectures tolérantes aux fautes, ou des mécanismes intégrés au système, permettant de réaliser les activités de détection, de diagnostic et de recouvrement d'erreur. L'utilisation d'UML permet alors de modéliser à la fois les aspects purement fonctionnels et les aspects concernant la tolérance aux fautes (Ferreira et Rubira, 1998). Beaucoup de travaux concernent cependant la fiabilité et les systèmes distribués (Selic, 2000), et plus rarement la sécurité dans des systèmes temps réel. De plus, cet aspect de la sûreté de fonctionnement concerne plus fréquemment la conception que l'analyse. C'est pour ces raisons que les aspects de la tolérance aux fautes que nous aborderons par la suite au sein de l'analyse du risque, ne concerneront que l'utilisation de patrons de sous-systèmes de tolérance aux fautes modélisables en UML.

3.3.4 Élimination des fautes

Cet aspect que l'on peut aussi qualifier de validation ou vérification de la sûreté de fonctionnement (Geffroy et Motet, 1998, Chap.9) regroupe principalement des techniques de vérification et de test. Ces vérifications peuvent être effectuées à plusieurs niveaux du processus de développement. Au niveau des spécifications, il convient de valider la *complétude* du modèle, qui exprime le fait qu'il existe une réponse du système pour toutes les séquences d'entrées possibles, et sa *consistance* reflétant l'absence d'exigences conflictuelles ou de non déterminisme. Ces aspects sont fréquemment étudiés en utilisant des diagrammes d'états lors de l'analyse (Pap et al., 2001). Pendant la conception, ce sont encore les diagrammes d'états qui sont le plus étudiés, mais les solutions consistent en général à transformer ces diagrammes UML en modèles exprimés avec d'autres langages formels (Bitsch, 2002; Traon et al., 2000; Chevalley et Thévenod-Fosse, 2001). Ceci est principalement dû au fait qu'il existe des logiciels de validation des modèles très performants s'appuyant sur d'autres langages formels que UML, alors que pour UML certains logiciels équivalents sont encore à l'état de développement.

Au sein d'un processus de développement, l'élimination des fautes fait partie intégrante de nombreuses méthodes, comme par exemple le cycle en V conseillé dans la norme IEC 61508 (2001), et est fortement lié au processus de gestion du risque (validation des exigences non fonctionnelles de sécurité par exemple). Cependant, pour notre étude nous n'aborderons pas ce point considérant que les techniques d'élimination des fautes sont utilisées après les principales étapes d'une analyse du risque sans qu'il y ait d'interaction entre ces activités.

3.3.5 Conclusions sur la sûreté de fonctionnement et le développement orienté objet

Cette section a montré qu'il existait pour chacun des moyens de la sûreté de fonctionnement (prévention, prévision, élimination et tolérance aux fautes) des interactions avec UML. Une grande partie de ces travaux concerne la prévention des fautes, et notamment celles introduites par l'homme. L'utilisation d'un langage de modélisation permet en effet de réduire les erreurs humaines de conception.

Une autre partie de ces travaux concerne les techniques d'élimination des fautes comme le test des modèles qui se concentre sur certaines constructions (diagramme d'états principalement). En raison de l'impossibilité d'exécuter les modèles UML (dans sa version actuelle), de nombreux travaux de recherche s'orientent vers la conversion des diagrammes UML vers d'autres modèles exécutables.

La tolérance aux fautes regroupe principalement des recherches sur des architectures de systèmes distribués, modélisables en UML. Dans le cadre de la gestion du risque, les techniques de tolérance aux fautes sont introduites lors de la maîtrise du risque. Ces techniques sont principalement des solutions technologiques qui ne seront abordées qu'au titre d'exemple dans ce mémoire.

La prévision des fautes concerne des techniques que l'on retrouve au sein des la gestion du risque. En effet, on retrouve dans cette thématique des travaux sur les interactions entre des techniques comme l'AMDEC ou les arbres de fautes avec UML. Au sein de l'analyse du risque, l'identification des dangers inclut la prévision des fautes et s'appuie donc sur les mêmes techniques que celles de la sûreté de fonctionnement. La plupart des travaux s'appuient sur la notion de composant et des premières connexions entre UML et ces techniques ont pu être faites. Sur ce point, nous montrerons dans le chapitre suivant comment notre approche se démarque de ces travaux, notamment par la prise en compte de concepts différents de celui de composant.

3.4 Conclusion

Il est important de noter que UML est un langage et non une méthode et c'est sans doute une des raisons qui a fait son succès. Cela nous permet donc de l'utiliser pour notre démarche basée sur l'analyse du risque et présentée dans le chapitre suivant.

Les différentes règles d'utilisation induites par les concepts objets et propres à ce langage, ainsi que la variété des diagrammes permettant différents points de vue, et les trois principes liés au processus de développement (itératif et incrémental, piloté par les cas d'utilisation, et centré sur l'architecture), en font une technique dite de *prévention des fautes* dans le vocabulaire du domaine de la sûreté de fonctionnement. Ces différents éléments permettent en effet de suivre un développement cohérent, où chaque étape repose sur les diagrammes de l'étape précédente, et permettent de réduire l'occurrence de fautes ou de modélisation erronée.

Les diagrammes des cas d'utilisation permettent notamment de garantir la traçabilité qui est un aspect fondamental de la certification. Il est aujourd'hui admis que l'utilisation des langages objets ont aussi des apports en modifiabilité et réutilisabilité.

De plus, le langage UML est un langage « centré humain », de par la modélisation des besoins par les cas d'utilisation dont c'est la caractéristique principale, mais aussi par la possibilité de représenter les acteurs humains dans la plupart des diagrammes.

Une des caractéristiques d'UML, qui répond à un des besoins de notre démarche exprimé dans la conclusion du premier chapitre, est son aspect *système* de par la complémentarité des diagrammes, dont la sémantique tout en étant formalisée par l'OMG permet des adaptations en fonction de l'application développée. De plus la notation graphique permet de communiquer plus facilement avec tous les acteurs du développement d'un système complexe.

Cependant, ce langage comporte des points négatifs qu'il convient de souligner. Tout d'abord le pouvoir de vérification automatique des modèles est réduit. Ceci a conduit au développement de techniques de conversion de ce langage vers d'autres langages, permettant d'utiliser des outils de vérification formelle. Ce point d'étude, source de nombreuses recherches, ne sera pas abordé dans ce mémoire. Un autre point important identifié dans ce chapitre est l'incompatibilité des techniques de la sûreté de fonctionnement utilisées jusqu'ici pour des analyses fonctionnelles ; le fait de passer en langage objet remet en question ces techniques. Il ressort à travers des études présentées dans la dernière section (3.3), que les rapports entre ces techniques et la modélisation orientée objet sont encore à l'état d'évaluation.

En dernier point, l'utilisation des concepts objet est aussi une technique qui peut être à l'origine d'incompatibilités avec le domaine des systèmes à sécurité critique. Tout d'abord en terme de performance des logiciels, les programmes développés en objet ont tendance à être plus volumineux, dont l'exécution requiert la collaboration de nombreux objets informatiques. De plus, pour une utilisation système, les concepts même de l'objet peuvent créer des difficultés de compréhension, notamment pour des concepteurs utilisant l'analyse fonctionnelle. Cependant, nous considérons que ces problématiques, concernant en fin de compte l'analyse et l'évaluation des risques liés à l'utilisation d'un langage objet pour le développement des systèmes à sécurité critique, sont sujets à d'autres travaux complémentaires à ce mémoire.

Chapitre 4

Intégration d'UML pour l'analyse du risque

4.1 Introduction

La maîtrise de la sécurité d'une application de robotique de service ne peut se réduire à l'utilisation de techniques particulières comme les muscles artificiels présentés au chapitre 2. Le chapitre 1 avait en effet montré qu'une approche de la sécurité plus générale et liée à la notion de risque était possible, et nous avions conclu sur les besoins d'une telle approche pour la robotique de service. Tout d'abord, face à la nécessité de définir un vocabulaire du risque stable et non ambigu, le chapitre 1 a présenté une synthèse des différents concepts présents dans d'autres domaines, et les a adaptés au domaine de la robotique de service (et plus particulièrement à la robotique médicale). Ceci a conduit à l'identification d'activités permettant de gérer le risque, la principale étant l'analyse du risque. Cette analyse, reposant sur la notion de modèle, doit pouvoir être couplée à un langage utilisé pour le processus de développement. Le chapitre 3 a présenté les caractéristiques du langage UML, et a montré comment il répond à différents besoins identifiés dans le chapitre 1. Parmi les points importants, il a été souligné la possibilité d'utiliser cette notation au sein des différentes étapes du développement, et ceci pour de nombreux domaines. Cet aspect pluridisciplinaire fait d'UML un langage adapté à la modélisation système. De plus, grâce aux cas d'utilisation, la prise en compte de l'humain est au centre de nombreux modèles. Enfin, l'utilisation de ce langage contribue à améliorer la traçabilité des choix de conception, essentielle pour la sécurité et la certification.

La démarche présentée dans ce chapitre valide l'utilisation d'UML pour l'analyse du risque d'un système de robotique de service. Les sections de ce chapitre suivent les étapes de l'analyse du risque introduites au chapitre 1. Une vue générale de ces étapes est donnée figure 4.1. Une première section (4.2) traite de la définition du système et de son utilisation, où sont présentées les étapes de définition du métier, d'intégration du robot dans la tâche de service, de définition des frontières du système et de description des tâches des acteurs. Ces activités permettront de souligner l'importance des cas d'utilisation et des concepts liés

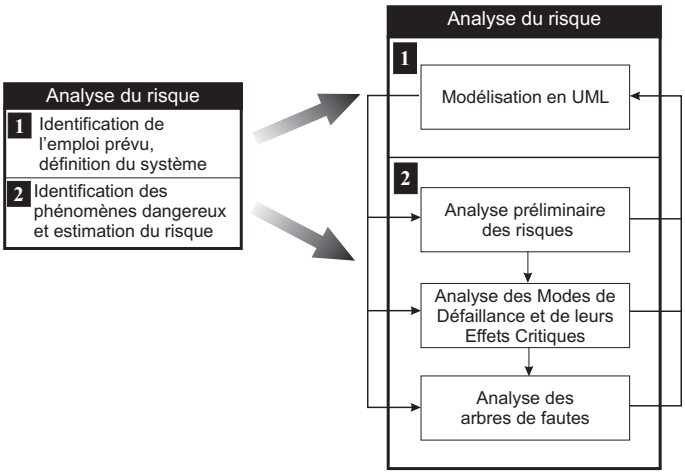

FIG. 4.1 – Vue globale de la démarche présentée dans ce chapitre

aux facteurs humains. Ceci conduira à la réalisation de modèles, qui seront exploités dans une deuxième section (4.3) sur l'identification des dangers et l'estimation du risque. Les techniques d'analyse que sont l'APR (Analyse Préliminaire des Risques), l'AMDEC (Analyse des Modes de Défaillance et de leurs Effets Critiques) et les arbres de fautes seront utilisées lors de cette étape. Une partie de cette analyse, concernant l'utilisation de l'AMDEC, montrera comment la notion de *Message* en UML permet de définir des modèles d'erreur applicables aux différents domaines que sont la mécanique, l'électronique, l'informatique et les erreurs humaines.

Tout au long de la présentation de cette démarche, l'accent est mis sur l'importance des analyses transversales. Comme cela est souligné précédemment, il est fondamental pour la sécurité, d'effectuer à un moment donné une analyse permettant de regrouper les diverses informations fournies par les spécialistes de chaque domaine. C'est à ce niveau que se place cette démarche « système ».

4.2 Description du système et de son utilisation

La partie 1.4.3.2 indique que toute analyse du risque s'appuie sur une description du système et de son utilisation prévue. Certains éléments comme les flux d'énergie, les frontières du système et les interfaces doivent être correctement définis pour pouvoir effectuer l'analyse du risque. Ces différents points peuvent se retrouver dans la vue des cas d'utilisation (cf. section 3.2.3.3) qui exprime la manière dont les acteurs veulent utiliser le système. Il est en effet possible, grâce au diagramme des cas d'utilisation, de modéliser les exigences fonctionnelles. Nous présenterons notamment ici une manière de rajouter les exigences non fonctionnelles liées à la sécurité. De plus, UML est un langage dit « orienté utilisateur », en raison notamment de la modélisation des utilisateurs au sein de la plupart des diagrammes.

Ceci est fondamental pour un système à sécurité critique, où, à tout instant du développement, le concepteur doit garder à l'esprit l'exigence principale : la sécurité des acteurs humains.

4.2.1 Modélisation du métier

Il existe des tâches impossibles à réaliser sans l'aide d'un robot, comme par exemple l'extraction de minerai sur Mars, cependant si l'on se place dans un cadre humain (pas sur Mars), la plupart des applications de robotique de service reproduisent des gestes que l'homme était amené à faire auparavant. Dans la sémantique d'UML, le terme de *métier* est utilisé pour désigner entre autres[1] une tâche ou un ensemble de tâches, spécifiques à un domaine, et généralement réalisées par plusieurs acteurs spécialisés. Ainsi, une opération chirurgicale de la hanche est un *métier*, comme l'est l'assistance à des personnes handicapées. Bien avant que ces métiers incluent l'utilisation de robots, les humains ont développés des organisations pour la réalisation de ces tâches.

En vue de l'inclusion du robot pour l'exécution de ces tâches, nous proposons d'utiliser UML et une extension de ce langage, la modélisation métier (ou *business modeling* présenté par Eriksson et Penker (2000)), permettant de décrire les tâches propres à un métier. Cela permet de mieux le comprendre, et de faciliter la communication, surtout lorsque des spécialistes, comme par exemple les médecins, et les ingénieurs en charge du développement devront collaborer.

La modélisation d'un métier peut être faite avec un diagramme de cas d'utilisation, comme cela sera présenté dans le chapitre 5. Nous ne donnons pas ici d'exemple complet car cela demande une description complète de l'organisation d'un métier. Cependant, les cas d'utilisation métier correspondent en général aux différentes tâches qui incombent aux acteurs comme par exemple dans le cadre médical : gérer les réglages d'un dispositif médical, réaliser un geste médico-chirugical, établir un diagnostic, gérer les instruments chirurgicaux, etc. À ces cas d'utilisation sont reliés les différents acteurs du métier (docteur, assistant, et patient), et il est possible de représenter les interactions entre ces acteurs avec des diagrammes de classes ou de séquence. Ainsi, Jannin et al. (2001) utilisent un diagramme de classes pour représenter les différents objets qu'un chirurgien manipule lors de la planification d'opérations de neurochirurgie guidée par l'image. D'une manière générique, cette modélisation métier permet de représenter les objets nécessaires à la planification (étapes, cibles, actions, section du cerveau, etc.) qui permettent de comprendre la problématique du métier et de spécifier les besoins du logiciel de planification.

Le travail d'analyse du métier consistant en séances d'observation, en interviews d'acteurs et en analyses diverses, est souvent réalisé de manière empirique où toutes les informations sont stockées de manière non formelle. Il est alors difficile d'assurer la traçabilité de ces informations lors de la suite du développement. Les cas d'utilisation permettent de donner un cadre aux différentes analyses du métier. Par exemple, le cas d'utilisation que l'on nomme

[1] Il est possible de trouver des objets *métier*, comme une sonde échographique par exemple

Réaliser geste médical (cf. figure 4.2(a)) dans le cadre d'une opération chirurgicale, doit être clairement défini en terme d'amplitude de mouvements et de forces appliquées sur le patient, mais est indépendant par exemple du cas d'utilisation *Régler les paramètres d'un dispositif médical*. Une modélisation complète doit alors permettre de classer chaque action des acteurs dans un cas d'utilisation.

Les cas d'utilisation sont documentés par une description textuelle que les concepteurs adaptent en fonction de l'application. Dans le cadre des robots de service nous proposons de décrire les cas d'utilisation suivant plusieurs champs :

- nom du cas d'utilisation ;
- acteurs principaux et secondaires ;
- description du scénario principal du cas d'utilisation ;
- pré-condition à l'exécution du cas d'utilisation ;
- post-condition à l'exécution du cas d'utilisation ;
- alternative au scénario principal ;
- contraintes relatives à la qualité du service (QoS) ;
- contraintes techniques liées à la conception ;
- remarques diverses.

Le concept de qualité de service (QoS) sera décrit ultérieurement (section 4.2.3), et un exemple de description textuelle d'un cas d'utilisation sera donné sur la figure 4.4.

Sur la base des cas d'utilisation métier, et de leur description textuelle, il est ensuite plus facile de formuler les exigences pour l'intégration d'un robot au sein de la tâche comme cela est présenté dans la section suivante.

4.2.2 Intégration du dispositif technologique dans la tâche de service

Cette étape consiste à intégrer la technologie que l'on va utiliser, en l'occurrence un robot, au métier considéré. Il va alors falloir choisir quels cas d'utilisation seront associés aux différents acteurs. Le choix des relations entre les acteurs et les cas d'utilisation vont influer sur la sécurité. En effet, les cas d'utilisation sont une modélisation de la manière dont certains acteurs vont utiliser le système, mais ceci peut faire l'objet d'un choix, et c'est ce que l'on nomme l'*allocation des tâches*. Il s'agit de décider quelle sera la distribution de la charge de travail entre les acteurs et le système. Ceci est particulièrement critique pour des tâches qui existaient auparavant, et que l'on souhaite réaliser grâce à un système technologique comme un système de robotique de service. Par exemple, les travaux sur les robots médicaux de (Coste-Manière et al., 2002) de l'équipe CHIR[2], ont amené les auteurs a développer une procédure comportant des étapes de planification, validation, vérification et exécution. Ceci rend plus complexe la tâche du chirurgien qui doit alors réaliser des actions qu'il n'effectuait pas jusqu'alors. Face à cet ensemble de nouvelles tâches, il est donc important de bien définir qui les réalisera et dans quelles conditions. Notons que cette procédure résulte de l'identification

[2]http ://www.inria.fr/chir

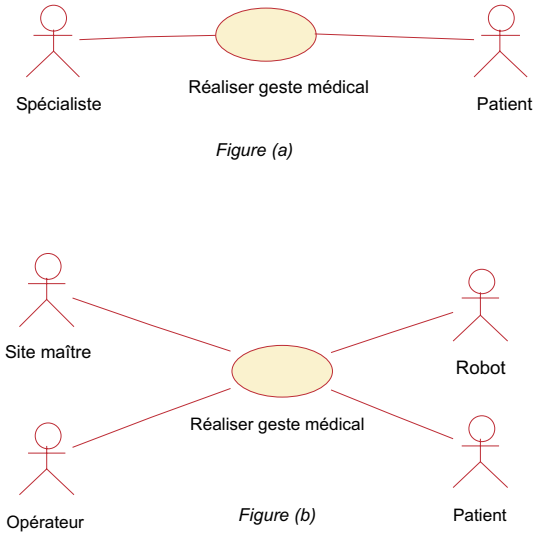

FIG. 4.2 – Modélisation d'un geste médical (a), et d'un cas de télé-médecine robotisée (b)

d'un phénomène dangereux dû à la complexité du système, qui est la présence d'états problématiques comme la collision des différents bras du robot portant des outils et un endoscope.

Bien que l'allocation des tâches soit fondamentale pour la sécurité, il est encore rare de trouver dans les conceptions de systèmes robotiques de telles analyses. Lorsqu'un concept naît dans l'esprit d'un concepteur, il est souvent associé à une première idée d'architecture sans qu'il y ait eu mise à plat des besoins, des acteurs et des contraintes. Nous proposons d'utiliser UML afin de modéliser la répartition des tâches. La détermination de cette allocation peut alors suivre des algorithmes comme ceux présentés par Laprie et al. (1995, p.231-236), Beevis et al. (1994) et Mersiol et al. (1998). Bien que cette approche s'effectue au début du processus de développement, elle permet néanmoins de guider les premiers choix technologiques, et de commencer à élaborer une architecture du système, et plus particulièrement du robot. C'est donc dès le début du développement que nous proposons d'utiliser la modélisation métier présentée dans la partie précédente, puis de dériver, de redéfinir ou de compléter chaque cas d'utilisation pour intégrer le nouveau dispositif technologique. Par exemple, pour un cas de télé-médecine, le cas d'utilisation *Réaliser geste médical*, identifié précédemment, ne sera plus relié au spécialiste car celui-ci se trouve à un autre endroit, mais à un acteur que l'on peut nommer *Site maître* et à un *Opérateur* étant sur place. Ce cas est représenté sur la figure 4.2(b).

4.2.3 Définition des frontières du système

Une des étapes de l'analyse du risque est la définition des frontières du système et, par extension, la définition des frontières de l'analyse. Elles sont modélisées en premier lieu sur les diagrammes de cas d'utilisation grâce à un cadre, où tout ce qui est à l'intérieur (cas d'utilisation ou acteurs) concerne le système que l'on souhaite analyser. On choisit alors parmi les cas d'utilisation identifiés précédemment, ceux qui vont concerner le sous-système analysé. Les cas non retenus sont considérés comme appartenant à un autre sous-système qu'il convient

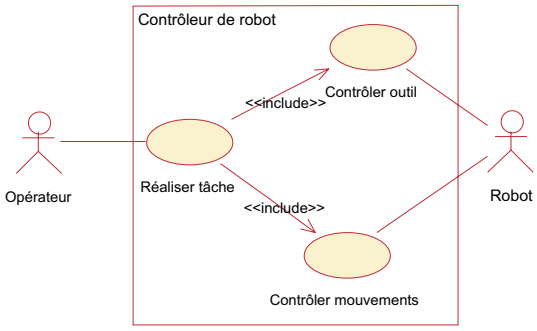

FIG. 4.3 – Exemple de diagramme des cas d'utilisation

Cas d'utilisation	Contrôler Mouvements
Acteurs	Opérateur, Robot
Description	L'opérateur contrôle les mouvements du robot en mode cartésien grâce à 2 joysticks. Un joystick permettant la localisation de l'effecteur suivant les axes x,y et z référencés par rapport au bâti, l'autre permettant d'effecteur les 3 rotations en ρ, θ, et ϕ
Pré-condition	Le robot est en position de repos
Post-condition	Le robot est remis en position de repos
Alternative	L'opérateur peut s'il le désire passer en mode articulaire
QoS	La vitesse de l'effecteur ou d'un axe ne doit pas dépasser 25 cm/s
Contraintes techniques	Les joysticks utilisés seront des LESIA2000 déjà utilisés pour un autre projet
Remarques	

FIG. 4.4 – Exemple de fiche de cas d'utilisation

néanmoins d'analyser séparément. Ainsi, sur la figure 4.3, on modélise avant tout les acteurs, l'*Opérateur* et le *robot*, ainsi que le *Contrôleur de robot* dont les frontières sont définies par le cadre extérieur. Cet exemple présente le cas très classique de la réalisation d'une tâche par un robot manipulateur comportant un outil (que ce soit un robot industriel ou de service).

Ce cas d'utilisation est détaillé de manière textuelle dans un document qui existe pour chaque cas d'utilisation. En effet, il est parfois impossible de représenter l'ensemble des exigences de manière graphique. Les informations concernant la sécurité peuvent apparaître dans ce document, et l'on peut choisir de regrouper l'ensemble des contraintes de sécurité, de fiabilité et de performances sous l'appellation QoS (*Quality of Service*) présentée au chapitre 3. Ce concept, dont l'intégration dans le langage UML est encore à l'étude au sein de l'OMG, devrait apparaître dans les prochaines versions d'UML. L'utilisation que l'on en fait ici est la plus simple qui existe, puisque nous proposons d'utiliser cette notion pour créer une rubrique supplémentaire dans la description textuelle des cas d'utilisation. Ainsi, pour le cas d'utilisation *Contrôler mouvements*, on peut réaliser une description textuelle comme celle présentée en figure 4.4. Cependant, la notion de QoS, comme il est dit au chapitre 3, est étroitement liée à la notion de sécurité, et les travaux actuels concernant les interactions entre la sémantique d'UML et cette notion, devraient permettre de faire le lien avec la gestion du risque.

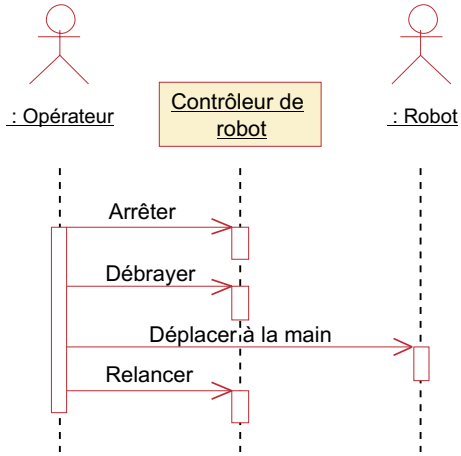

FIG. 4.5 – Exemple de diagramme de séquence pour un scénario du cas d'utilisation *Réaliser tâche* de la figure 4.3

4.2.4 Description des tâches des acteurs

La description du système est faite grâce aux cas d'utilisation, mais aussi grâce à des représentations plus concrètes, les diagrammes de séquence ou de collaboration, qui permettent de modéliser des scénarios appartenant aux cas d'utilisation. On dit alors que les diagrammes de séquence *réalisent* les cas d'utilisation, puisqu'ils donnent une description d'un cas concret de scénario. La figure 4.5 illustre l'utilisation d'un diagramme de séquence permettant de décrire un scénario du cas d'utilisation *Réaliser tâche* de la figure 4.3. Lorsque le système à analyser est simplement représenté par une instance, comme *Contrôleur de robot*, on parle de diagramme de *contexte*. On exprime effectivement le contexte dans lequel le système est utilisé. En définissant ainsi chaque scénario pour chaque cas d'utilisation, on regroupe l'ensemble des actions que les acteurs devront effectuer. Ceci permet de représenter l'ordre des actions qui doit être respecté, et de détailler les tâches dévolues à chaque acteur du système, et plus particulièrement aux acteurs humains. Cette activité, est fonction du degré d'avancement du développement, puisqu'il est possible de décrire chaque action avec un simple message général, comme par exemple *Arrêter* sur la figure 4.5, mais aussi de rentrer dans les détails pour exprimer comment l'Opérateur effectue cette action (boutons, interrupteurs, etc.), à quel moment, pourquoi, etc. Cette modélisation, tout en assurant une continuité avec le développement du système et donc la cohérence du modèle, permet de définir des tâches précises et détaillées, donc non ambiguës pour les acteurs.

Un avantage supplémentaire de cette représentation est de fournir une spécification des interfaces homme-machine. En effet, l'ensemble des messages émis par les acteurs humains permet de dresser l'ensemble des exigences fonctionnelles des interfaces. Le langage UML contribue ainsi à la prévention des fautes de conception, dues à des incompréhensions ou des ambiguïtés au niveau des interfaces.

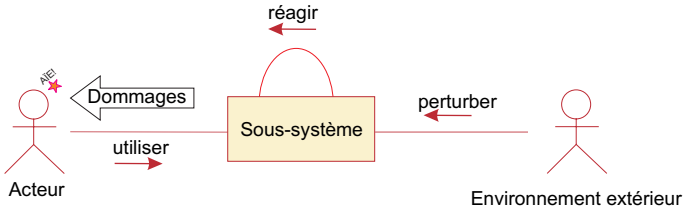

FIG. 4.6 – Interactions principales lors de l'utilisation d'un sous-système

4.3 Analyse des dangers et estimation du risque

4.3.1 Problématique

D'une manière simplifiée, pour tout système on peut représenter les interactions avec le diagramme de collaboration présenté sur la figure 4.6. L'*Environnement extérieur* est représenté sous la forme d'un bonhomme (un *stickman*) pour signifier qu'il agit à la manière d'un acteur sur le système. Le message *perturber* n'est pas forcément négatif, car le système peut être prévu pour répondre à certaines perturbations. Sur ce diagramme très simple ne sont représentés que les flots de contrôle. Tous les flots d'informations en sens direct ou inverse des sens des messages sont sous-entendus. Les dangers que l'on peut identifier sur ce diagramme sont :

- des perturbations négatives liées à l'environnement extérieur (message *perturber*) ;
- des mauvaises utilisations (message *utiliser*) ;
- des défaillances internes (message *réagir*) ;
- des combinaisons de ces phénomènes dangereux ;
- et des combinaisons d'interactions (échange de messages) non erronées mais induisant un danger (situation dangereuse).

Cette manière de représenter les interactions d'un système permet d'identifier les techniques nécessaires à l'analyse des dangers que l'on effectue ici. Tout d'abord, la complexité d'un système *socio-technique* comme celui représenté ici, nécessite la collaboration de nombreux spécialistes de chaque domaine. Ainsi, une première étape, largement utilisée dans de nombreux domaines, consiste à dialoguer avec ces spécialistes pour identifier les dangers d'un tel système d'une manière très globale. Cette technique, intitulée *analyse préliminaire des dangers*, est présentée dans la section 4.3.2.

Les trois premiers dangers de la liste ci-dessus, concernant les perturbations extérieures, les mauvaises utilisations et les défaillances internes, peuvent être vues comme des modes de défaillance des messages émis par l'environnement extérieur, les utilisateurs et le sous-système considéré. Cette analyse peut donc se faire entité par entité, en utilisant la technique de l'AMDEC, qui consiste à identifier puis à analyser l'ensemble de ces modes de défaillance en se basant sur la notion de *Message* fondamentale UML. Ce point est abordé dans la section 4.3.3, et l'on étudiera dans la section 4.3.4 comment cette démarche s'applique aux différents domaines.

Les deux derniers points de la liste, concernant les combinaisons des trois phénomènes dangereux identifiés ci-dessus et les combinaisons de messages non-erronés mais induisant une situation dangereuse (comme par exemple un interblocage), ne peuvent être traités par la technique de l'AMDEC. La technique d'analyse des arbres de fautes est un des moyens permettant de réaliser une telle analyse, et nous étudierons comment cette technique s'intègre dans une modélisation en UML dans la section 4.3.5.

> N.B. : L'identification des phénomènes dangereux et l'estimation du risque étant des activités que l'on mène simultanément, il convient de définir dès le départ les niveaux de probabilité d'occurrence et de gravité que l'on choisit pour l'estimation du risque. Sans définir quels risques seront acceptables ou non, on peut utiliser des échelles de valeur propre à l'application considérée. Dans les sections suivantes, nous utiliserons, à titre d'exemple, les échelles de valeur présentées à la section 1.4.3.4.

4.3.2 Analyse préliminaire des dangers

Beaucoup de travaux font référence à l'APR (analyse préliminaire du risque) ou à PHA (*Preliminary Hazard Analysis*), pour introduire au sein de l'analyse du risque une étape précédent l'analyse technique détaillée. Cette analyse préliminaire consiste à dialoguer avec des spécialistes du métier ou des utilisateurs, pour identifier les principaux événements redoutés. C'est une méthode participative au sein de groupes de travail. Il est possible de faire cette analyse très tôt dans le processus de développement. Par exemple, on peut effectuer une analyse du concept de télé-médecine, en réfléchissant sur les dangers majeurs, sans se préoccuper de la réalisation.

Les différences de terminologie au sein de cette activité (certains emploient le terme de *risque*, d'autres de *danger*, ou de *facteur de risque*) proviennent en partie de son aspect transversal et du fait de la collaboration entre des acteurs ayant une « culture du risque », et des acteurs du domaine concerné, possédant leur propre langage. Nous utilisons ici le terme de *danger* et non de *phénomène dangereux* comme préconisé par certaines normes (comme la ISO 14971 (2000)), car nous pensons qu'il est possible d'identifier lors de cette étape à la fois des *phénomènes dangereux*, mais aussi des *situations* et des *événements dangereux*, ce que l'on englobe par la notion de *danger*. De plus, lors de cette étape, intervenant dès le début de la conception, il est impossible d'évaluer les probabilités d'occurrence des événements redoutés, et donc de calculer le risque associé. Pour cela, le terme d'analyse préliminaire du *risque* ne peut être utilisé.

Dans le cadre de la robotique de service il est possible d'utiliser les différents moyens présentés en 1.4.3.3, comme la consultation de bases de données, l'interview de spécialistes, d'utilisateurs, etc. Les diagrammes des cas d'utilisation servent alors à spécifier les acteurs exposés aux dangers et à définir dans quel cas d'utilisation le système se trouve. Beaucoup d'études sur les robots industriels s'appuient sur une classification des dangers suivant les

Domaine	*Danger*	*Acteurs*			*Cas d'utilisation*		*Gravité*	*Solutions possibles*	*Remarques*
		Opérateur	Patient	Robot	Réaliser tâche	Configurer tâche			
Physique	Déplacement sur une zone sensible ou dangereuse du patient	X			X		1	Spécification d'une zone de travail avant la tâche. Contraindre les mouvements dans cette zone.	Envisager des solutions logicielles mais aussi matérielles
	Pression trop importante sur le patient	X			X		1	Contraindre les efforts des mouvement grâce à un capteur de force	
Electrique	Electrocution	X	X	X	X	X	1	Aucune partie métallique ne doit être en contact avec le patient	Répondre aux normes en vigueur pour le système électrique (fusible, alimentation aux normes, etc…)
	Décharge électrostatique	X	X	X	X	X	2	Les parties métalliques du robot doivent être reliées à la masse	

FIG. 4.7 – Exemple de tableau pouvant servir pour l'analyse préliminaire des dangers

domaines mécanique, physique, chimique, psychologique, etc. De telle démarches aident les concepteurs à identifier les dangers, et la présentation que l'on fait sous forme de tableaux permet de donner différents points de vue. Il est possible d'effectuer alors des classements en fonction des acteurs, ou des cas d'utilisation, ou du domaine du danger comme sur la figure 4.7. C'est pourquoi nous proposons d'utiliser un tableau modulaire contenant l'ensemble de ces informations. Il est possible d'utiliser pour ce tableau la classification de la gravité exposée dans la partie 1.2.1.2. Lorsque cela est possible, les concepteurs peuvent aussi indiquer la probabilité d'occurrence du dommage, et ainsi exclure du traitement les dangers présentant un risque nul ou minime.

Une première itération de cette activité permet dès le début de se poser des questions concernant la sécurité. Et plutôt que de proposer des solutions de conception, ce tableau doit permettre de compléter les cas d'utilisation et notamment les descriptions textuelles en introduisant des exigences de type QoS, ou des alternatives de scénarios.

La représentation de tels tableaux permet de guider les concepteurs dans l'identification des dangers. Ainsi, une représentation des dangers en fonction des cas d'utilisation, ou des acteurs exposés, offre un point de vue différent sur la problématique, et peut permettre ainsi de compléter l'analyse. De plus, lors des étapes suivantes, d'autres dangers seront identifiés, et ce type de tableau permet de synthétiser les informations.

4.3.3 Analyse des modes de défaillance

La technique présentée ci-dessus ne saurait suffire à une analyse de la sécurité, et il est évident qu'elle ne traite qu'un ensemble de dangers facilement identifiables. De plus, les mécanismes de production de ces dangers ne sont pas analysés. Une autre technique complémentaire, dite de *forward*, part des défaillances des composants du système, puis identifie les

dangers et les dommages induits par ces défaillances. Pour suivre cette démarche, la technique de l'AMDEC (Analyse des Modes de Défaillance et de leurs Effets Critiques), présentée dans le premier chapitre (1.5) est la plus utilisée. Elle est basée sur la connaissance des modes de défaillance des composants et de leurs effets sur le système. Le tableau présenté dans le chapitre 1 est un exemple d'utilisation de l'AMDEC pour des composants électroniques, mais cette technique est utilisée dans de nombreux autres domaines, dont celui des facteurs humains.

4.3.3.1 Analyse des modes de défaillance des messages

La notion de mode de défaillance est proche de celle de type d'erreur, et dans le cadre de notre analyse, nous utiliserons par la suite l'un ou l'autre de ces concepts de manière équivalente.

La technique de l'AMDEC consiste avant tout à identifier les erreurs pouvant apparaître au sein d'un système avant que la fabrication ait eu lieu. Pour la plupart des analyses, les erreurs identifiées sont spécifiques à l'application. À titre d'exemple, une rotation trop importante d'un actionneur d'un robot neuro-chirurgical est une erreur propre à l'application, et une même rotation sera considérée comme un fonctionnement normal pour une autre utilisation. Pour automatiser l'identification des erreurs, il est néanmoins possible d'utiliser des modèles d'erreurs génériques et donc utilisables pour différentes applications. Par exemple, lorsque des électroniciens analysent les circuits comportant des portes logiques, ils utilisent un modèle d'erreur commun à tous les systèmes utilisant ces composants :

- sortie collée à 1,
- sortie collée à 0.

De la même manière en robotique, il existe des classes de composants que l'on retrouve dans de nombreuses applications, comme les actionneurs et les capteurs, et il est possible d'utiliser des modèles d'erreur de ces composants. Cependant, ces modèles d'erreur s'appliquent de manière très réduite et ne concernent que quelques éléments physiques. Ainsi, des modèles d'erreurs plus élaborés devraient permettre de ne pas se restreindre aux composants électroniques. Pour cela, nous proposons de nous concentrer sur un des éléments qui n'est pas propre à une application : *le langage de modélisation UML*. En effet, par analogie avec les dispositifs électroniques comme les actionneurs et les capteurs, les constructions d'UML sont utilisées d'une application à l'autre. De plus, une autre dimension à cette généricité provient de la pluridisciplinarité du langage permettant de modéliser les éléments électroniques, informatiques, mécaniques et les acteurs humains. Cette généricité est double : elle devrait permettre de proposer des modèles d'erreur génériques aux applications (applicables à différents systèmes), mais aussi transversaux aux différents domaines (de l'électronique à la composante humaine).

La richesse de la notation UML nous amène à nous concentrer vers une des constructions de ce langage qu'il convient de définir avant de proposer des modèles d'erreurs. Le langage UML est défini par un métamodèle, où toutes les constructions sont inclues dans des *pa-*

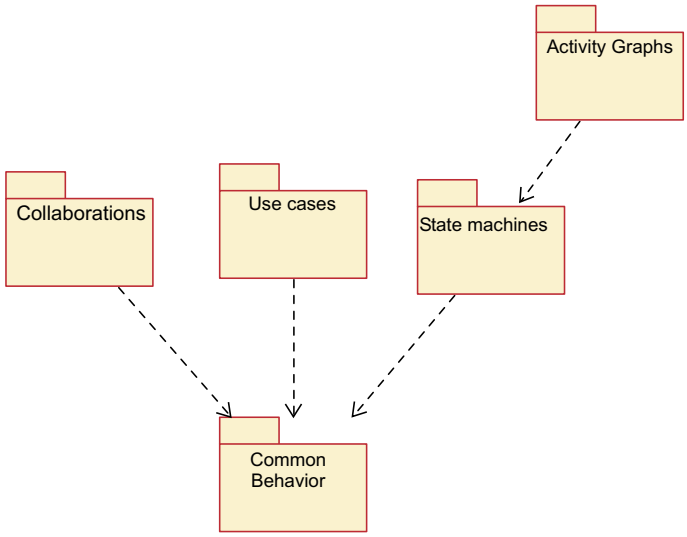

FIG. 4.8 – *Package Behavioral Elements*, diagramme tiré de la spécification 1.4 d'UML

ckages. La construction que nous proposons d'analyser, fait partie du *package Collaborations* (lui même inclus dans un des *packages* fondamentaux d'UML, *Behavioral Elements* présenté sur la figure 4.8), et concerne tous les éléments ayant trait à la notion de *Message*. Différentes raisons ont conduit à ce choix. Tout d'abord, l'analyse des défaillances doit permettre d'identifier le comportement du système pouvant induire des dommages. Il s'agit de considérer le système lors de son utilisation, et donc lorsqu'il est en action. Par analogie, un composant qui a une défaillance mais n'échange aucun message avec les autres composants qui sont eux-mêmes en relation avec les acteurs du système, ne sera pas dangereux, et ne nécessitera donc aucune analyse relative à la sécurité. Le danger provient de l'échange de ces messages entre composants ; s'il n'y a pas de message, il n'y a pas de danger. C'est donc l'analyse des modes de défaillance de ces messages et non des composants que nous proposons d'analyser.

Cette manière d'aborder le problème est déjà sous-jacente en électronique, même si l'approche est de prime abord basée sur la notion de composant. En effet, lorsque les électroniciens utilisent l'AMDEC pour les composants électroniques, ils étudient chaque composant et analysent les sorties de ces composants, et donc les messages qui sont envoyés. Par exemple, une analyse d'un capteur de position permet d'identifier les modes de défaillance suivants : sortie nulle, constante, aléatoire, dérive, etc. qui ne correspondent en fait qu'aux modes de défaillance du message *LirePosition()* émis par les composants souhaitant utiliser ce service. Comme il s'agit de son unique fonction, et par soucis de simplification, on parle alors de l'analyse des modes de défaillance du *Capteur*, alors qu'il s'agit de l'analyse des modes de défaillance de sa *fonction* principale, appelée par le message *LirePosition()*. Par extension, on peut donc dire que l'analyse des modes de défaillance du système passe par l'analyse des messages échangés entre les composants du système.

Cependant, le fait de se concentrer sur la notion de *Message* réduit la généricité de notre approche. Il existe en effet au sein de la notation UML le concept d'*Action* qui est l'élément

fondamental du métamodèle pour la modélisation du comportement dynamique. Cependant, afin de limiter notre étude et de se restreindre à des concepts proche des utilisateurs d'UML (le métamodèle est un concept qui n'est pas orienté utilisateur), nous avons préféré nous concentrer sur la notion de *Message*. De plus, les travaux actuels de formalisation d'UML montrent des différences importantes sur le concept d'*Action*. Ainsi, la version 1.5 d'UML, publiée par l'OMG (2003c) lors de la rédaction de ce manuscrit, a fortement modifié les concepts liés à l'*Action* en introduisant notamment un *package Action*, absent de la version 1.4 (OMG, 2001). De plus, cette spécification diffère actuellement des travaux actuels pour la version 2.0 (OMG, 2003a). En revanche, au sein de ces trois documents formalisant UML, la notion de *Message* semble stable, prouvant ainsi la force et l'importance de ce concept.

4.3.3.2 Modèles d'erreur des messages

Nous avons donc choisi de nous concentrer sur une des constructions d'UML : les Messages. Il s'agit maintenant de déterminer les erreurs possibles liées à ce concept.

Certains langages (comme ADA par exemple), possèdent ce que l'on nomme une *sémantique opérationnelle* et une *sémantique de vérification*. La sémantique opérationnelle permet de spécifier les aspects fonctionnels d'un système, et décrit donc la manière dont le système délivrera le service. En UML, c'est l'ensemble des diagrammes que l'on peut nommer sémantique opérationnelle. La sémantique de vérification définit des propriétés permettant de vérifier si des règles sont respectées. Par exemple, en ADA, il est possible d'exprimer des contraintes sur les valeurs des variables utilisées dans la partie opérative d'un programme. Cette sémantique permet d'une part d'identifier les erreurs consistant en un non-respect d'une de ces propriétés, puis de les traiter. La démarche possible est alors de regrouper l'ensemble des propriétés de la sémantique de vérification, et d'en dériver des modèles d'erreur. Par exemple, dans le cas classique du typage en ADA (on spécifie les valeurs admissibles d'une variable), un modèle d'erreur est l'utilisation d'une variable en dehors de ses limites.

Cette démarche peut s'appliquer à la plupart des langages, dont UML, mais dans de nombreux cas, la sémantique de vérification est implicite, noyée dans la sémantique opérationnelle, voire absente du langage. En UML, il existe un certain nombre d'éléments que l'on peut classer dans la sémantique de vérification. Tout d'abord l'utilisation de contraintes, représentées graphiquement par de accolades (comme par exemple *{ordered}*), permet de spécifier pour certains éléments une restriction d'utilisation, et ceci fournit donc une source d'erreur si elle n'est pas respectée. Il existe aussi, dans la spécification d'UML, les *Well-Formedness Rules* définissant un ensemble de contraintes exprimées avec le langage OCL (OMG, 2001). Cependant, la plupart des propriétés de vérification ne sont pas exprimées explicitement, et sont intégrées implicitement dans la sémantique opérationnelle. Par exemple, le fait d'avoir une séquence ordonnée de messages, au sein d'une interaction, est représenté dans le métamodèle, comme présenté sur la figure 4.9, par des associations *+predecessor* et *+successor*. Ces éléments ne sont pas des contraintes ou des propriétés permettant de vérifier le modèle, alors que

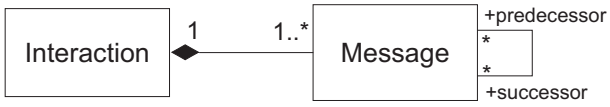

FIG. 4.9 – Métamodèle des éléments *Interaction* et *Message* dans la spécification 1.4 d'UML

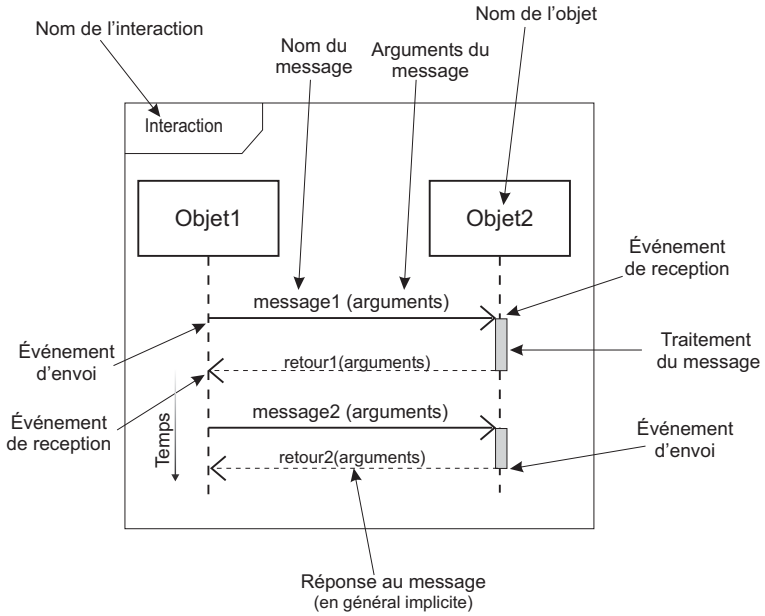

FIG. 4.10 – Éléments d'une interaction réalisée par l'envoi de deux messages

le fait de ne pas respecter l'ordre des messages est une source d'erreur. En conclusion, au lieu de chercher à regrouper l'ensemble des contraintes ou des *Well-Formedness Rules* relatives à la notion de *Message* au sein de la spécification 1.4 d'UML, nous proposons de définir les concepts inhérents à la notion de *Message* d'une manière générale. Cette démarche synthétise les éléments spécifiés dans la version 1.4, et permet en plus d'intégrer des éléments comme les contraintes de temps actuellement absentes du métamodèle (travail en cours par l'OMG notamment pour la version 2.0 d'UML).

Un message pouvant être la création d'un signal, l'appel d'une opération, la destruction ou la création d'une instance, nous présentons une description générique de l'ensemble des caractéristiques d'un *Message*. La représentation graphique sous forme de diagramme de séquence est donnée figure 4.10. On définit alors les différents éléments d'un message :

1. l'interaction à laquelle il appartient ;
2. les messages suivants et précédents ;
3. les objets émetteur et receveur de ce message ;
4. les événements d'envoi et de réception ;
5. les arguments définis en nombre, type et valeur ;
6. la réponse implicite définie aussi par des arguments, et des événements d'envoi et de réception ;

7. une durée de traitement du message.

À partir de ces éléments, on établit les erreurs possibles pour un message. Tout d'abord, un message appartient à une interaction, et l'envoi non prévu de ce message au sein d'une autre interaction est un type d'erreur que l'on trouve notamment lors de la manipulation d'interface homme-ordinateur. D'une manière générale on peut élargir ce type d'erreur en établissant un premier modèle d'erreur :

E.1. *Envoi d'un message n'appartenant pas à l'interaction prévue.*

Le deuxième point concernant l'ordre des messages peut aussi être la source d'erreur, et notamment pour les objets du type acteur humain. En effet, un utilisateur ayant à réaliser plusieurs tâches consistant en l'envoi de message, peut alors inverser ou oublier une des tâches. Il est alors possible d'étendre cette erreur à tout modèle en spécifiant deux types d'erreur :

E.2. *Exécution d'un ou plusieurs messages dans un ordre incorrect.*

E.3. *Omission d'un message au sein d'une séquence de messages.*

Un message est envoyé par un objet, mais il est possible que l'objet receveur soit inexistant. Ce type d'erreur, commun en informatique, permet de formuler une erreur générique :

E.4. *Absence de l'instance devant recevoir le message.*

Les caractéristiques liées aux événements d'envoi et de réception des messages permettent de définir des propriétés temporelles. En effet, pour ces événements, le temps est l'élément fondamental, et on en déduit des erreurs comme les retards (voire des retards infinis comme des blocages), ou même des messages en avance par rapport aux spécifications :

E.5. *Envoi ou réception d'un message en dehors des limites de temps spécifiées (trop tôt ou trop tard)*

Les arguments d'un message constituant les paramètres de l'opération ou du signal appelé doivent correspondre en nombre, type, et valeur à celles attendues pour l'objet receveur. Cette propriété partiellement exprimée en OCL dans les *Well-Formedness Rules* de la spécification d'UML, permet d'exprimer trois types d'erreur :

E.6. *Le type des arguments d'un message est différent du type des paramètres attendus par le receveur.*

E.7. *Le nombre d'arguments d'un message est différent du nombre de paramètres attendu par le receveur.*

E.8. *La valeur des arguments d'un message est différent de la valeur des paramètres attendus par le receveur.*

La réponse souvent implicite à un message peut être caractérisée par des arguments (par exemple l'appel d'une opération retourne une valeur), mais aussi par des événements d'envoi et de réception. Les erreurs temporelles liées à la réception ou à l'envoi de la réponse sont

déjà exprimées dans l'erreur E.5. Ceci mène à l'identification d'une erreur liée en général à un message faisant appel à une opération (comme par exemple *LirePosition* pour un capteur de position) :

E.9. *Les valeurs retournées par la réponse d'un message ne correspondent pas aux valeurs attendues (comme par exemple : constante, aléatoire, hors limites, etc.)*

La dernière caractéristique concernant la durée de traitement d'un message, correspond au temps compris entre le moment où l'objet reçoit le message et le moment où le receveur renvoie une réponse. Cette réponse peut être une valeur retournée, la construction ou la destruction de l'objet, l'envoi d'un signal, etc.). On identifie alors le type d'erreur suivant :

E.10. *Traitement d'un message en dehors des limites de temps spécifiées*

En complément des éléments que l'on a notés sur le diagramme, il convient de prendre en compte l'élément *lien* caractérisant la relation entre les objets émetteur et récepteur, et permettant le transmission du message. Cela permet alors de formuler le type d'erreur suivant :

E.11. *Absence du lien entre les objets émetteur et récepteur*

4.3.3.3 Proposition de tableau générique de l'AMDEC pour une analyse système

En se basant sur les normes (MIL-STD-1629A, 1980) pour des analyses fonctionnelles ou par composants, ainsi que sur les concepts exposés au chapitre 1 et sur la notion de message analysées précédemment, nous proposons ici de représenter dans le tableau de l'AMDEC les éléments suivants (cf. figure 4.11) :

- le nom du composant s'il n'a qu'une fonction, ou le nom du message analysé,
- les modes de défaillance, ou les erreurs identifiées grâce à des modèles d'erreur comme ceux proposés dans la section 4.3.3.2,
- les causes de ces modes de défaillance en restant dans une limite de causalité assez réduite,
- les effets locaux, sur un niveau supérieur, et sur le système (et notamment les dommages),
- les données pour l'estimation du risque (gravité du dommage, occurrence du mode de défaillance, risque associé),
- les moyens de détection en ligne des modes de défaillance et des effets,
- les moyens possibles de prévention et de protection du risque,
- et diverses informations.

Il est important de bien noter l'utilisation de la terminologie du risque exposée au chapitre 1. Dans l'AMDEC, ce qui est nommé *Modes de défaillance* correspond à la notion d'erreur que l'on retrouve dans la terminologie de la sûreté de fonctionnement exposée par Laprie (1992). Le but de ces tableaux n'est pas d'effectuer une étude en profondeur de chacun des points et, notamment pour la colonne relative aux causes, il est important de ne pas se perdre

Date :17/01/2003 **Par** : J.Guiochet			**Nom du projet**						**Page :** /
Code	**Composant / Fonction/ Message**	**Mode de défaillance** (erreur)	**Cause de la défaillance**	**Effets** : a. Niveau local b. Niveau supérieur c. Niveau système	**Risque**			**Moyens de détection possibles** (en ligne) a. Mode défaillance b. Effets	**Solutions possibles** : a. Prévention b. Protection c. Autres actions d. Remarques
					Gravité	Occurrence	Risque		

FIG. 4.11 – Exemple de tableau de l'AMDEC

dans des causes de causes, mais de synthétiser les principales informations qui permettent d'obtenir une analyse système.

Le risque est ici calculé à partir d'une estimation qualitative de la probabilité d'occurrence d'un mode de défaillance et de la gravité du dommage induit. Or, par définition (cf. chapitre 1), le risque doit être calculé avec la probabilité d'occurrence du dommage lui-même (et non du mode de défaillance). De plus dans une analyse comme celle-ci, il est possible de retrouver le même dommage pour différents modes de défaillance. Cette différence s'explique par les besoins pratiques des concepts de calcul du risque. Il est plus facile de calculer d'abord qualitativement le risque relatif à chaque mode de défaillance, puis d'en déduire le risque du dommage associé, en combinant les différents calculs de probabilité des modes de défaillance.

La colonne *Solutions possibles* du tableau de la figure 4.11 concerne des moyens possibles pour la réduction du risque. Il est important de noter que ces moyens ne sont pas implémentés directement, et qu'une évaluation du risque doit être faite préalablement. Nous avons choisi de représenter les moyens de prévention et de protection, mais une telle analyse peut mener à d'autres moyens. Par exemple, ayant identifié des messages dont il est difficile d'estimer la probabilité d'occurrence, mais que l'on peut qualifier de critiques, un moyen de réduire le risque est l'utilisation de techniques d'élimination des fautes (vérification, validations, tests, etc.). D'une manière générale, l'utilisation de l'AMDEC que l'on propose ne suit pas une démarche systématique, où chaque mode de défaillance est évalué en terme de probabilité et de gravité, puis traité. La principale raison vient de l'impossibilité d'estimer la probabilité, même de manière qualitative, de tous les modes de défaillance. Un des exemples le plus probant est sans doute les modes de défaillance du logiciel dont il est impossible de déterminer la probabilité d'occurrence. Ainsi, l'AMDEC permet avant tout de souligner des points de conception critiques en terme de sécurité, et lorsque cela est possible d'estimer le risque.

En dernier lieu, l'utilisation de l'AMDEC dépend fortement du degré d'avancement du développement, car elle est directement dépendante du niveau de détails des modèles. Dans notre approche, il nous semble important de se concentrer sur les premières étapes du processus de développement, car c'est à ce moment que les exigences de sécurité, les choix d'architecture, et les principaux phénomènes dangereux sont identifiés.

4.3.4 Application aux différents domaines

Une approche système doit permettre la prise en compte de toutes les composantes (électronique, informatique, mécanique et facteurs humains), mais il est évident que ce sont les spécialistes de chaque domaine qui possèdent la maîtrise des informations. Chacun possédant son propre langage et ses propres techniques, il est parfois complexe de réunir l'ensemble de ces informations pour effectuer une analyse globale du système. En utilisant le langage UML, ainsi que les modèles d'erreur associés au concept de message, on peut rendre homogène l'analyse du risque des différents domaines. Les sections suivantes présentent des exemples d'utilisation des modèles d'erreur identifiés dans la section précédente afin de démontrer la faisabilité d'un telle démarche.

Tout d'abord, les modèles d'erreur seront appliqués aux acteurs identifiés dans la modélisation en UML. Les acteurs, dans la sémantique d'UML, sont en général les utilisateurs ou les dispositifs extérieurs interagissant avec le système. Les particularités de ces deux types nous amènent à les traiter de manière séparée dans cette section. Il existe cependant un autre type d'acteur qui apparaît sur le diagramme de collaboration présenté précédemment sur la figure 4.5 : l'*environnement extérieur*. Dans notre cas nous n'analyserons pas les perturbations provenant de cet acteur, car pour une tâche de service en milieu protégé humain comme une tâche médicale, ces interactions sont minimes, ou prévues par la spécification. Cependant, il est possible d'analyser les effets des messages émis par l'acteur *environnement extérieur* en répertoriant toutes ses actions néfastes sur le système.

Cette partie présente donc des exemples d'analyses des différents messages :

- les messages émis par les acteurs de type dispositif extérieur (section 4.3.4.1) ;
- les messages émis par les acteurs humains (section 4.3.4.2) ;
- les messages échangés entre composants électroniques (section 4.3.4.3) ;
- les messages échangés entre composants mécaniques (section 4.3.4.4) ;
- le messages échangés au sein du logiciel (section 4.3.4.5).

Pour chaque domaine trois points seront abordés. Tout d'abord la manière dont il est possible de modéliser les objets du domaine est présentée (paragraphe *Modélisation des objets et des interactions*), puis les types d'erreur spécifiques au modèle sont identifiés à partir des modèles d'erreur établis précédemment (paragraphe *Types d'erreur*), et enfin un exemple d'analyse des erreurs des messages est développé (paragraphe *Analyse des modes de défaillance*). Pour les paragraphes *Types d'erreur*, les modèles d'erreur de E.1 à E.10 ne seront pas instanciés systématiquement considérant qu'il existe des cas impossibles ou triviaux.

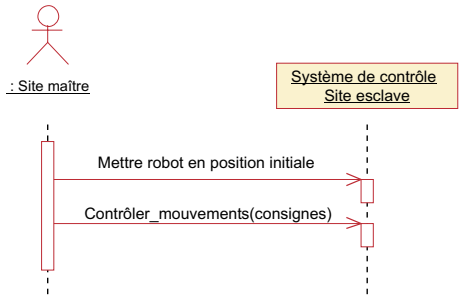

FIG. 4.12 – Diagramme de séquence illustrant des interactions entre un site maître et un système esclave

4.3.4.1 Analyse des modes de défaillance des messages provenant des acteurs de type dispositif extérieur

Modélisation des objets et des interactions

Pour de nombreux systèmes, il est possible de représenter des dispositifs extérieurs comme des acteurs, s'ils interagissent avec le système de manière autonome. Ainsi, pour la télémédecine, le *Site maître* de la figure 4.2 peut être noté comme un acteur. Les messages échangés avec le système peuvent se modéliser grâce à des diagrammes de collaboration ou de séquence comme sur la figure 4.12. Les dispositifs extérieurs annotés en tant qu'acteurs représentent souvent des systèmes imposants des contraintes de temps. Par exemple, certains utilisateurs d'UML (Douglass, 1999) représentent de simples *timers* en tant qu'acteurs, pour exprimer notamment le fait qu'il s'agit d'une application temps-réel. Les messages envoyés sont alors annotés de contraintes exprimées en tant que notes comprises entre accolades.

Dans les modèles que l'on a présentés précédemment, nous avons fait apparaître un acteur propre à la robotique : le robot réduit à son architecture mécanique. Ce choix est très controversé, car il est possible de considérer le robot comme un dispositif au même titre que les capteurs et les actionneurs faisant partie du système modélisé par les cas d'utilisation. Cependant, au cours des différentes modélisations que nous avons pu réaliser, il s'est avéré que la structure même du robot devait faire l'objet d'une séparation du système ne serait-ce que pour les interactions particulières entre les humains et la structure mécanique du robot. Bien que nous abordions cet acteur dans cette section, il est évident que l'analyse des modes de défaillance de cet acteur particulier concerne le domaine mécanique. La notion d'acteur est une notion transversale aux différents domaines.

Types d'erreur

Pour un acteur du type dispositif extérieur, les messages échangés correspondent soit à des opérations, soit à des signaux. Tous les modèles d'erreur de E.1 à E.10, peuvent être utilisés pour ces messages. Ainsi, pour le message *Contrôler_mouvements(consignes)* de la figure 4.12, les erreurs peuvent être l'envoi d'arguments faux (erreur E.6, E.7 et E.8), des

Date :17/01/2003 **Par** : J.Guiochet			**Nom du projet**						**Page :** /
Code	**Composant / Fonction/ Message**	**Mode de défaillance** (erreur)	**Cause de la défaillance**	**Effets** : a. Niveau local b. Niveau supérieur c. Niveau système	**Risque**			**Moyens de détection possibles** (en ligne) a. Mode défaillance b. Effets	**Solutions possibles** : a. Prévention b. Protection c. Autres actions d. Remarques
					Gravité	Occurrence	Risque		
	Contrôler_mouvements (consignes)	Consignes fausses (erreur de type E.6 à E.8)	Faute `Site Maître` ou faute lien	c. Mouvement non désiré	1			Vérification cohérence des consignes (filtrage numérique)	a. Utilisation protocole standardisé b. Ignorer la valeur des consignes, mettre le contrôleur en position d'attente

FIG. 4.13 – Analyse d'un mode de défaillance du message *Controler_mouvements()*

retards ou des décalages dans le temps (E.5 et E.10), ou des erreurs du *Site maître* lors de l'envoi des messages (E.1, E.2, E.3, et E.4). Pour un acteur de ce type, il est important de noter que les types d'erreur incluent les défaillances de l'objet émetteur mais aussi les défaillances du lien avec l'objet receveur (E.11). Cela permet entre autres de couvrir les aspects de télécommunication de certaines applications comme celles de la télémédecine.

Analyse des modes de défaillance

La modélisation de l'ensemble des messages échangés en diagramme de séquence permet d'effectuer une analyse très tôt dans le processus de développement, et de formuler dès le début des exigences de sécurité ou de sûreté de fonctionnement, sans préciser les choix de conceptions pour la communication entre les dispositifs. Par exemple, dans le cas du diagramme de la figure 4.12, le type de communication n'est pas spécifié (intranet, internet, RS232, etc.), mais il est possible d'ores et déjà d'envisager les défaillances des messages provenant du site maître.

La figure 4.13 montre une analyse générique du mode de défaillance de type E.6 ou E.7 ou E.8, du message *Contrôler_mouvements(consignes)* du diagramme de séquence de la figure 4.12. En se basant sur les autres diagrammes UML, on évalue les effets de ce mode de défaillance. Puis, à partir de cette analyse, il est possible de proposer une solution pour ce mode de défaillance en modifiant les modèles UML du système. La solution proposée ici consiste à changer temporairement l'état du contrôleur de robot. Ceci est illustré sur le diagramme d'état de la figure 4.14. Lors des premières étapes du processus de développement, il est parfois difficile d'estimer la probabilité d'occurrence d'un mode de défaillance. Ce champ est laissé vide dans le tableau de l'AMDEC, alors que la gravité est estimée à un niveau 1 (cf. tableau d'estimation du risque du chapitre 1). Cependant, lorsque les modèles sont développés plus précisément, et que certains choix de conception sont effectués, l'estimation de la probabilité d'occurrence devient possible.

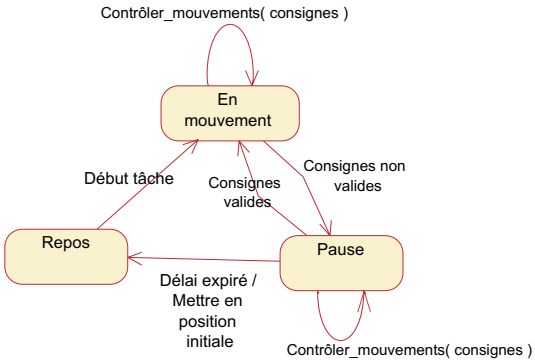

FIG. 4.14 – Diagramme d'états prenant en compte un mode de défaillance de *Contrôler_mouvements()*

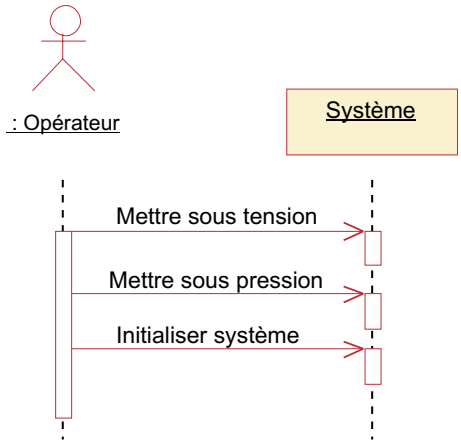

FIG. 4.15 – Procédure de mise en route d'un système quelconque

4.3.4.2 Analyse des modes de défaillance des messages provenant des acteurs humains

Modélisation des objets et des interactions

La modélisation des acteurs et de leurs interactions avec les objets du système est décrite dans la première partie de ce chapitre (4.2). Les acteurs apparaissent effectivement dans de nombreux diagrammes, les plus importants étant les diagrammes des cas d'utilisation (voir figures 4.2 et 4.3) et de séquence (voir figure 4.15). Les messages provenant des acteurs sont principalement identifiés sur les diagrammes de séquence et constituent les tâches qui incombent aux acteurs. Ces tâches sont souvent définies comme des séquences ordonnées de messages envoyés au système. L'objet receveur de ces interactions est avant tout le système global. Mais en raffinant les diagrammes, on fait apparaître les véritables objets recevant les messages émis par les acteurs : les interfaces. Ceci montre bien l'importance de ces objets dans l'analyse des erreurs humaines et notamment pour la réduction des risques.

Types d'erreur

L'ensemble des types d'erreur que l'on a identifié s'appliquent à ce domaine. Les erreurs les plus communes sont par exemple l'exécution dans un ordre incorrect d'une séquence de message (erreur E.2), l'omission d'un message (erreur E.3) et l'occurrence d'un message non connu au sein d'un interaction (erreur E.1). Cette dernière erreur peut consister en l'appui non prévu sur un bouton au sein d'un scénario. Un autre type d'erreur commun est le passage d'arguments qui ne correspondent pas à ceux attendus en nombre, en type ou en valeur par le receveur (erreurs E.6, E.7 et E.8). L'interprétation que l'on peut donner de cette erreur est la mauvaise exécution d'une tâche, de sorte que les données envoyées au système soient fausses. Un exemple est la mise sous pression d'un système hydraulique, où l'opérateur tourne un bouton proche d'un manomètre. L'action *Mettre sous pression* modélisée comme un message sur la figure 4.15, sous-entend le passage d'un argument qu'est la pression désirée. L'opérateur peut alors s'il n'y a pas de sous-système prévu, monter la pression du système jusqu'à un niveau trop important ou insuffisant. Le moyen de réduire le risque est alors d'introduire des moyens de prévention comme par exemple un manomètre bien visible comportant des marquages, ou un mécanisme de protection limitant la pression quoique fasse l'opérateur. De nombreux mécanismes présents dans les interfaces réduisent ainsi le risque lié à cette erreur humaine. Un autre exemple plus courant est la vérification au niveau de l'interface graphique d'un ordinateur de la validité des données entrées par un utilisateur.

L'erreur de type E.4 est plus rare car elle consiste en l'absence de l'objet devant recevoir le message, c'est-à-dire en l'absence d'interface. Dans un système graphique de fenêtres, cette erreur est possible mais reste tout de même assez rare. En revanche l'erreur E.5 peut être la source d'importants dégâts s'il existe des contraintes de temps sur l'envoi des messages, mais ceci reste très spécifique à chaque application tout comme l'erreur E.10 (durée de traitement du message).

Le type d'erreur E.9 concernant la validité des valeurs retournées après l'envoi d'un message, comme par exemple l'accès à une variable d'état d'un système, peut être une source de dommage si l'acteur doit ensuite agir en fonction de cette valeur.

Analyse des modes de défaillance

La démarche que nous proposons pour l'analyse des erreurs humaines, suit le même processus que pour l'analyse des autres phénomènes dangereux, mais intègre des notions supplémentaires. Au sein de l'analyse du risque la démarche que l'on propose suit les étapes de la figure 4.16. Cette représentation est proche de celle proposée par Pocock et al. (2001), qui n'incluaient cependant pas les liens avec les modèles et le processus d'analyse du risque.

C'est la connaissance croisée entre les propositions d'interface, les scénarios d'utilisation et les modèles d'erreur humaine qui permettent d'identifier les erreurs possibles propres à l'application. Pour chaque cas d'utilisation où des scénarios sont déclenchés par des acteurs humains, une description complète des messages avec des diagrammes de séquence,

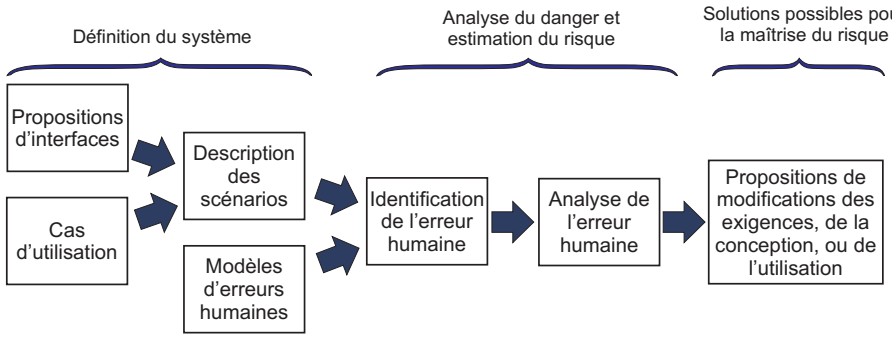

FIG. 4.16 – Gestion du risque induit par les erreurs humaines

permet d'analyser l'ensemble des erreurs humaines. Il est possible de représenter graphiquement certaines erreurs (inversion de message, omission, etc.) directement sur les modèles, et d'en déduire les effets sur le système et leur gravité. L'estimation du risque reste cependant difficile puisque la probabilité d'occurrence d'une erreur humaine reste insondable. Cependant des modèles de performance expérimentaux (bases de données) mais aussi des données empiriques (comme le « bon sens commun ») peuvent aider à estimer le risque lié à chaque erreur. L'analyse que l'on fait est avant tout une estimation de la criticité de certaines tâches (ou certains messages) qui permet ensuite de sécuriser les plus critiques. Comme pour le cas précédent, les tableaux de l'AMDEC conviennent à la documentation de cette analyse, et les solutions possibles identifiées sont directement réinjectées dans les modèles UML, ou dans la conception des interfaces.

4.3.4.3 Analyse des modes de défaillance des composants électroniques

Modélisation des objets et des interactions

Les diagrammes de déploiement sont en général utilisés pour la modélisation des composants électroniques. Ceux-ci ont été créés à la base pour montrer les nœuds d'informations, et la manière dont le logiciel se déployait sur ces nœuds. Ces diagrammes sont particulièrement adaptés aux systèmes distribués, où les objets informatiques qui communiquent peuvent être disposés sur des processeurs différents. Pour les systèmes de l'informatique industrielle, dont font partie les robots, ces diagrammes ont été adaptés (Douglass, 1999), et l'utilisation des nœuds a été étendue aux différents dispositifs se trouvant en relation avec les processeurs. Un exemple de diagramme de déploiement est présenté sur la figure 4.17. Il s'agit d'un système de contrôle d'un actionneur fonctionnant avec une pression régulée par un *Convertisseur Intensité/Pression*. L'intérêt d'un tel diagramme est qu'il montre les choix technologiques qui sont faits (par exemple le PC sous VxWorks), mais son utilisation pour une analyse des défaillances du système est très réduite. En effet, seuls les composants purement électroniques sont modélisés et le sens des informations échangées n'est pas représenté. Pour effectuer une analyse des modes de défaillance, une représentation sous forme de diagramme de classes est plus complète et peut suffire. Un exemple est donné sur la figure 4.18. Il est possible d'annoter sur ce

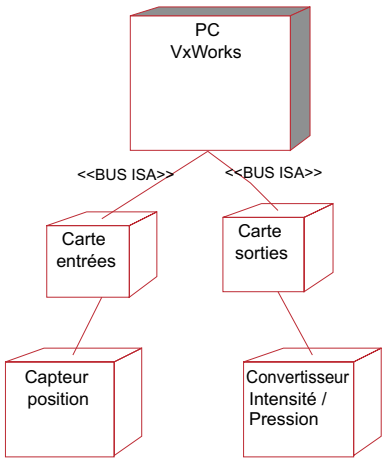

FIG. 4.17 – Exemple d'utilisation d'un diagramme de déploiement en informatique industrielle

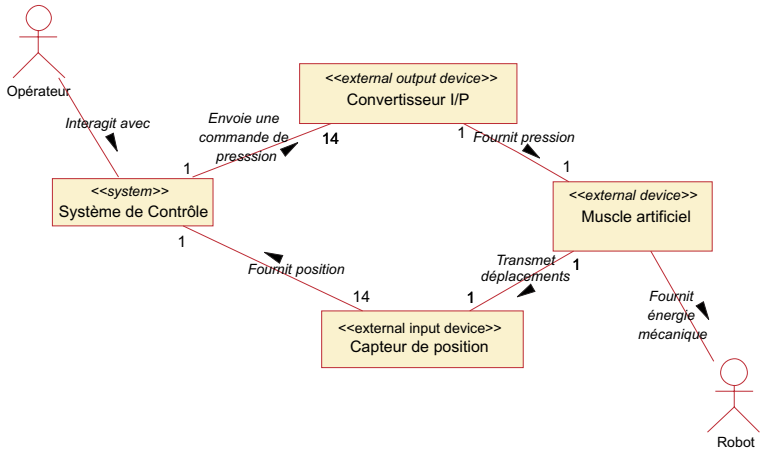

FIG. 4.18 – Diagramme de classes représentant les dispositifs extérieurs au logiciel

diagramme de classes, comme le propose Gomaa (2000), le type de relation entre les classes en notant le sens des informations échangées et en utilisant les stéréotypes *«external device»*, *«external output device»* et *«external input device»*. Le diagramme ainsi représenté exprime le fait que les classes modélisent des dispositifs extérieurs au sous-système comportant du logiciel annoté par le stéréotype *«system»*, car l'utilisation d'UML dans ce cas est essentiellement orientée vers le développement du logiciel. L'intérêt de ce diagramme est sa capacité à représenter sur une même vue les composants électroniques, les sous-systèmes comportant du logiciel comme le *Système de contrôle*, et des composants matériels directement dépendants de composants électroniques comme le *Muscle artificiel*.

Le but n'est pas de représenter ici la composition même des composants électroniques mais d'identifier les modes de défaillances de ces composants et d'analyser les effets sur le système. En conséquence, sans connaître la nature des échanges électroniques (intensité, tension, trames, etc.) entre les dispositifs, il est possible d'effectuer une analyse en se basant sur la notion de message. Ainsi, un capteur de position peut être vu comme un composant délivrant le service *lirePosition()* qui retourne une valeur lue de la position. On peut aussi parler de l'opération *lirePosition()* que l'on peut invoquer par un message. Il est possible de faire le lien avec une AMDEC classique qui se base sur la notion de *fonction* pour décrire ce service. Dans notre approche objet, on peut interpréter la notion de *message*, comme un appel à un *service*. Les diagrammes de séquence ou de collaboration sont alors utilisables.

Pour identifier les effets des défaillances, il peut être aussi utile de s'appuyer sur des modèles mathématiques, non exprimables en UML. Le cas le plus courant est la modélisation d'une boucle de rétroaction, toujours présente dans les contrôleurs de robot, grâce aux schémas blocs décrivant la formule mathématique du calcul de la commande dépendant de valeurs fournies par les différents composants électroniques. On peut imaginer que d'autres modèles (formules, autres diagrammes, etc.) décrivant l'aspect automatique ou mathématique peuvent être intégrés pour identifier les effets, et cela bien avant qu'une modélisation en UML du contrôleur soit effectuée.

Types d'erreur

Comme présenté précédemment, l'accès à un service rendu par un capteur (comme *lirePosition*) peut être interprété comme un message retournant une valeur. L'erreur la plus fréquente est donc une valeur retournée différente de celle attendue (erreur E.9) et donc ne reflétant pas la véritable valeur de la grandeur mesurée. Les instances communes d'erreur sont un pic de valeur, une dérive, une valeur constante, une valeur aléatoire, etc. Il est possible d'observer ces mêmes erreurs pour l'envoi d'argument (erreur E.8). Par exemple, l'envoi d'un message avec un argument faux est le cas d'un convertisseur tension/intensité envoyant un pic d'intensité à un moteur.

L'instance principale de l'erreur E.1 (envoi d'un message n'appartenant pas à l'interaction prévue) est en général caractérisée par l'envoi d'un flux d'énergie non prévu, vers un objet

receveur non prévu, l'objet receveur pouvant être un humain, ou une partie du système. Les plus communs sont les décharges électrostatiques, les explosions, etc. Par exemple, une erreur de ce type est possible pour le *Convertisseur Intensité/Pression* de la figure 4.18, qui peut provoquer un flot d'air en cas de fuite, vers les acteurs humains (un autre effet étant la chute de pression dans un muscle). Il est donc important pour cette analyse de bien prendre en compte d'une part les messages émis par chaque composant vers le composant de même niveau pour identifier la propagation des fautes, mais aussi d'imaginer les actions possibles vers d'autres composants du système (acteurs ou non).

Les erreurs E.5 ainsi que E.10 exprimant l'envoi d'un message et le traitement d'un message en dehors des limites de temps spécifiées, sont des modes de défaillance fréquents pour les dispositifs électroniques effectuant un traitement complexe d'une information. Le cas le plus flagrant est celui des microprocesseurs, dont l'activité peut résulter en un retard des messages échangés. En considérant ici l'aspect purement électronique (l'aspect logiciel est traité ultérieurement), ces dispositifs peuvent être vus comme les autres composants électroniques, mais avec des responsabilités supplémentaires.

Il est possible d'instancier le type d'erreur E.11, concernant l'absence de lien entre les objets, notamment pour les composants électroniques dont les communications requièrent des liens complexes comme des bus de données.

Analyse des modes de défaillance

Dans des systèmes électroniques complexes, la techniques de l'AMDEC est utilisée pour décrire des erreurs à un niveau de détail très élevé. Par exemple, on décrit pour les composants électroniques les erreurs de type micro-coupure, ou les défaillances d'un transistor (court-circuit ou circuit ouvert), et cela permet d'obtenir des estimations du risque quantitative. Une autre utilisation de l'AMDEC est celle qui consiste à considérer non pas chaque composant, mais chaque dispositif électronique d'un point de vue fonctionnel. Le niveau de détail étant plus faible, les analyses permettent d'estimer les effets plus facilement. Dans notre cas, pour les systèmes robotiques, le fait de se placer dans une analyse système des messages modifie l'utilisation que l'on peut faire de ces techniques et plus particulièrement de l'AMDEC.

Un concept fondamental de cette approche est de ne pas dissocier les composants de leur utilisation. Dans une approche plus classique, le fait de savoir qu'un capteur peut avoir un certain nombre de défaillances sans les associer à des messages fournit des modèles d'erreurs mais ne met pas en valeur le contexte dans lequel ce capteur est utilisé. Or le contexte d'utilisation est particulièrement important pour le développement de systèmes innovants où les composants peuvent être détournés de leur utilisation habituelle.

Un autre intérêt d'utiliser une analyse en fonction des messages est la prise en compte des défaillances des liens de communication (des associations) entre les composants. Cet aspect est très important pour les composants électroniques. En effet, chaque message est dépendant de la cohérence du lien entre deux objets, et lorsque l'on analyse une erreur de message

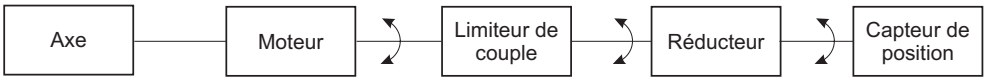

FIG. 4.19 – Diagramme de blocs tiré de l'article de Dombre et al. (2001)

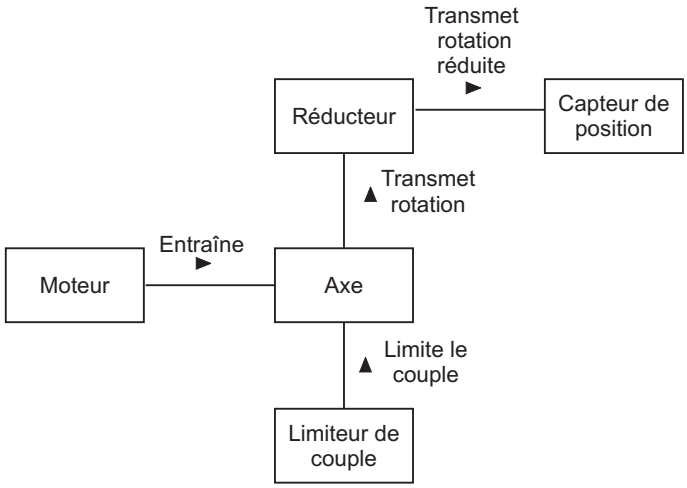

FIG. 4.20 – Diagramme de classes représentant les composants mécaniques d'une articulation

il convient de prendre en compte la défaillance de ce lien. Par exemple, si l'on étudie les erreurs du message *LirePosition()* envoyé à un capteur, et que l'on identifie une erreur du type incohérence des valeurs retournées (E.9), les causes peuvent être des défaillances internes du capteur, mais aussi des perturbations électromagnétiques sur le capteur ou sur le lien reliant le capteur. Ceci nous permet de soulever le fait que la probabilité d'occurrence de défaillance d'un message invoquant un service d'un composant ne se réduit pas aux données concernant les défaillances de ce composant fournis par le constructeur. Elle prend en compte l'ensemble des probabilités d'occurrence des modes d'erreur liés à ce message.

Une fois les modes de défaillance identifiés, les modèles UML permettent d'analyser les effets. Comme précédemment, les solutions possibles pour la maîtrise du risque concernent des modifications des exigences, de l'utilisation et de la conception.

4.3.4.4 Analyse des modes de défaillance des composants mécaniques

Modélisation des objets et des interactions

L'architecture et les composants mécaniques d'un robot sont choisis par les mécaniciens en fonction des exigences fonctionnelles et de performances. Pour ce développement, les techniques de modélisation sont très spécialisées à la mécanique. Il est possible de trouver cependant dans les études mécaniques une modélisation très simplifiée en schéma blocs présenté en figure 4.19 (reproduction partielle de l'article de Dombre et al. (2001)).

Un tel diagramme fait apparaître les composants les plus importants de la chaîne cinétique permettant d'effectuer les mouvements d'une articulation, et le *Réducteur* réalise une mise en

forme de l'amplitude de la rotation de l'axe en vue d'une mesure par le *Capteur de position*. Cette représentation prend en compte l'aspect physique de la distribution des composants, et bien que les composants interagissent avec l'axe, ils sont représentés de manière séquentielle comme dans la réalité. La modélisation en objets ne tient compte que des interactions, et donc permet de mettre en évidence les principaux échanges de messages. Il est important de noter que tout message sera du type *transmission d'énergie mécanique*. Cela nous conduit à proposer une modélisation représentée sur la figure 4.20. Il est ainsi possible d'interpréter une conception mécanique en un ensemble d'objets communicants. Ainsi, l'objet *Moteur* recevant un message dont les arguments sont une commande en position ou en vitesse, traitera ce message, et entraînera le moteur ce qui peut être vu comme un message *Tourner(vitesse)* par exemple.

Types d'erreur

Les messages reçus ou émis par les composants étant identifiés comme présenté précédemment, il est possible d'utiliser les modèles d'erreur. Tout d'abord, avant de considérer les défaillances du service rendu par les composants, il est possible d'identifier des erreurs de type E.1, consistant en l'envoi de messages non prévus au sein d'une interaction. Par exemple, cela correspond aux défaillances mécaniques résultant en des projections, des explosions, des fouets, des décharges électrostatiques, etc. Ce type d'erreur dépend entièrement de la catégorie du composant physique (une poulie ne produira pas de décharge électrostatique par exemple), et correspond à une défaillance n'ayant aucun rapport avec la fonction de l'objet.

Bien que l'on puisse appliquer pratiquement tous les modèles d'erreur que nous avons proposés, les types d'erreur E.2, E.3 et E.4 ne sont pas applicables car, dans la plupart des cas, les composants mécaniques ne reçoivent ou n'émettent qu'un seul type de message. Il en est de même du type E.9 puisque aucune valeur n'est retournée par un dispositif mécanique (en considérant que les capteurs sont des dispositifs électroniques). En revanche, les erreurs liées au temps E.5 et E.10, sont envisageables pour certains composants dont le temps de réponse par exemple peut être critique pour la sécurité. C'est notamment le cas pour des composants comme les freins électromagnétiques par exemple.

Le cas le plus courant est sans doute l'erreur de type E.8 concernant la valeur des arguments envoyés par un message. Ce mode de défaillance peut en effet être assimilé à une transmission d'énergie mécanique erronée par rapport à ce qui est attendu. Les cas les plus courants sont :

- une énergie nulle (rupture de la chaîne cinétique) ;
- une énergie constante (blocage par exemple) ;
- un pic (discontinuité) ;
- une dérive (pièces qui se desserrent)
- une énergie trop faible ou trop forte.

Analyse des modes de défaillance

L'approche que l'on propose se base sur la notion de message afin de pouvoir utiliser les modèles d'erreur. Cependant, l'analyse des modes de défaillance peut se faire grâce aux tableaux de l'AMDEC, en introduisant les noms des composants (et non des messages) lorsque le service se réduit à un seul message, et dont le nombre d'erreur est réduit.

Le fait d'analyser les erreurs des messages échangés entre les composants et non pas les erreurs des composants eux-mêmes contribue à amener une dimension système à l'analyse. Par exemple, nous travaillons au laboratoire LESIA sur des actionneurs pneumatiques, les muscles artificiels présentés dans le chapitre 2, qui peuvent être étudiés suivant l'analyse classique de l'AMDEC mais aussi suivant notre démarche. Dans une analyse classique, on identifie à partir du composant un ensemble de modes de défaillance du type *crevé* (fuite d'air) ou *rupture totale* (rupture des fibres). Les causes identifiées sont du type mauvaise conception, mauvaise maintenance, ou mauvaise utilisation, etc. Cette approche concerne l'analyse qu'un mécanicien pourrait faire pour étudier les aspects concernant essentiellement le muscle artificiel. Mais dans le cadre d'une application concrète et de l'utilisation de ces muscles, nous proposons d'identifier le service principal d'un muscle qui est de convertir une énergie pneumatique en énergie mécanique. Les modes de défaillance sont alors différents, on peut identifier suivant les modes d'erreur proposés précédemment :

- énergie transmise trop faible / trop forte (erreur E.8),
- énergie transmise nulle ou constante (erreur E.8),
- dérive de l'énergie transmise (erreur E.8),
- conversion trop lente (erreur E.10) .

Dans ce cas, ceux qui précédemment étaient identifiés en tant que *modes de défaillance* deviennent les causes (la crevaison, la rupture, etc). L'AMDEC permet alors de se poser les questions relatives à l'utilisation des muscles pour cette application (le temps de réponse est-il suffisant ? Les exigences de performances sont-elles respectées ? Etc.).

4.3.4.5 Analyse des modes de défaillance du logiciel

Modélisation des objets et des interactions

Dans les systèmes informatiques comme les robots médicaux, il existe des composants à part, ce sont les sous-systèmes incluant du logiciel. Sur la figure 4.18, l'objet *Système de contrôle* est celui qui contient le logiciel de l'application comme cela est présenté sur la figure 4.21. Le modélisation du composant *Logiciel de l'application* peut alors être réalisée en utilisant l'ensemble des diagrammes UML, et notamment les diagrammes de séquence. Les messages échangés peuvent être des signaux ou des appels de service (opérations) comportant des arguments et parfois des contraintes de temps. Cet objet est critique dans la plupart des cas, car il peut être à l'origine de nombreux dommages.

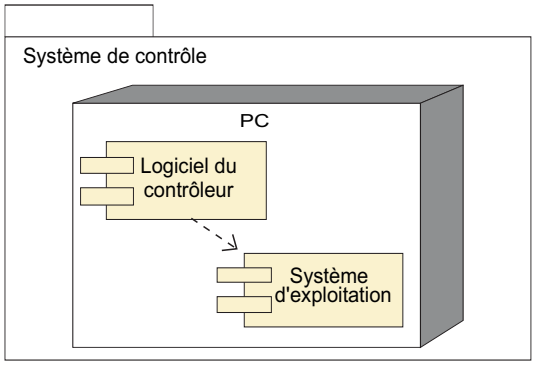

FIG. 4.21 – Package représentant les éléments du *Système de contrôle* de la figure 4.18

Types d'erreur et analyse des modes de défaillance

Il est possible d'instancier l'ensemble des modèles d'erreur pour les messages échangés entre les objets informatiques. Cependant, deux contraintes principales propres au logiciel apparaissent. Tout d'abord, alors que pour les dispositifs électroniques et mécaniques il est possible d'estimer la probabilité de la défaillance d'un message (en général liée au composant lui même et à ses connexions), cela est en revanche impossible pour un objet informatique. Cette défaillance étant due à une erreur de programmation, il est impossible d'en prévoir la probabilité d'occurrence. Et ensuite, l'importance du nombre de messages échangés au sein d'un logiciel rend toute analyse systématique très difficile.

La norme générique IEC 61508 (2001), concernant les dispositifs électroniques programmables, aborde ce problème en introduisant la notion de SIL (*Safety Integrity Level*). Nous détaillons ce concept car il semble qu'aujourd'hui cette norme soit la plus utilisée dans le domaine des dispositifs électroniques programmables. Le SIL est un niveau (chiffré de 1 à 4) attribué à un logiciel, qui est considéré comme un des composants du système (composant boîte noire), en fonction des dommages qu'il peut induire pour le système. Puis, en fonction des choix d'architecture de ce logiciel et les séparations possibles en différents modules logiciels indépendants, un niveau SIL est attribué à chacun de ces modules. L'intégration de techniques permettant de modifier l'architecture de ces modules comme la redondance, les voteurs[3], etc., permet d'attribuer de nouveaux SIL aux différents modules. Cette norme propose ensuite d'utiliser, en fonction du SIL, des tableaux permettant de choisir des techniques de prévention (langages de modélisation), d'élimination (vérifications, validations, tests, etc.) et de tolérance aux fautes, en donnant pour chaque technique un critère de sélection :

- Non Recommandé (NR) ;
- Recommandé (R) ;
- Hautement Recommandé (HR).

[3]Les voteurs sont des sous-systèmes qui effectuent un choix entre plusieurs sorties possibles calculées par des sous-systèmes redondants

A défaut de pouvoir estimer les risques de certaines constructions, cette norme se concentre sur la manière de construire le système, et considère le logiciel comme un composant. Dans le cas des robots de service ayant des contacts avec les humains, le SIL du logiciel sera toujours au maximum (en l'occurrence 4), et les techniques seront donc toujours les mêmes.

Les constructions UML basées sur la notion d'objet permettent de modéliser un logiciel en modules indépendants qui correspondent aux *composants* dans la terminologie d'UML. Cependant, les notions même d'objet et de classe ne peuvent être évaluées en terme de criticité. C'est le service délivré par le système, induit par la collaboration des différents objets, qui peut être qualifié de critique, et pas les objets. Cette différence fondamentale rend impossible l'application directe de la notion de SIL à un logiciel développé en UML. De plus, tous les aspects humains et mécaniques ont été volontairement écartés de la norme IEC 61508 (2001). Cependant, en considérant les messages échangés entre les objets, il est possible d'identifier ceux dont les erreurs pourraient induire plus ou moins de dommages. Il est alors possible d'attribuer un niveau de criticité, proche du concept de SIL, à chaque message. Cependant, en raison du très grand nombre de messages échangés au sein d'un logiciel, il semble très difficile de pouvoir effectuer une analyse complète du logiciel. Afin d'effectuer tout de même une analyse, il est possible de se restreindre à différentes catégories de messages (liste non exhaustive) :

- messages entre objets instanciés à partir de classes dites actives (celles qui possèdent leur propre fil d'exécution),
- messages émis ou reçus par les interfaces,
- messages annotés par des contraintes de type QoS,
- messages liés à la surveillance des capteurs et des actionneurs (détection, signalisation, gestion des défauts), éventuellement à l'auto-surveillance du logiciel.

Il est évident que pour tout système où il est impossible d'effectuer une analyse complète, le choix des messages qui seront analysés dépendra du concepteur. Cependant, en suivant cette liste et en appliquant les modèles d'erreurs présentés dans la section 4.3.3.2, il sera possible de guider les choix de conception du logiciel en fonction de la criticité de certains messages.

Comme pour les sections précédentes, les tableaux de l'AMDEC sont utilisables, en enlevant la colonne *cause* qui ne concerne que les erreurs humaines de conception. L'aspect probabilité du risque peut aussi être retiré, mais il est important de conserver la notion de gravité. L'analyse des risques induits par le logiciel est l'objet de nombreuses études du fait de sa complexité. Et le fait que l'on propose dans notre démarche de se limiter à des messages « critiques », montre bien toute la subjectivité qui peut exister dans une telle analyse. L'approche que l'on propose ici ne permet donc pas de donner une méthode exhaustive, elle se base comme la majorité des activités de l'analyse du risque sur l'expertise du concepteur. La complexité des systèmes robotiques actuels, impose l'utilisation de techniques non formelles, et qualitatives comme celle que nous proposons.

4.3.5 Analyse des arbres de fautes

L'analyse des arbres de fautes est une méthode inverse de l'AMDEC, puisque l'on part d'un danger et on remonte aux événements source qui peuvent être des défaillances (des erreurs), ou d'autres événements. Il s'agit d'une technique dite de *backward*. En complément de l'AMDEC, l'analyse des arbres de fautes permet de prendre en compte les défaillances multiples, ainsi que les interactions entre les fautes de différents domaines (électronique, informatique, mécanique, et erreurs humaines). De plus, elle permet d'identifier des situations dangereuses résultant de l'interaction de fonctionnements normaux non défaillants (comme un interblocage par exemple). Un des intérêts de l'utilisation des arbres de fautes en complément de l'AMDEC est de reconsidérer le système en changeant de point de vue. C'est la richesse des différents points de vue qui permettra d'aborder le problème dans son intégralité[4]. Un arbre de faute permet d'illustrer la terminologie présentée au chapitre 1, puisque l'on peut retrouver dans la combinaison de ces événements des phénomènes dangereux (défaillances, états dangereux, et des propriétés inhérentes dangereuses), des situations dangereuses et des événements dommageables.

Le lien se fait naturellement avec les tableaux de l'AMDEC puisque des nombreux événements sont identifiés dans les colonnes des modes de défaillance et des effets. Cependant, l'analyse des arbres de fautes ne peut s'appuyer directement sur la notion de message, car un arbre est constitué d'événements qui peuvent être notamment :

- des événements non modélisables en UML (par exemple *Inattention de l'opérateur*),
- des situations plus complexes qu'un simple état et non modélisables en tant que message (par exemple *Un opérateur pénètre dans la zone de travail du robot*),
- un état d'un sous-système (comme *Robot en mouvement*).

L'utilisation que l'on en a faite s'est donc restreinte à l'étude d'un cas particulier, celle de la défaillance de l'interaction prévue entre le robot et l'humain.

Pour partir de ce danger, appelé élément racine de l'arbre, résultant d'une interaction négative entre un acteur et le système technologique, il est possible de s'appuyer sur une modélisation de cette interaction en UML. En effet, il est possible de spécifier cette interaction grâce à un diagramme de classes comme celui de la figure 4.22. Dans ce cas, l'interaction qui nous intéresse est les déplacements de la sonde sur le patient. Pour déterminer l'arbre de fautes, on peut partir d'une interaction néfaste entre la sonde et le patient, et remonter les composants jusqu'au contrôleur. Les événements ici sont génériques, mais il est possible de détailler chaque événement en précisant les défaillances (la défaillance robot pouvant être une pièce qui tombe, un axe qui est bloqué, etc.). Une partie de cette analyse est donnée figure 4.23.

À partir du diagramme de classes de la figure 4.22, on identifie donc les composants pouvant mener au danger racine. Cependant, il existe d'autres interactions non modélisables

[4]Cependant les points de vue concernent la manière dont on décrit un système, et ce n'est pas la multiplication de techniques ou de langages utilisés qui augmentent le nombre de points de vue. Comme dans UML, on peut utiliser le même langage pour exprimer plusieurs points de vue (voir section 3.2.3.3).

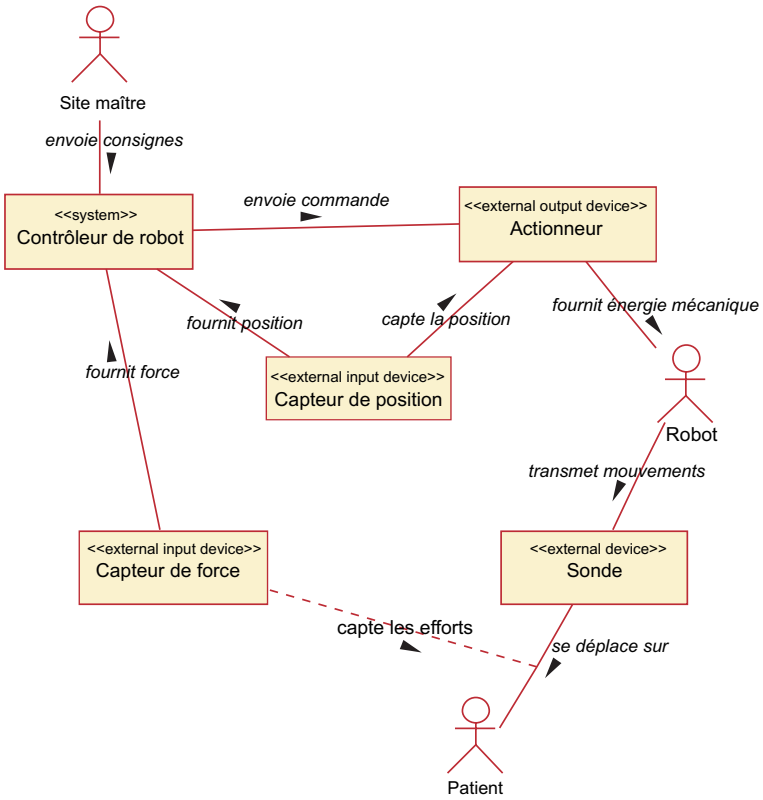

FIG. 4.22 – Diagramme de classes d'un robot manipulateur d'une sonde quelconque pour l'examen d'un patient

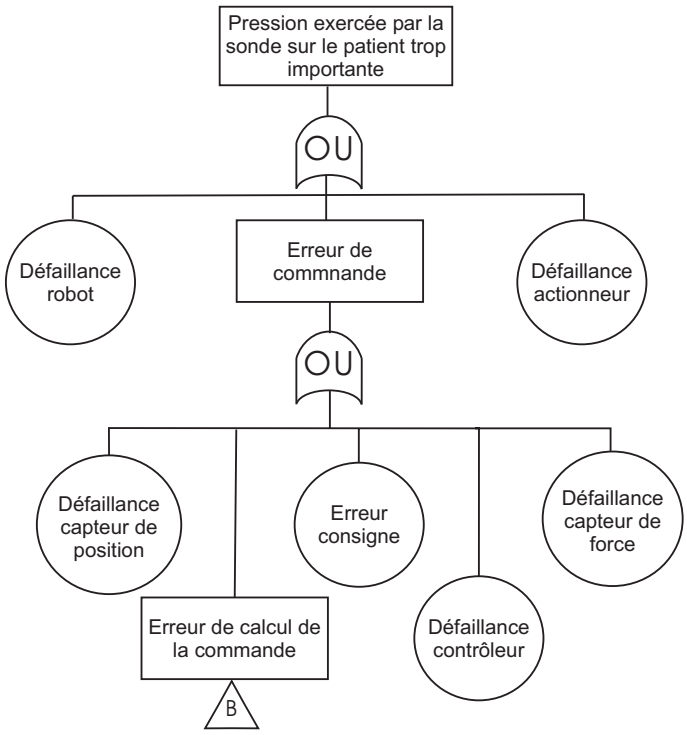

FIG. 4.23 – Exemple d'arbre de fautes pour un effort de la sonde trop important sur le patient

comme un pièce projetée sur le patient, des transferts d'énergie non prévus, etc. Ces cas restent tout de même moins complexes car ils résultent de l'effet direct d'une défaillance sur les humains dans l'environnement du robot, et l'utilisation des arbres de fautes apporte une synthèse des éléments de l'AMDEC correspondants.

L'étape suivante d'analyse correspond à une utilisation classique des arbres de fautes, qui consiste à identifier des *coupes minimales* correspondant à un ensemble de fautes directement reliées à l'élément racine (Leveson, 1995). Puis chaque événement est traité par des techniques d'élimination, de prévention et de tolérance aux fautes. Ces informations sont communes aux résultats de l'AMDEC, mais on obtient une vue plus synthétique du risque lié à un danger.

4.4 Conclusion

La démarche proposée s'intègre dans un processus de développement utilisant la notation UML. Les différentes étapes peuvent être complétées de manière continue, et suivre un processus itératif et incrémental. Pour cela, il suffit de raffiner les différents tableaux de l'analyse préliminaire et de l'AMDEC, puis de compléter les arbres de fautes avec les défaillances des mécanismes et des composants introduits au fur et à mesure des itérations. Il existe donc en permanence des relations entre le processus de développement et l'analyse du risque. On peut modéliser de façon très simple le processus comme présenté sur la figure 4.24, et pour la première itération suivre ces étapes :

- modéliser le métier en analysant chaque cas d'utilisation et en se concentrant sur les échanges d'énergie, les mouvements effectués par les acteurs, les contacts entre acteurs et éléments d'interface avec la tâche ;
- intégrer le robot dans la modélisation du métier, allouer les tâches entre les acteurs et la technologie et détailler les fiches des cas d'utilisation (notamment les contraintes de sécurité et de performances) ;
- définir les frontières du système et décrire les tâches des acteurs avec des diagrammes de séquence ;
- identifier les principaux dangers par une analyse préliminaire et compléter les cas d'utilisation avec les résultats de cette analyse ;
- identifier les phénomènes dangereux et estimer les risques liés aux messages en utilisant l'AMDEC et en se basant sur des analyses faites par des spécialistes de chaque domaine ;
- synthétiser les différentes analyses de l'AMDEC avec des arbres de fautes en prenant en compte les défaillances multiples.

Cette démarche permet dans un premier temps de définir les tâches et les rôles des acteurs de manière non ambiguë grâce aux cas d'utilisation et aux diagrammes de séquence. Il est évident que les modèles ne peuvent suffire à eux seuls à la capture des besoins et des informations sur le système. Cependant, les modèles doivent toujours être le point de départ d'études

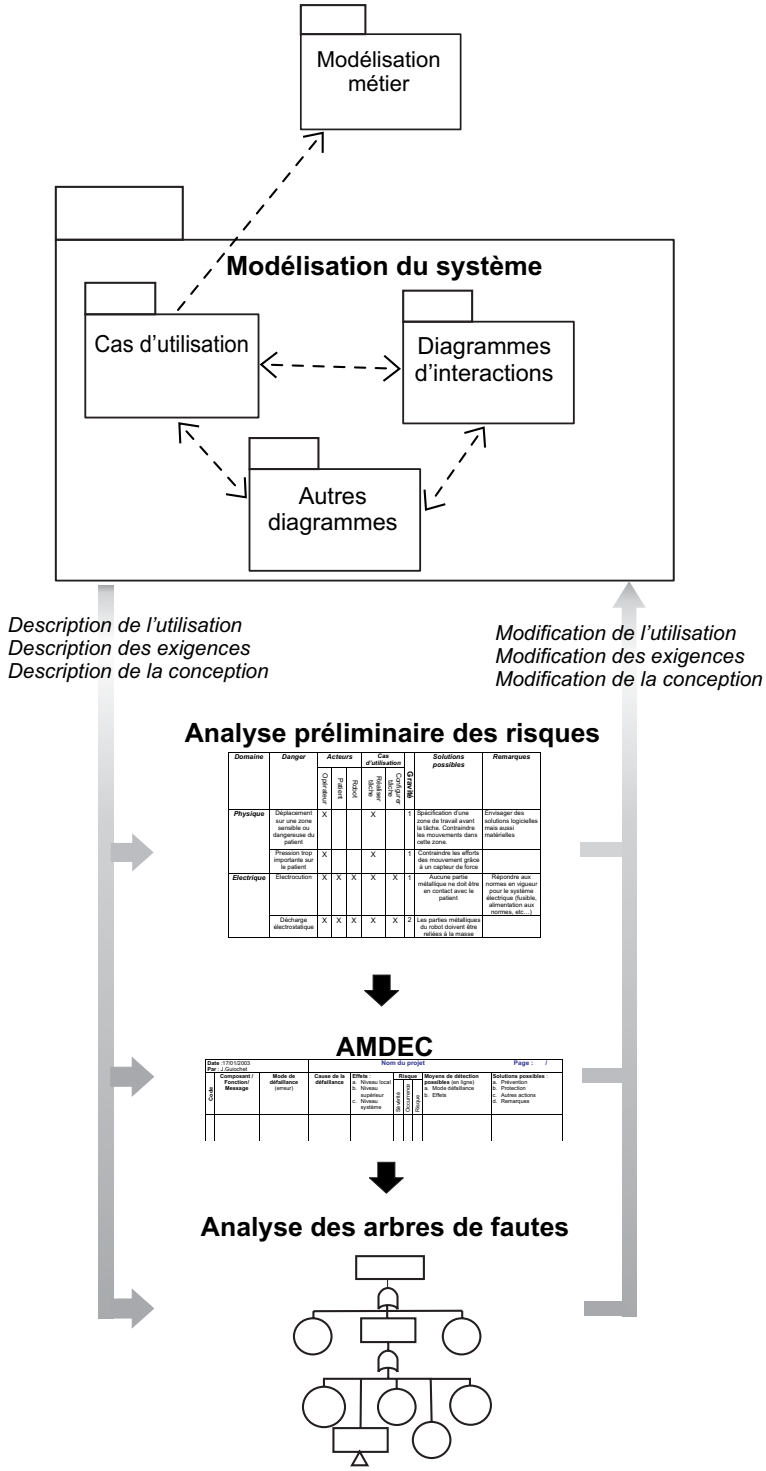

FIG. 4.24 – Résumé de la démarche présentée

plus pointues. Ainsi, un diagramme de classes des acteurs permet de décrire les interactions entre acteurs et les responsabilités de chacun, mais peut être accompagné de documents relatant les capacités mentales et physiques, les compétences ou la formation de chaque acteur. La documentation liée aux cas d'utilisation permet de donner un cadre aux différentes analyses effectuées dès le début d'un projet. Ainsi, il est possible de spécifier pour le cas d'utilisation générique *Réaliser tâche*, tous les échanges d'énergie critiques pour la sécurité et le type d'interactions entre le système et les humains. Il est alors fondamental d'accompagner les cas d'utilisation de leur description textuelle. De plus, il est possible de faire apparaître dans ces documents des contraintes de sécurité absente de la modélisation graphique. Enfin, il est important de souligner que les cas d'utilisation permettent de définir les frontières du système analysé, et concentrer ainsi les efforts des analystes.

L'apparition de dommages au sein d'un système n'est due qu'à sa propre dynamique. Et par conséquent, les dangers sont en partie liés à l'échange de messages au sein d'un système. En se basant sur la notion de *Message* en UML, nous avons présenté des modèles d'erreur liés à ce concept. Nous avons montré comme ces modèles étaient applicables à différents domaines d'études. De plus l'ensemble des exemples montre que ces modèles d'erreur peuvent être utilisés de manière générique à différentes applications. Le lien a donc pu être établi entre les concepts objet, et la technique de l'AMDEC pourtant dédiée à l'origine à l'analyse fonctionnelle. Notons que la terminologie présentée au chapitre 1 s'adapte à ces différentes techniques, et l'on a notamment illustré certains concepts au sein des tableaux de l'AMDEC. Cependant, un tel travail n'a pu être mené avec l'analyse des arbres de fautes, dont l'utilisation ne prend en compte qu'un type de danger et un effort doit être réalisé dans ce sens.

L'utilisation combinée des techniques de l'AMDEC, des arbres de fautes et d'UML, permet d'avoir véritablement une analyse système. Certains diagrammes du langage UML permettent en effet de représenter les interactions entre les humains et différents composants électroniques ou mécaniques, et les arbres de fautes regroupent les défaillances de tous les domaines. Au sein des tableaux de l'AMDEC, les modes de défaillance sont propres à un domaine, mais les effets identifiés ainsi que les solutions proposées couvrent tous les domaines, dont les facteurs humains. Ce dernier aspect s'il n'est pas évoqué explicitement lors d'une étape apparaît tout au long de la démarche. En effet, la description du système inclut les activités d'allocation et d'analyse des tâches, et l'identification des dangers inclut l'analyse des erreurs humaines. Il est alors possible de regrouper ces activités (Guiochet et al., 2003b) pour souligner l'aspect novateur de cette démarche, mais elles doivent être intégrées tout au long du processus.

Un dernier point relatif aux exigences que l'on s'était fixées au départ, concerne l'analyse du risque du logiciel. Cette problématique, étant traitée actuellement par plusieurs normes, est encore sujette à de nombreux travaux. La solution que l'on propose dans ce chapitre repose en partie sur l'expertise du concepteur et sa capacité à identifier les messages critiques, ce qui ne peut être ni exhaustif ni vérifiable. Cependant, pour une démarche système, le fait de considérer le logiciel comme un composant à part entière permet déjà de réduire les principaux

risques. L'analyse plus pointue du logiciel, et des messages échangés entre ses objets, est un domaine ouvert à de nombreuses recherches.

La démarche proposée semble répondre aux différentes exigences identifiées au premier chapitre. On dispose d'une analyse du risque système, incluant les facteurs humains, reposant sur des techniques connues des ingénieurs, intégrant les analyses des défaillances du logiciel, et favorisant la traçabilité par l'utilisation du langage UML. Bien que certains aspects soient limités dans leur présentation théorique, nous proposons dans le chapitre suivant une application de cette démarche à un système de robot médical pour la télé-échographie.

Chapitre 5

Application de l'analyse du risque au développement en UML d'un robot télé-échographe

5.1 Introduction

Ce chapitre consiste en l'application de la démarche proposée au chapitre 4 et à l'utilisation des muscles artificiels présentés dans le chapitre 2, à un robot médical pour la télé-échographie. Des membres du LESIA sont intervenus à titre d'experts pour la fabrication et la commande des muscles artificiels, ainsi que pour une analyse de la sécurité du site esclave. Ce travail a pu être réalisé grâce un travail en collaboration avec Adriana Vilchis de l'équipe GMCAO du laboratoire TIMC/IMAG[1]. Le lecteur pourra trouver de nombreuses informations relatives à la conception de ce système dans son mémoire de thèse (Vilchis, 2003).

Bien que la démarche proposée précédemment s'applique dès le début du processus de développement, nous n'avons pu intervenir qu'à une étape avancée du développement du site robotisé. Il n'existait alors aucune modélisation du système et le processus de développement ne suivait aucune méthode spécifique. Par ailleurs, il n'existait pas d'étude détaillée concernant la sécurité. Ceci était principalement dû au fait que ce développement se faisait dans le cadre d'un projet de recherche, et que le premier objectif était de valider des choix de conception du robot et de sa commande, et de prouver la faisabilité d'un tel projet en vue d'une version ultérieure.

Le travail a donc consisté à synthétiser et à modéliser en UML les résultats de différentes étapes du processus de développement. Puis à appliquer la démarche d'analyse du risque basée sur ces modèles. Les résultats de cette analyse ont servi à redéfinir certaines exigences et à modifier certains points de la conception. Cependant, l'état d'avancement du projet a rendu impossible certaines propositions qui pourront être implantées sur une prochaine version.

[1] http ://www-timc.imag.fr/gmcao

Les sections de ce chapitre suivent les étapes présentées au chapitre 3 concernant l'analyse du risque. Ainsi, après une présentation du projet TER dans la section 5.2, la section suivante (5.3) aborde la description du système et de son utilisation. La section 5.4 étudie ensuite l'identification des phénomènes dangereux et l'estimation du risque.

5.2 Présentation du projet TER

L'échographie est un examen très fréquent dans le milieu médical, notamment lors d'une grossesse, pour vérifier l'état de santé du fœtus et de la mère. Le diagnostic repose sur la confrontation entre la lecture des images de l'échographie et la connaissance des déplacements de la sonde sur le ventre de la patiente. Pour cela, cet examen dépend entièrement de l'expertise du médecin, et demande un long apprentissage et des connaissances très spécifiques. Cet examen est donc difficile d'accès pour les patients se trouvant loin de grands centres hospitaliers.

La télé-médecine, grâce aux télécommunications, permet à des spécialistes de réaliser de tels gestes, en se trouvant dans une région différente de celle du patient. Il est donc possible d'appliquer ce concept à l'examen d'échographie comme cela a été proposé par de nombreux projets. Vilchis et al. (2001b) proposent une synthèse de différents projets.

Le projet TER concerne la Télé-Échographie Robotisée et se place dans le cadre plus général de la télé-médecine. Il devrait permettre à un spécialiste d'ausculter à distance une patiente, grâce à un système robotisé permettant de réaliser les déplacements de la sonde sur la patiente. C'est un projet regroupant plusieurs industriels (France Telecom, SINTERS, PRAXIM), hôpitaux (CHU Trousseau, CHU de Grenoble) et laboratoires (TIMC/IMAG, LVR) français (Vilchis et al., 2001a), et financé par le Ministère de l'Éducation Nationale, de la Recherche et de la Technologie en 1999.

Le système proposé pour la télé-échographie consiste en un site maître et un site esclave. Le médecin spécialiste se trouvant sur le site maître, déplace une sonde virtuelle et consulte les images produites par les déplacements de la sonde réelle sur le ventre de la patiente se trouvant sur le site esclave. L'architecture générale du système est représentée sur la figure 5.1. Les informations échangées entre les sites concernent les images de l'échographie, les données liées aux déplacements du robot, les forces exercées par la sonde (données haptiques), et des informations audiovisuelles concernant les opérateurs des deux sites. Cette architecture a été choisie par les différents partenaires avant d'effectuer l'analyse présentée par la suite. Elle correspond à une solution qui découle de nombreuses études menées par les consortium TER dont les documents ne sont pas tous accessibles. Cependant nous reviendrons sur cette solution pour justifier les choix qui ont été faits.

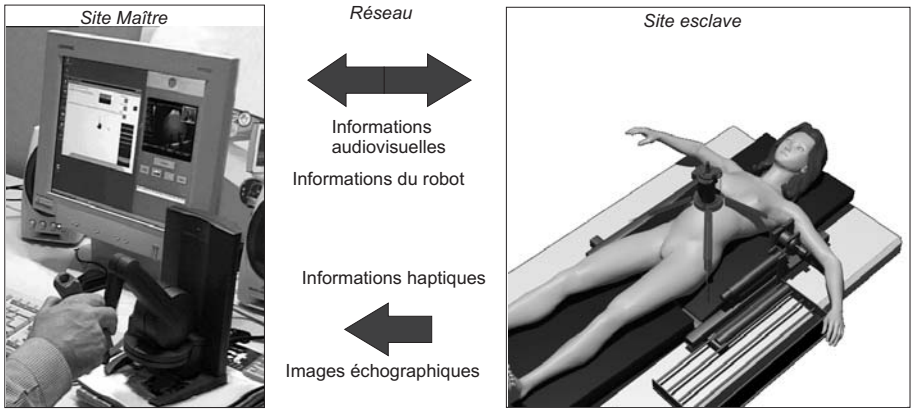

FIG. 5.1 – Vue générale du système TER

5.3 Définition du système et de son utilisation

Comme présenté dans le chapitre 3, la définition du système constitue la première étape d'une analyse du risque. Elle concerne successivement la modélisation du métier (section 5.3.1), l'introduction du robot dans le métier avec une allocation des tâches (section 5.3.2), la définition des frontières de l'analyse (section 5.3.3), puis une analyse des tâches (section 5.3.4).

5.3.1 Modélisation du métier : l'examen échographique

L'observation d'examens échographiques, la consultation d'experts, l'interview des différents acteurs, et l'observation de documents vidéo, ont abouti sur un concept de dispositif technologique de télé-échographie original mais, sans avoir réellement pris en compte l'ensemble du système comprenant les acteurs. La première étape de notre démarche, qui correspond à ce qui a été fait par les équipes du consortium TER, est l'observation du métier de l'échographie sans introduire le système robotique. En donnant un cadre par une modélisation, on assure que les différents éléments ont bien été appréhendés et seront dérivés pour la suite du développement. La détermination de ces cas d'utilisation découle d'un travail d'identification des acteurs et des tâches effectuées, puis de synthèse pour exprimer les relations avec chaque acteur. Le point de vue que l'on prend ici est celui de l'acteur principal, le spécialiste médical, que l'on nommera par la suite *Spécialiste*. Un autre acteur est évidemment le *Patient*. Nous avons choisi de représenter le *Système d'échographie*, aussi appelé échographe, comme un acteur au sens où le *Spécialiste* interagit avec ce système et modifie ses actions en fonction des résultats fournis par le système. De plus le système d'échographie est fourni par des constructeurs et peut être modélisé avec le diagramme de déploiement de la figure 5.2, en représentant les deux composants principaux : la sonde et la station de contrôle. Les acteurs et leurs relations peuvent être exprimés avec un diagramme de classes comme sur la figure 5.3. On identifie ainsi lors de l'examen échographique quatre cas d'utilisation représentés sur la figure 5.4 :

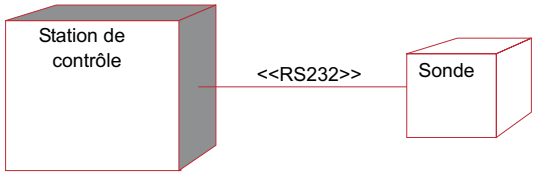

FIG. 5.2 – Diagramme de déploiement du système d'échographie

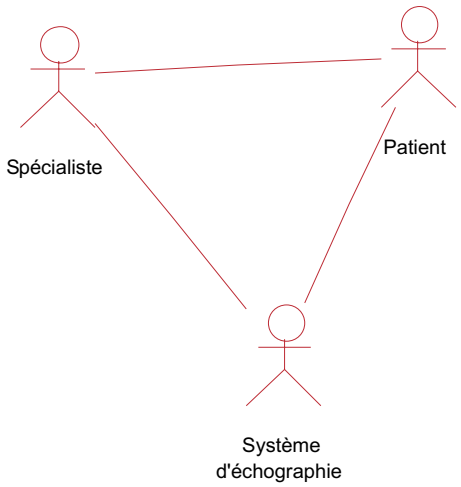

FIG. 5.3 – Diagramme de classes des acteurs de l'examen échographique

- *Établir un diagnostic* ;
- *Réaliser le geste médical* ;
- *Gestion du système d'échographie* ;
- *Gestion patient*.

Cette représentation fournit un cadre pour l'observation et la collecte d'informations relatives au métier. Il est en effet possible de noter grâce aux descriptions textuelles des cas d'utilisation, toutes les informations concernant l'examen. S'il existe une information ne pouvant rentrer dans un cas d'utilisation, il y a de fortes chances pour que le diagramme des cas d'utilisation soit incomplet. Un autre point qui est fondamental pour la modélisation en cas d'utilisation est le fait qu'ils doivent pouvoir s'exécuter en parallèle, ce qui est bien respecté ici. Les descriptions textuelles des cas d'utilisation sont données ci-dessous en utilisant les fiches présentées dans le chapitre précédent (4.2).

Cas d'utilisation	**Établir un diagnostic**
Acteurs	Spécialiste
Description	Le *Spécialiste* établit un diagnostic en confrontant les images fournies par le système d'échographie et la connaissance des déplacements de la sonde sur le *Patient*
Pré-conditions	Les images sont affichées et le *Spécialiste* connaît les mouvements effectués
Post-conditions	Le *Spécialiste* informe le *Patient* du diagnostic

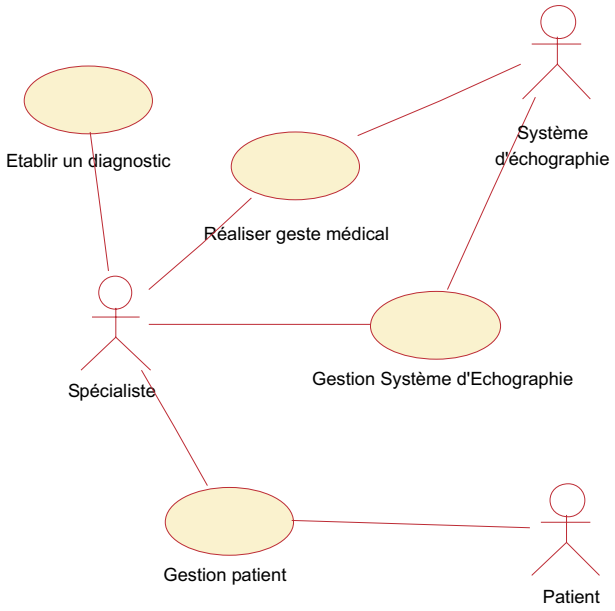

FIG. 5.4 – Diagramme des cas d'utilisation de l'examen échographique

Cas d'utilisation	**Gestion du système d'échographie**
Acteurs	Principal : Spécialiste Secondaire : Système d'échographie
Description	Le système d'échographie fournit les images au *Spécialiste*. Le *Spécialiste* modifie à tout moment les réglages du Système d'échographie et notamment les paramètres des ultrasons. Il utilise certains outils disponibles de traitement de l'image (calcul de longueur, agrandissements, etc.)
QoS	La gestion du système d'échographie doit pouvoir être réalisée tout en effectuant les déplacements de la sonde
Contraintes techniques	Les systèmes d'échographie utilisés sont certifiés et les interfaces sont standardisées

Cas d'utilisation	**Gestion patient**
Acteurs	*Spécialiste* et *Patient*
Description	Le *Spécialiste* installe et désinstalle le *Patient* et répond à ses questions. Il recouvre la surface qui sera en contact avec la sonde avec du gel permettant une meilleure conduction des ultrasons et facilitant les déplacements. Il délivre au *Patient* par voie orale le résultat du diagnostic (sachant que le diagnostic peut être une impossibilité d'interprétation). Le *Patient* répond aux questions du *Spécialiste* ou émet des requêtes.
QoS	Il s'agit de mettre en confiance le *Patient* et l'installer de façon à limiter le stress. La dose de gel doit être suffisante et présente partout où la sonde se déplacera

Cas d'utilisation	**Réaliser le geste médical**
Acteurs	Principal : Spécialiste Secondaire : Système d'échographie
Description	Le *Spécialiste* réalise le geste de l'échographie en déplaçant la sonde sur la surface recouverte de gel. Il adapte ses mouvements en fonction des images fournies par le système d'échographie. Les mouvements de la sonde sont : – des translations dans le plan liées à la surface du *Patient*, – des rotations suivant les trois axes x, y, et z référencés par rapport à la sonde, – une translation orthogonale à la surface pour augmenter la pression. Une étude détaillée de ces mouvements est fournie dans les thèses de Guerraz (2002) et Vilchis (2003).
Pré-conditions	Le système d'échographie est connecté et fonctionne correctement. Le *Patient* est installé et la surface pour l'auscultation est recouverte de gel.
Alternative	Le *Spécialiste* peut stopper à tout moment le geste et retirer la sonde du *Patient* puis : – il peut ensuite reprendre l'examen, – il peut alors remettre du gel et reprendre le geste, – il peut arrêter l'examen.
QoS	La pression sur le ventre ne dépasse pas 1,5 daN (effort maximum mesuré de 1,3daN (Guerraz, 2002)) sur une surface de 20cm^2 La vitesse de déplacement de la sonde est de l'ordre du centimètre par seconde Les déplacements ne sortent pas de la zone où le gel est apposé

5.3.2 Système robotisé pour la télé-échographie et allocation des tâches

À partir de la modélisation précédente, il est possible de spécifier les tâches des acteurs et du système, et de conserver ainsi une cohérence (donc réduire les risques d'erreurs) entre la tâche sans robot et celle de télé-médecine robotisée. L'utilisation d'un robot peut se résumer aux cas d'utilisation génériques présentés sur le diagramme de la figure 5.5. On définit ces cas d'utilisation ainsi :

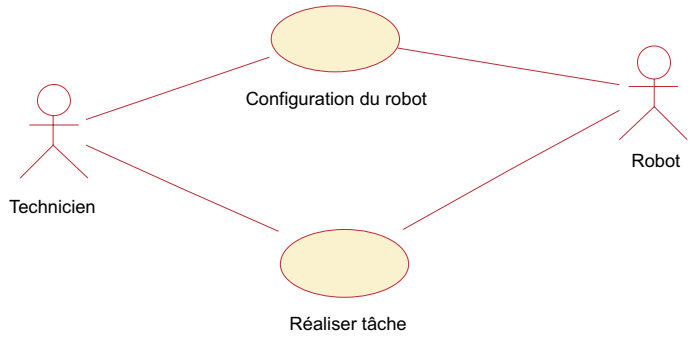

FIG. 5.5 – Diagramme des cas d'utilisation d'un robot

Cas d'utilisation	**Réaliser tâche**
Acteurs	*Technicien* et *Robot*
Description	Le *Robot* réalise une tâche en suivant les ordres du *Technicien*. L'*Opérateur* s'il le désire peut interrompre la tâche à tout moment.
Pré-conditions	Le contrôleur est configuré pour effectuer la tâche.
QoS	Les contraintes sont spécifiques à la tâche

Cas d'utilisation	**Configuration du Robot**
Acteurs	*Technicien* et *Robot*
Description	Le *Technicien* entre les données qui permettent de réaliser la tâche. Il installe et désinstalle certaines composantes du *Robot*. Il assure la maintenance du *Robot*.
Remarques	Les étapes d'installation et de configuration sont particulièrement critiques pour la sécurité.

L'intégration d'un système robotique dans une tâche d'échographie implique l'intégration des cas d'utilisation liés à l'utilisation d'un robot. Il faut donc intégrer les deux cas d'utilisation de la figure 5.5 au diagramme de la figure 5.4. Le fait d'introduire un système de télé-opération modifie le diagramme des cas d'utilisation de différentes manières :

– il existe maintenant deux points de vue suivant le site sur lequel on effectue l'analyse, le *Site maître* et le *Site esclave*. Le *Spécialiste* est localisé sur le *Site maître*. Nous nous intéressons pour la suite au *Site esclave* où la sécurité est critique ;
– l'acteur *Spécialiste* ne se trouve plus sur le site esclave, mais il est remplacé par l'acteur *Site maître* avec lequel le système communique. De ce fait, le cas d'utilisation *Établir un diagnostic* n'apparaît pas sur le diagramme des cas d'utilisation du site maître ;
– le choix a été fait d'introduire un sous-système de visio-conférence pour gérer la communication entre les deux sites. Les acteurs doivent alors gérer ce sous-système ;

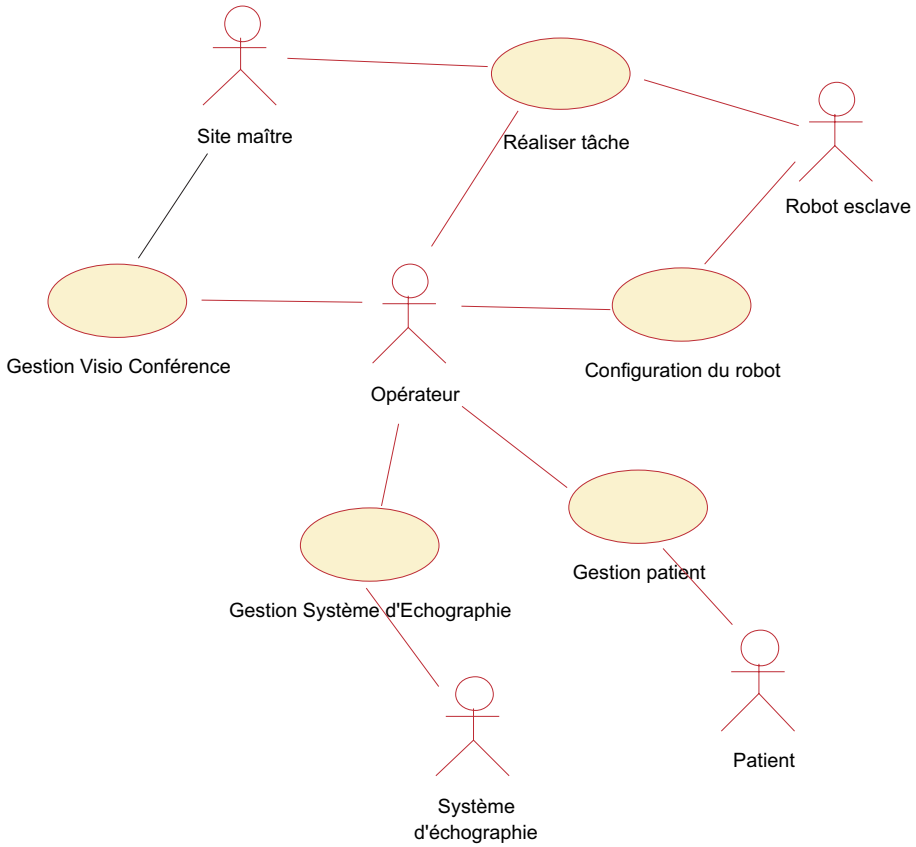

FIG. 5.6 – Diagramme des cas d'utilisation du site esclave

– le cas d'utilisation de la figure 5.4 nommé *Réaliser geste médical* devient *Réaliser tâche* (en considérant que la tâche effectuée par le *Robot* sera le geste médical qui consiste à déplacer la sonde échographique).

On identifie alors les différents cas d'utilisation de la figure 5.6, qui décrivent les rôles des acteurs, et la manière dont ils vont utiliser le système. Une description plus précise des tâches est effectuée dans la section suivante au travers des diagrammes de séquence. On définit ainsi un nouvel acteur nommé *Opérateur*, qui hérite d'une partie des aptitudes du *Spécialiste*, et du *Technicien*. Pour exprimer le fait qu'il existe un acteur pouvant être capable de gérer le *Patient* et de régler le système d'échographie, on peut introduire un nouvel acteur nommé *Assistant* n'ayant pas la formation pour effecteur le geste médical et le diagnostic. Le *Spécialiste* hérite donc de cet acteur car il possède toutes les caractéristiques de l'*Assistant* plus d'autre aptitudes (en UML on parle alors de spécialisation). Ces concepts sont modélisés sur le diagramme de classes de la figure 5.7. De l'analyse du diagramme des cas d'utilisation de la figure 5.6, on peut déjà conclure que le rôle de l'*Opérateur* sera important, car il concerne cinq cas d'utilisation. Ceci peut mener à se poser la question des capacités de l'*Opérateur* à réaliser l'ensemble des tâches qui lui seront assignées, et ainsi réduire la probabilité d'occurrence d'une erreur humaine due à une tâche trop complexe. Cette allocation, fondamentale pour la sécurité, doit faire l'objet d'un choix des concepteurs. Une autre solution aurait été de prévoir

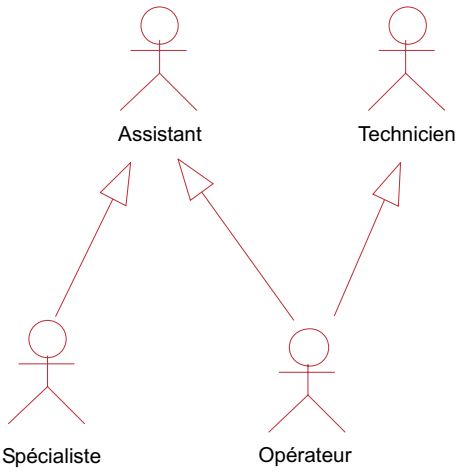

FIG. 5.7 – Diagramme de classes des acteurs

la présence sur le site esclave de deux acteurs, un *Assistant* et un *Technicien*. Ces deux acteurs sont décrits dans le diagramme de classes de la figure 5.7.

L'allocation des tâches effectuée ici résulte des étapes précédentes mais aussi de l'analyse des tâches que l'on effectue ultérieurement. La présentation des modèles est faite ici de manière séquentielle mais leur réalisation a suivi un processus itératif et incrémental.

5.3.3 Définition des frontières de l'analyse

Sur la base du diagramme des cas d'utilisation de la figure 5.6, il convient de définir les frontières du système que l'on souhaite analyser. Dans le cadre du projet global TER, l'ensemble des cas d'utilisation doit être traité. En ce qui concerne notre analyse guidée par l'analyse du risque, on choisit de se restreindre au système de contrôle du robot du site esclave. Les cas d'utilisation qui sont exclus alors du système de contrôle sont *Gestion Visio-conférence* et *Gestion Système d'échographie*. Ces cas d'utilisation concernent en effet deux sous-systèmes qui n'interviennent pas dans la réalisation de la tâche d'échographie. Il est alors possible de représenter ces deux sous-systèmes sous forme de *packages* comme cela est présenté sur la figure 5.8. Il est aussi intéressant de modéliser la répartition des acteurs au sein des *packages* comme cela est représenté. Le fait que le *package Visio-conférence esclave* utilise le *package Système d'échographie* exprime le fait que les images de l'échographie seront traitées en même temps que les autres données audio-visuelles envoyées sur le réseau. Afin d'alléger les diagrammes, nous n'utiliserons plus par la suite la notion de *Site esclave* qui sera implicite ; nous noterons *Système de contrôle TER* au lieu de *Système de contrôle du site esclave*. On en déduit alors le diagramme général des cas d'utilisation de la figure 5.9.

Les différents points suivants consistent à décrire chaque cas d'utilisation pour raffiner les tâches que les acteurs auront à effectuer. Cela permet aussi de définir l'ensemble des contraintes, dont certaines influent sur la sécurité, et de définir une base pour l'analyse du risque.

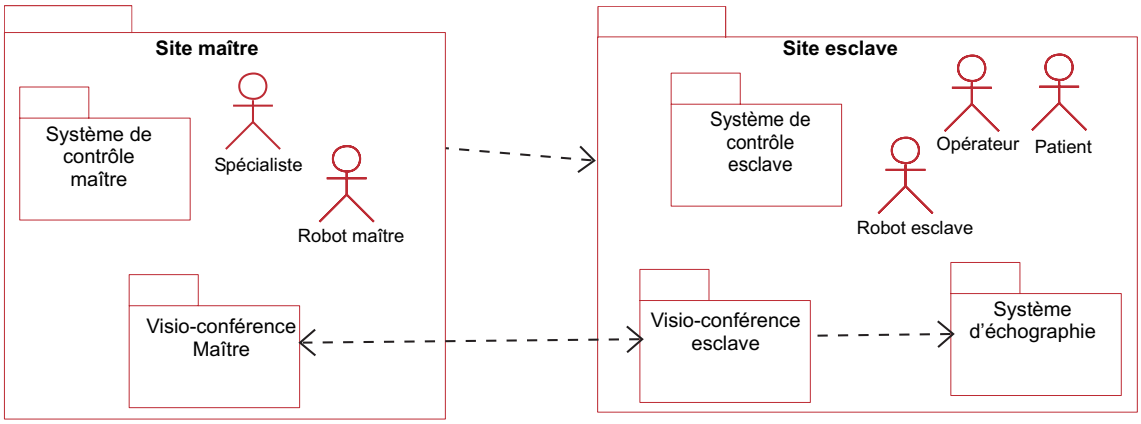

FIG. 5.8 – *Packages* et acteurs du système global

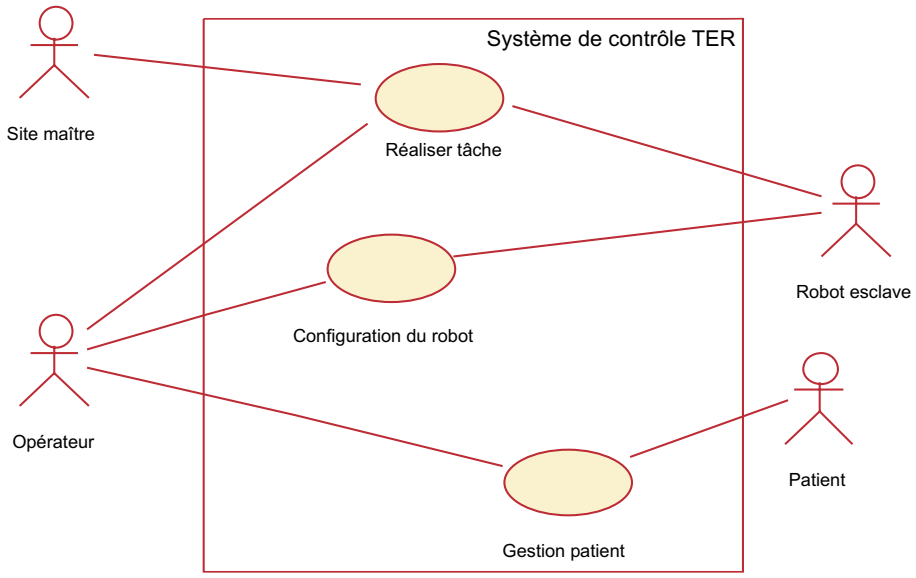

FIG. 5.9 – Diagramme des cas d'utilisation du système de contrôle esclave, que l'on note par la suite *Système de contrôle TER*

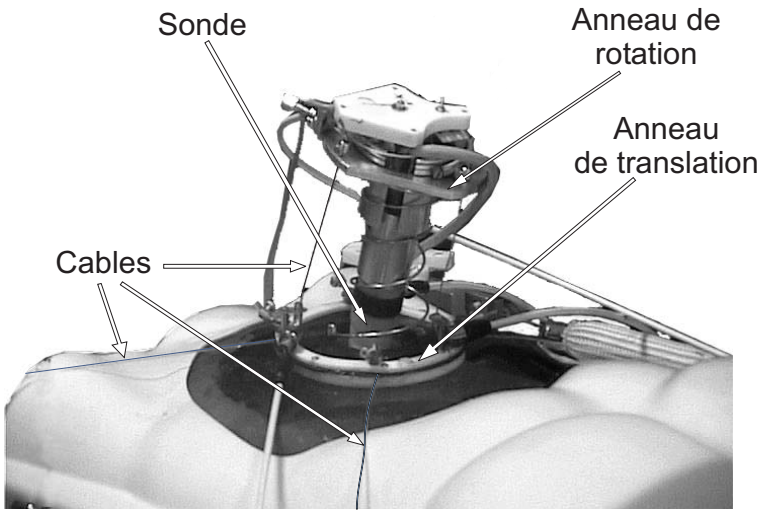

FIG. 5.10 – Vue de l'effecteur du robot esclave TER

5.3.4 Analyse des tâches

Pour chaque cas d'utilisation, nous montrons comment la modélisation guide certains choix de conception du système, puis comment les tâches des acteurs peuvent être analysées en utilisant les diagrammes de séquence.

5.3.4.1 Cas d'utilisation *Réaliser tâche*

La description de la tâche d'échographie en terme de mouvements et les différentes contraintes exprimées dans la partie 5.3.1, ont conduit les concepteurs à proposer une architecture robotique originale. Ils ont aussi pris en compte des contraintes d'encombrement réduit et de portabilité système en vue d'une utilisation dans des régions isolées. L'architecture conçue se différencie des structures de type bras manipulateur par une architecture *parallèle*, comme présentée sur la figure 5.10. Ce robot ne comporte donc aucune articulation en série, comme un bras articulé, et les mouvements se font par des variations de longueur des différents câbles reliés aux anneaux soutenant la sonde échographique. Une description plus détaillée est fournie par Vilchis et al. (2001a,b); Vilchis (2003).

Un autre point important de la conception est l'utilisation des muscles artificiels présentés dans le chapitre 2. Le choix de ces actionneurs a d'abord été fait dans un but de recherche, et non en suivant les résultats d'une analyse du risque. Nous montrerons par la suite ce qu'apportent ces actionneurs à la sécurité de l'examen échographique robotisé.

Le cas d'utilisation *Réaliser tâche* est principalement déclenché par le *Site maître*. C'est lui qui envoie les consignes de mouvements au site esclave, et ce dernier retourne les informations liées au déplacement réel du *Robot* sur le *Patient*. Une des exigences d'un tel système est la transmission au site maître des efforts exercés sur le patient. Le scénario principal du cas d'utilisation *Réaliser tâche* est représenté sur la figure 5.11. Ce cas d'utilisation très clas-

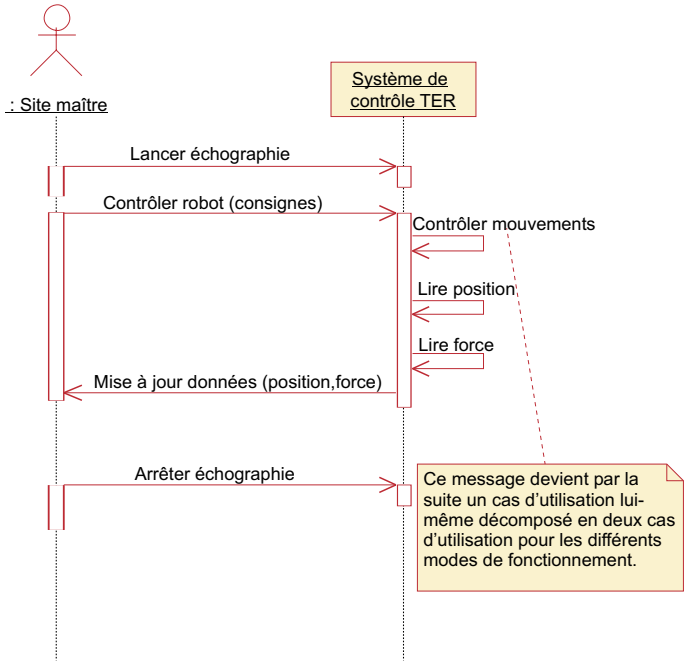

FIG. 5.11 – Diagramme de séquence du scénario principal de *Réaliser tâche*

sique en robotique inclut très souvent trois cas d'utilisation présentés par Carroll (1999), et représentés sur le diagramme des cas d'utilisation de la figure 5.12 :

- la planification et/ou l'interprétation des tâches ;
- le contrôle des mouvements ;
- le contrôle de l'outil.

Dans le cas du robot esclave TER, il n'y a pas de planification de tâche à effectuer puisque le site esclave reçoit directement des ordres de mouvements. Cependant, il est possible de conserver un cas d'utilisation nommé *Interpréter tâche*, notamment pour interpréter les changements de modes de déplacement exposés ultérieurement. De plus, l'absence d'*outil* sur le robot esclave TER nous conduit à supprimer le cas d'utilisation *Contrôle de l'outil*.

Afin de pouvoir répondre aux exigences liées à la réalisation du geste médical, deux modes de fonctionnement ont été identifiés :

- un mode de télé-opération continu : les consignes sont envoyées avec une période fixée à 500 ms ;
- un mode de suivi de trajectoire : le site maître envoie la position initiale, la position finale, la vitesse du déplacement et l'accélération, puis le site esclave génère et contrôle les mouvements suivant cette trajectoire.

En considérant que l'analyse est pour l'instant de très haut niveau, et comme le nombre de cas d'utilisation est très réduit, il est possible de créer deux cas d'utilisation qui héritent du cas d'utilisation *Réaliser tâche* et qui représentent les deux modes de fonctionnement : *Contrôler mouvements en télé-opération* et *Contrôler mouvements en trajectoire générée*.

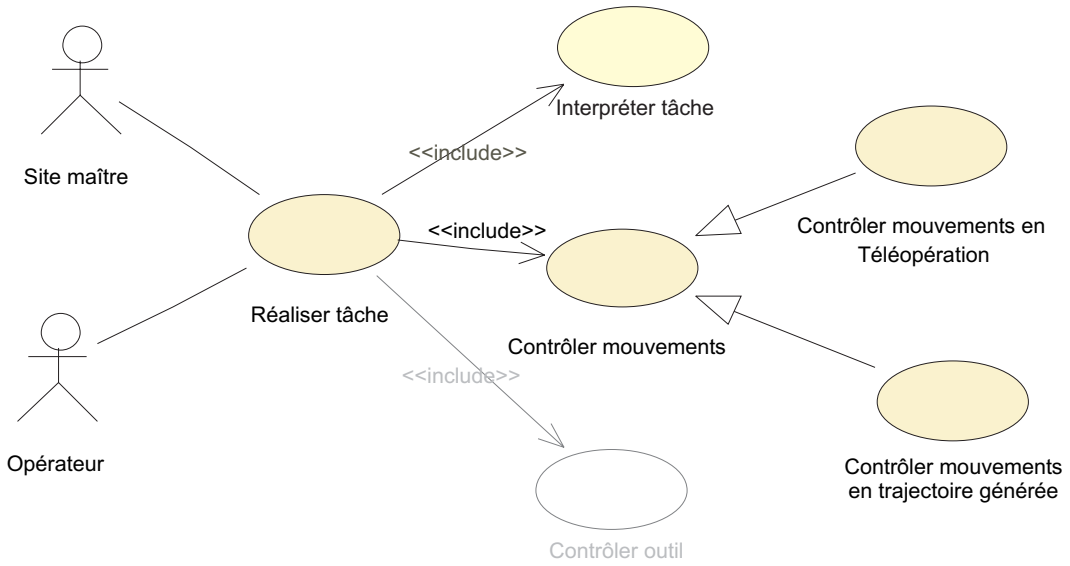

FIG. 5.12 – Cas d'utilisation en relation avec *Réaliser tâche*

À partir des fiches des cas d'utilisation *Réaliser tâche* et *Réaliser le geste médical* des diagrammes exposés précédemment, on définit une nouvelle fiche pour le cas d'utilisation *Réaliser tâche* propre au site esclave TER. Cette description textuelle est donnée ci-dessous.

Cas d'utilisation	**Réaliser tâche**
Acteurs	*Site maître*, *Robot* et *Opérateur*
Description	Les consignes de mouvements de la sonde sont envoyées par le site maître. Ces mouvements sont effectués par le *Robot* auquel est fixé la sonde. Les données de position réelle du *Robot* esclave et de force exercée sur le *Patient* sont renvoyées au site maître. Deux modes de fonctionnement sont prévus : – mode téléopération : les consignes de mouvements sont envoyées toutes les 500ms et le *Robot* suit les mouvements du *Site maître* – mode génération : le *Site maître* envoie un point de départ, un point d'arrivée, une vitesse et une accélération, les points de la trajectoire sont calculés, puis le mouvement est contrôlé.
Pré-conditions	*Patient* et *Robot* installés, contrôleur configuré pour la tâche et la connexion au site maître établie.
Post-conditions	Le *Robot* est retiré du *Patient*.
Alternative	L'*Opérateur* peut contrôler l'exécution de la tâche : – il peut suspendre la tâche et l'annuler ou la reprendre – il peut annuler la tâche.
QoS	La pression sur le ventre ne dépasse pas 3daN sur une surface de $20cm^2$. La vitesse de déplacement de la sonde est de l'ordre du centimètre par seconde. Les déplacements ne sortent pas de la zone où le gel est apposé. Les déplacements ne sortent pas d'une zone pré-définie non dangereuse.
Contraintes techniques	La localisation 3D de la sonde se fera grâce à un système Polaris (produit de Northern Digital Inc, http ://www.ndigital.com) déjà utilisé pour d'autres applications. Les actionneurs du robot sont des muscles artificiels. L'architecture du robot est *parallèle* comme présentée sur la figure 5.10.
Remarques	En raison de la structure innovante du robot, et de l'utilisation des muscles artificiels dont la commande en boucle fermée est complexe, le contrôle des mouvements doit faire l'objet d'une évaluation très poussée pour estimer la stabilité et la fluidité des mouvements.

De cette description, on peut en déduire le diagramme de séquence de la figure 5.13 représentant le scénario principal du contrôle de l'exécution de la tâche par l'*Opérateur* (ligne

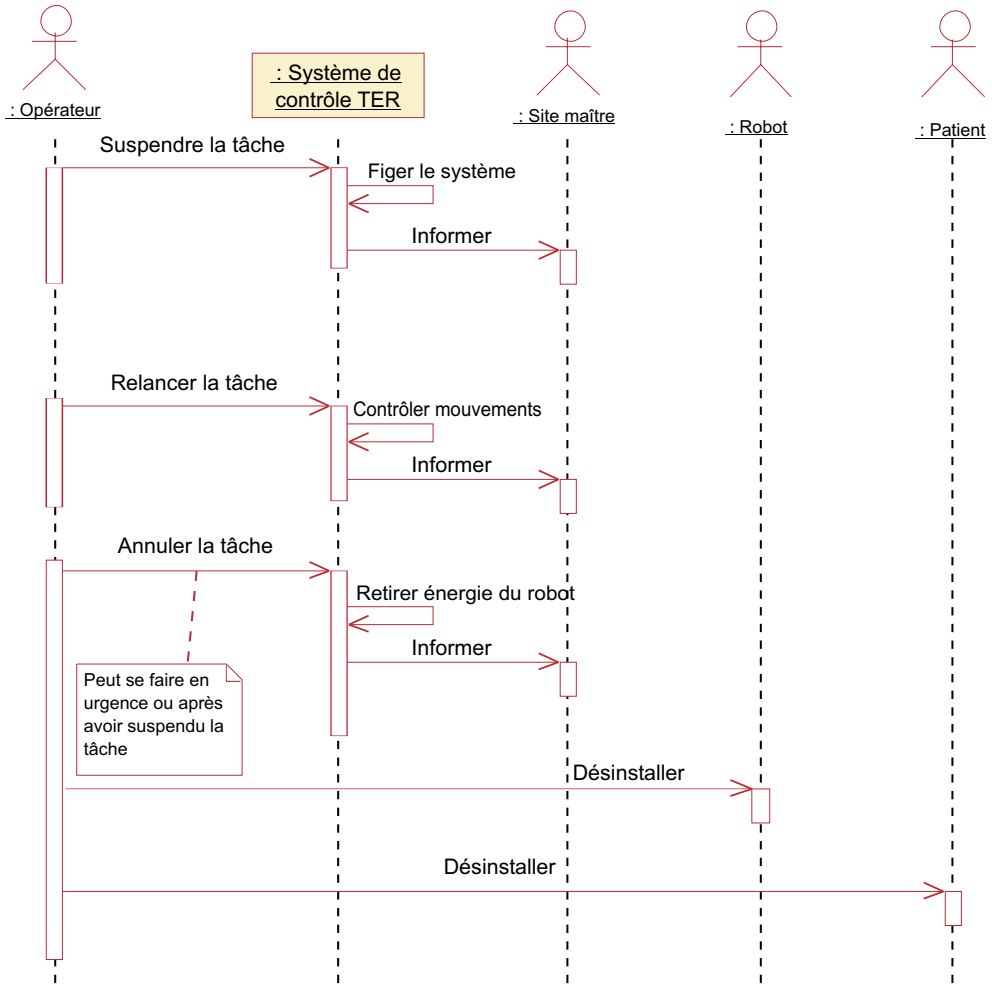

FIG. 5.13 – Diagramme de séquence du contrôle de l'exécution de la tâche par l'*Opérateur*

Alternative dans le tableau ci-dessus). Les différents messages que peut envoyer l'*Opérateur* sont indépendants, mais ils sont représentés ici sur le même diagramme pour en limiter le nombre. Il n'y a en effet aucune contrainte sur l'ordre de ces messages. Cette contrainte sera exprimée ultérieurement avec un diagramme d'états, et on représente ici tous les messages possibles que l'*Opérateur* peut émettre. Les différents messages que l'on identifie ici peuvent faire l'objet d'un cas d'utilisation séparé de *Réaliser tâche*. Tout en s'y rapportant, ces scénarios sont en effet différents du cas d'utilisation principal, ils représentent une utilisation particulière. UML permet alors d'utiliser la notion d'extension pour le diagramme des cas d'utilisation (stéréotype *«extend»*). On modélise ainsi un nouveau cas d'utilisation, *Contrôler l'exécution de la tâche* qui étend le cas d'utilisation *Réaliser tâche* (cf. figure 5.14).

Les cas d'utilisation *Contrôler mouvements*, *Contrôler mouvements en téléopération*, et *Contrôler mouvements en trajectoire générée*, ne sont pas détaillés ici sous forme de fiches, car ils sont inclus dans la description du cas d'utilisation *Réaliser tâche*. Le cas d'utilisation *Interpréter tâche* ne fait pas non plus l'objet d'une description textuelle car dans le cas d'un système TER, il se limite à l'interprétation d'un message émis par le site maître.

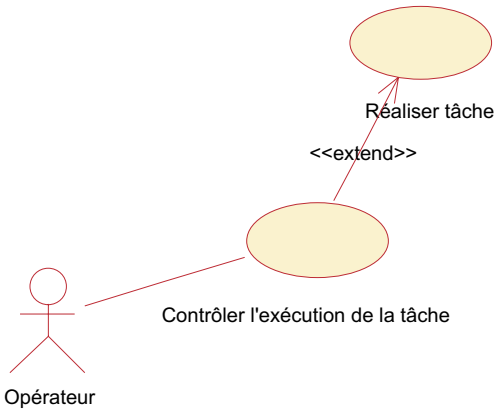

FIG. 5.14 – Cas d'utilisation *Contrôler l'exécution de la tâche*

5.3.4.2 Cas d'utilisation *Gestion patient*

Ce cas d'utilisation est proche de celui présenté lors de la modélisation métier. L'acteur *Opérateur* hérite donc d'une partie des tâches du *Spécialiste*, ou plus précisément d'un acteur *Assistant* défini dans le diagramme de classes de la figure 5.7.

Cas d'utilisation	**Gestion patient**
Acteurs	*Opérateur* et *Patient*
Description	L'*Opérateur* installe et désinstalle le *Patient*, le surveille et répond à ses questions. Il recouvre la surface qui sera en contact avec la sonde avec du gel permettant une meilleure conduction des ultrasons et facilitant les déplacements. Le *Patient* répond aux questions de l'*Opérateur* ou émet des requêtes.
Alternative	Lors de l'exécution de la tâche, l'*Opérateur* peut suspendre la tâche (cas d'utilisation *Contrôler l'exécution de la tâche*), et remettre du gel sur certaines parties de la surface.
QoS	Il s'agit de mettre en confiance le *Patient* et l'installer de façon à limiter le stress. La dose de gel doit être suffisante et présente partout où la sonde se déplacera.

5.3.4.3 Cas d'utilisation *Configuration du robot*

La *Configuration du robot* peut souvent faire l'objet de deux cas d'utilisation séparés : la programmation de la tâche et la configuration du contrôleur exposés par Carroll (1999). Dans le cas du robot TER, la structure particulière du robot esclave, et surtout le fait qu'il faille installer d'abord le patient, puis le robot, implique une imbrication de ces deux cas d'utilisation. Nous avons donc choisi de représenter ces exigences en un seul cas d'utilisation que l'on nomme donc *Configuration du robot*. L'ensemble des actions que l'*Opérateur* aura à effectuer sont décrites par le diagramme de séquence de la figure 5.15. Au sein de ce diagramme on

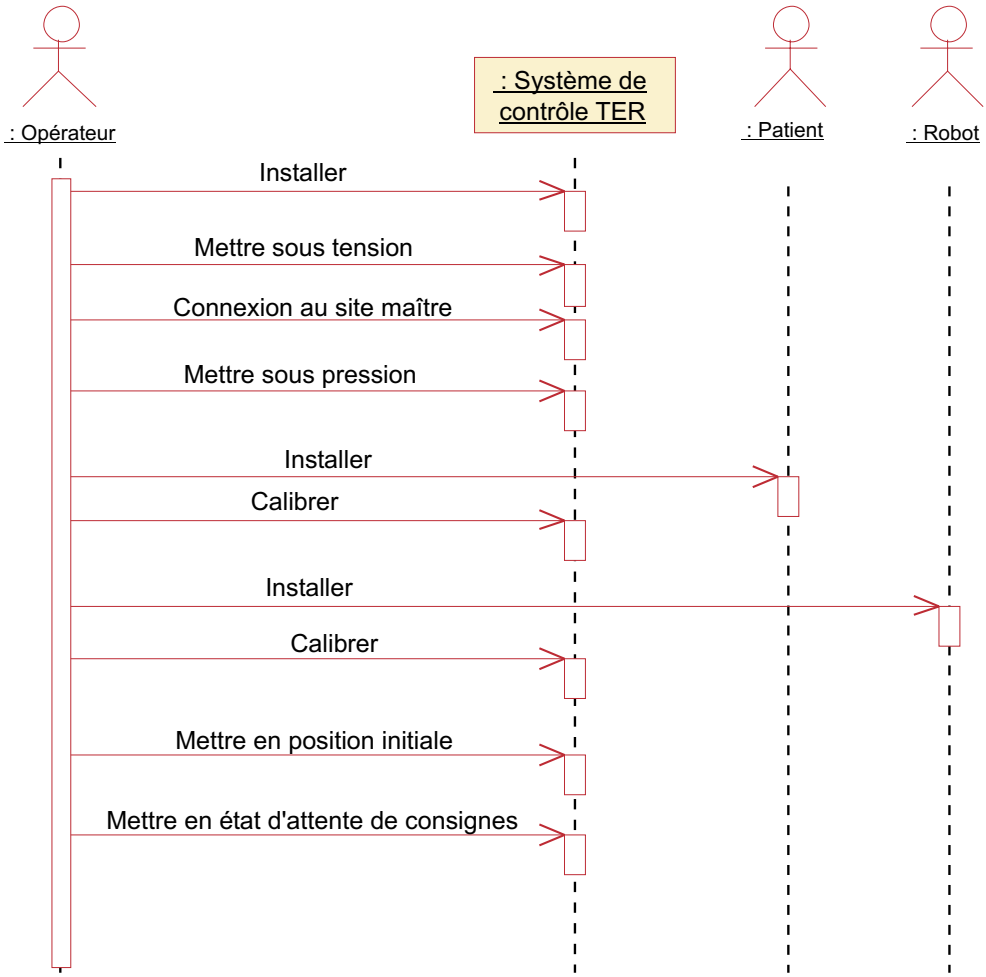

FIG. 5.15 – Diagramme de séquence de la configuration du robot

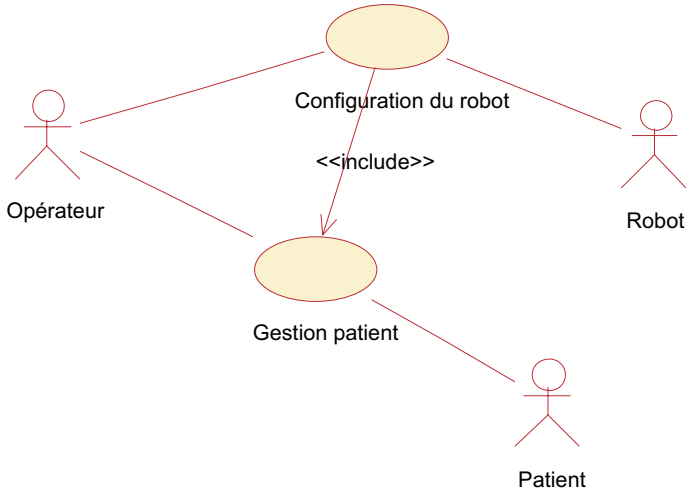

FIG. 5.16 – Diagramme des cas d'utilisation illustrant la relation *«include»*

peut trouver un message nommé *Installer* qui s'adresse au *Patient*. Or ce message déclenche un scénario (l'installation du *Patient*, la pose du gel, etc.) qui appartient au cas d'utilisation *Gestion patient*. Ceci permet de noter une relation de type *«include»* entre les cas d'utilisation *Configuration du robot* et *Gestion patient* (cf. figure 5.16). De la même manière que pour les autres cas d'utilisation, il est possible de décrire les cas d'utilisation de manière textuelle comme présenté dans le tableau ci-dessous.

Cas d'utilisation	**Configuration du robot**
Acteurs	*Opérateur* et *Robot*
Description	L'*Opérateur* installe et désinstalle le *Robot*. Il entre les données liées au *Patient* et au *Robot* pour réaliser la tâche. Il assure la maintenance du *Robot*.
Alternative	Si l'*Opérateur* décèle un problème lors de l'installation, il peut résoudre le problème et poursuivre la configuration, ou arrêter l'installation.
QoS	Les opérations doivent être suffisamment rapides pour ne pas faire attendre le *Patient* dans une position inconfortable.
Remarques	Ces étapes sont particulièrement critiques pour la sécurité du *Patient*

5.3.5 Diagramme global des cas d'utilisation, structure dynamique et statique du *Système de contrôle TER*

Le diagramme global des cas d'utilisation est donné figure 5.17. Il reprend chaque relation identifiée précédemment. Il est alors possible de définir un diagramme d'états partiel présenté figure 5.18, représentant les états du *Système de contrôle TER*. Les changements d'états, principalement induits par les messages envoyés par l'*Opérateur* et le *Site maître* sont retranscrits

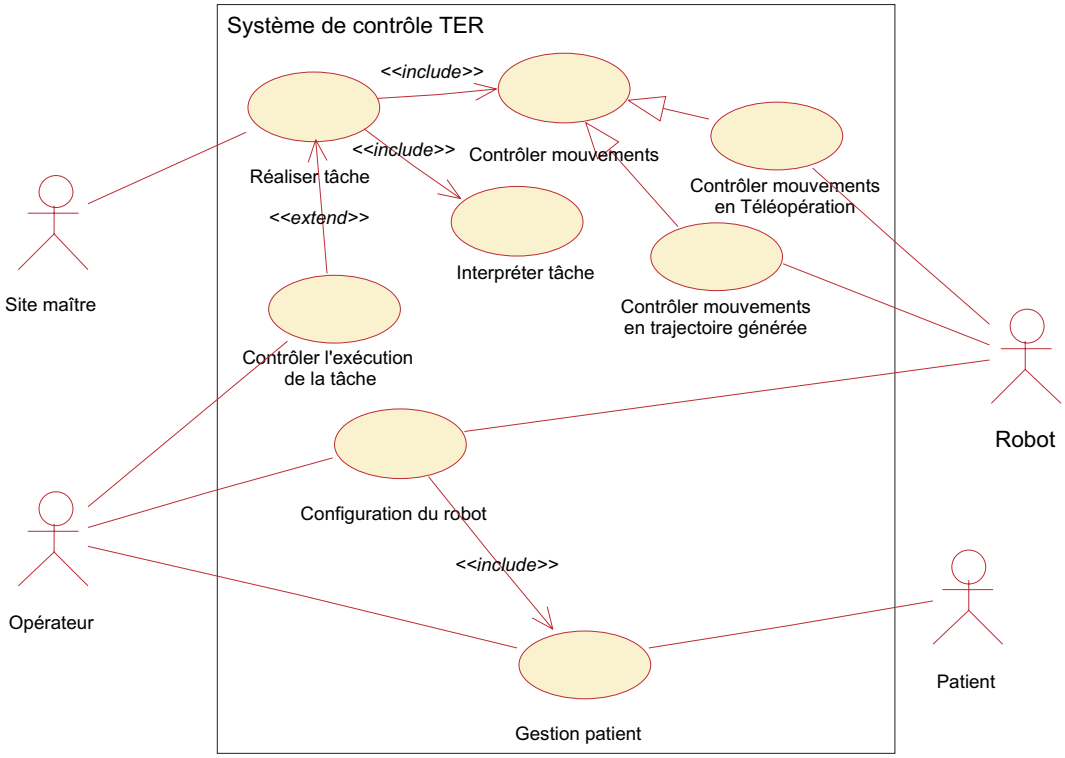

FIG. 5.17 – Diagramme général des cas d'utilisation

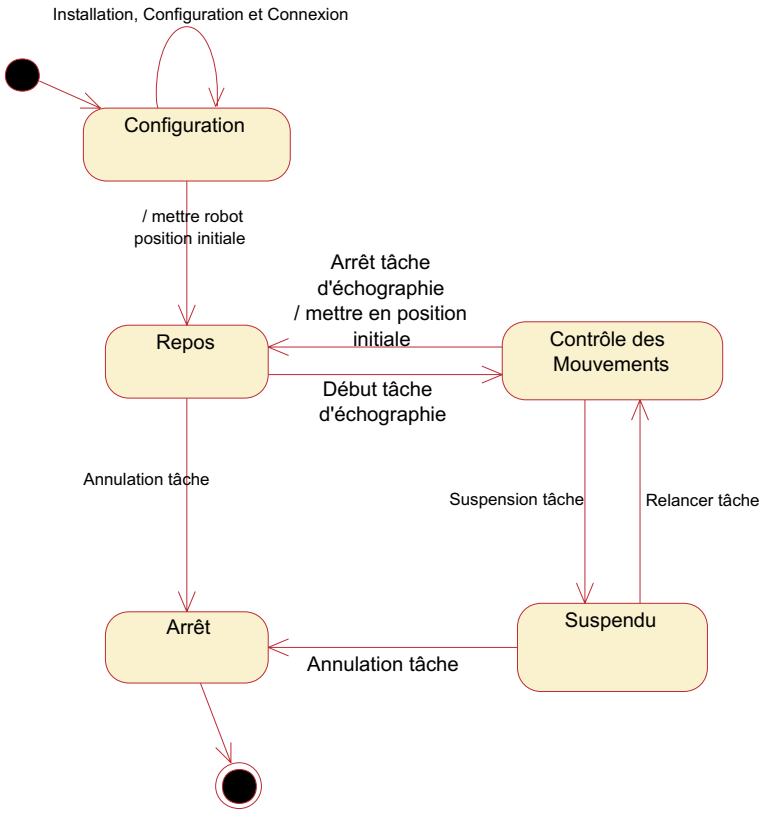

FIG. 5.18 – Diagramme d'états du *Système de contrôle TER*

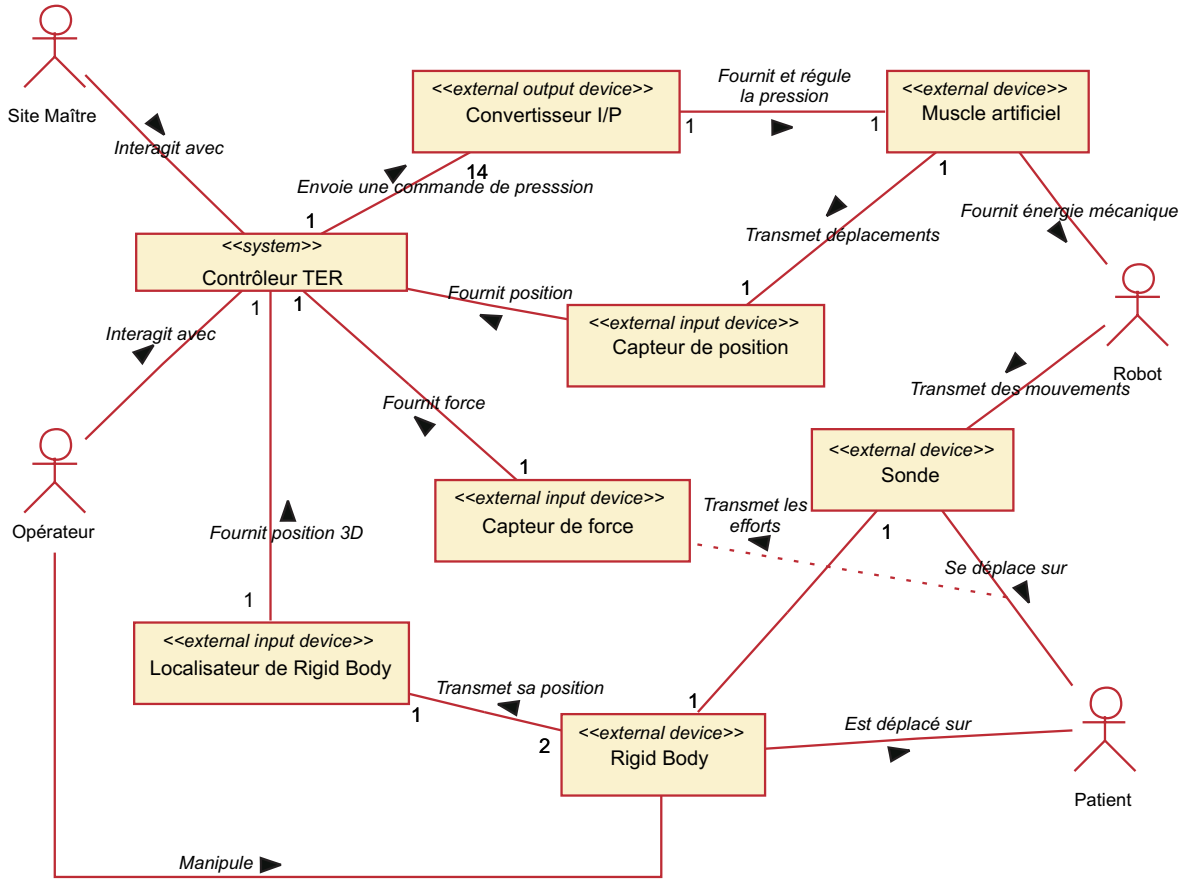

FIG. 5.19 – Diagramme de classes du *Système de contrôle TER*

ici en tant qu'événements. Ainsi, *Arrêt tâche d'échographie* et *Début tâche d'échographie* correspondent aux messages du *Site maître* présentés sur le diagramme de séquence de la figure 5.11. *Suspension*, *Relancer* et *Annulation tâche* correspondent aux messages émis par l'*Opérateur*, présentés sur le diagramme de séquence de la figure 5.13, et correspondant au contrôle de l'exécution de la tâche.

Sur la base des exigences fonctionnelles et des contraintes techniques (dont l'utilisation des muscles artificiels et d'un localisateur 3D Polaris), une structure électronique, décrite par le diagramme de classes de la figure 5.19, a été conçue.

5.4 Identification des dangers et estimation du risque

Bien que cette section constitue en général le cœur de l'analyse du risque, l'étape précédente de définition du système est fondamentale pour notre démarche. Tout d'abord elle a permis de définir des tâches cohérentes pour les différents acteurs, ce qui est fondamental pour la sécurité, puis a fourni une base pour la suite de l'analyse du risque. Comme cela est présenté dans le chapitre 4, les modèles UML sont injectés dans les différentes analyses, puis

Fréquence d'événement	Gravité du dommage				
	1 Catastrophique	2 Majeure	3 Mineure	4 Minime	5 Négligeable
Fréquente	H	H	H	H	I
Probable	H	H	H	I	I
Occasionnelle	H	H	I	I	L
Rare	H	I	I	L	T
Improbable	I	I	L	T	T
Invraisemblable	I	L	T	T	T

FIG. 5.20 – Table d'estimation du risque

les modifications concernant l'utilisation, les exigences et la conception sont intégrées aux modèles.

Pour chaque étape de l'analyse présentée ici, il convient d'estimer, lorsque cela est possible, le risque induit par un phénomène dangereux, une situation dangereuse ou un événement dommageable. Pour cela, il convient de choisir des niveaux de gravité et de probabilité d'occurrence, selon une table de risque comme cela est présenté au chapitre 1. Dans notre analyse nous utiliserons cette table d'estimation du risque qui est reproduite figure 5.20. Les couples (fréquence, gravité) résultant en un risque élevé (H), sont mis en valeur car ils représentent les risques les moins tolérables. Nous n'effectuerons pas d'évaluation du risque, qui consiste à choisir le niveau de risque acceptable, mais afin d'illustrer la démarche, nous nous appuierons tout de même sur le fait qu'un risque de niveau H, fera sans aucun doute l'objet d'une réduction. Cela permet alors de proposer des solutions cohérentes et d'illustrer comment les moyens de maîtrise du risque sont introduits dans notre démarche.

Une première section présente l'analyse préliminaire des risques induits par le système robotique esclave TER. Les sections suivantes sont dédiées à l'analyse des modes de défaillance des messages échangés entre tous les éléments du système (acteurs et éléments électroniques, mécaniques ou informatiques). Une dernière section présente une utilisation des arbres de fautes. Par souci de présentation, les diagrammes UML et les tableaux de l'AMDEC développés sur papier lors de cette analyse, ne seront pas tous présentés. Seuls les plus importants, et ceux qui illustrent au mieux notre démarche seront inclus dans ce chapitre. Des diagrammes et tableaux d'analyse supplémentaires sont données en annexe B.

5.4.1 Analyse préliminaire des dangers

On effectue sur la base de la description du système, présentée précédemment, une analyse préliminaire des dangers. On choisit de classer ces phénomènes dangereux en plusieurs catégories :

- physique : le phénomène dangereux est d'origine physique, il résulte en général en un dommage physique ;
- électrique (électrocution, décharge électrostatique, etc.) ;
- psychologique : le phénomène dangereux peut conduire à des dommages d'ordre psychologique ;
- environnement : le phénomène dangereux concerne l'environnement d'un acteur (auditif, visuel, température, etc.). Il peut résulter en des dommages psychologiques (stress) ou même physiques (perte de l'audition).

Une analyse préliminaire peut devenir complexe si l'on souhaite trop détailler chaque élément. Il est important de bien faire la différence entre les phénomènes dangereux et les dommages. La figure 5.21 représente l'analyse préliminaire des phénomènes dangereux que l'on peut identifier dès le début du projet. Les colonnes *Solutions possibles* et *Remarques* fournissent des modifications que l'on peut intégrer à différents niveaux du développement :

- modification de l'utilisation (comme par exemple la spécification d'une zone travail avant la réalisation de la tâche) que l'on introduit dans les diagrammes de séquence ;
- modification des exigences et plus particulièrement des contraintes de type QoS (réduire le temps d'attente du *Patient* avec le *Robot* installé, contraindre la vitesse de déplacement, etc.) qui sont ajoutées aux fiches des cas d'utilisation correspondantes ;
- modification de la conception des dispositifs matériels (implantation d'un arrêt d'urgence, isolation des composants relatifs à la pression, implanter un visuel de la force exercée sur le *Patient*) et du logiciel (contraindre la vitesse de déplacement) que l'on répercute sur les modèles.

La modification des modèles concerne tous les diagrammes. L'exemple classique est ici l'implantation d'un arrêt d'urgence qui provoque la sortie de l'état *En fonctionnement* regroupant tous les états du diagramme de la figure 5.18. Ce changement d'état implique alors la mise à zéro de la pression dans les muscles de manière matérielle, puis les actions correspondant à la désinstallation du *Robot* et du *Patient* (cf. figure 5.22). Il convient alors de mettre à jour le diagramme de séquence du contrôle de l'exécution de la tâche par l'*Opérateur* de la figure 5.13, en ajoutant le message *Arrêter d'urgence*, envoyé par l'*Opérateur* vers le *Système de contrôle TER*. On assigne ainsi à l'*Opérateur* une nouvelle responsabilité qui est la surveillance, et l'on introduit un arrêt d'urgence résultant en la mise à zéro de la pression des muscles. L'arrêt d'urgence, qui est implanté lors de l'activité de maîtrise du risque, doit donc consister en un sous-système coupant directement l'alimentation en pression des muscles. Cette exigence est fondamentale, car les systèmes d'arrêt d'urgence sont en général des coupe-circuits électroniques, ce qui ne convient pas ici.

5.4.2 Analyse des modes de défaillance des acteurs

Les acteurs principaux du site esclave TER sont le *Site maître*, l'*Opérateur* et le *Robot*. Les modes de défaillance du *Robot* seront abordés dans la section concernant les composants

	Danger	Acteurs			Cas d'utilisation			Gravité	Solutions possibles	Remarques
		Opérateur	Patient	Robot	Gestion du patient	Réaliser tâche	Configurer tâche			
Physique	Déplacement sur une zone sensible ou dangereuse du patient		X		X	X		1	- Spécification d'une zone de travail avant la tâche. - Contraindre les mouvements dans cette zone. - Prévoir un arrêt d'urgence.	Envisager des solutions logicielles mais aussi matérielles
	Pression trop importante sur le patient		X		X	X		1	- Contraindre les efforts des mouvement grâce à un capteur de force. - Visuel de la force exercée sur le patient. - Prévoir un arrêt d'urgence coupant la pression dans les muscles	Estimer la force maximale de l'architecture envisagée (comprenant les muscles artificiels)
	Déplacement brusque sur le patient		X			X		2	Contraindre la vitesse de déplacement de manière logicielle.	
	Projections d'éléments (fouet des câbles, explosion des muscles, etc.)	X	X	X		X		1	- Isoler au maximum les composants matériels transmettant les forces mouvement du patient et de l'opérateur. - Prévoir un programme de maintenance.	
	Chute du robot		X	X		X		1	Système maintenant le robot en cas de défaillance	
	Frottement des câbles sur le patient		X			X		2	Créer un système intermédiaire pour isoler le patient des câbles	
Electrique	Electrocution	X	X	X	X		X	1	Aucune partie métallique ne doit être en contact avec le patient	Répondre aux normes en vigueur pour le système électrique (fusible, alimentation aux normes, etc.)
	Décharge électrostatique	X	X	X	X		X	2	Les parties métalliques du robot doivent être reliées à la masse	
Psychologique	Attente	X			X	X	X	4	Si le système est bloqué, l'opérateur doit pouvoir annuler la tâche en cours et dégager le patient	Il convient de réduire au maximum le temps d'attente du patient lors de l'installation du patient
	Stress	X			X	X	X	4	Le patient doit pouvoir accéder au système de visioconférence pour dialoguer avec le spécialiste	
Environnement	Bruit dû au flot d'air dans les muscles	X	X			X		4	Isoler les composants transmettant la pression (compresseur, convertisseur I/P, muscle artificiels)	
	Flot d'air sur le patient							3	Isoler les composants transmettant la pression (compresseur, convertisseur I/P, muscle artificiels)	
	Choc sonore (fuite d'air)							3	Isoler les composants transmettant la pression (compresseur, convertisseur I/P, muscle artificiels)	

FIG. 5.21 – Table d'analyse préliminaire des dangers

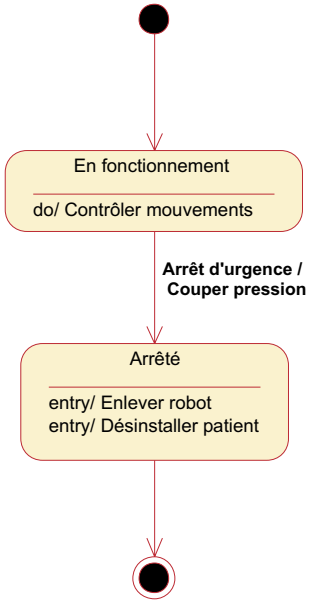

FIG. 5.22 – Diagramme d'états illustrant l'implantation d'un arrêt d'urgence

mécaniques. Nous définissons en effet le *Robot* par la structure mécanique articulée composée essentiellement de composants mécaniques. Tous les autres composants comme les capteurs et les actionneurs sont intégrés dans le sous-système que l'on nomme *Système de contrôle TER* (cf. figure 5.17).

5.4.2.1 Modes de défaillance du *Site maître*

On analyse les messages provenant du *Site maître*, en se basant sur le diagramme de séquence présenté précédemment sur la figure 5.11. Le scénario principal du cas d'utilisation *Contrôler mouvements en téléopération* est présenté figure 5.23. Ce diagramme de séquence fait apparaître une contrainte de temps exprimée précédemment en utilisant le formalisme de la version 1.4 d'UML (OMG, 2001), qui consiste à étiqueter les messages (ici *a* et *b*), puis à exprimer une contrainte en utilisant les mots clés *SendTime*, *ReceiveTime*, etc. En plus de cette contrainte, il est possible de rajouter sur ces diagrammes des notes sur le côté pour exprimer de façon textuelle certaines informations (la limite étant fixée par la lisibilité du diagramme).

À partir de ce diagramme, on utilise un tableau de l'AMDEC pour analyser les messages, en utilisant les modèles d'erreur fournis au chapitre 4. Il est important de noter que la partie communication avec le *Site maître* a été gérée par France Telecom R&D, qui a établi un système de communication basé sur un protocole et des algorithmes de détection des fautes. La modélisation de ce partage de travail est représentée sur la figure 5.24.

Le tableau d'analyse des modes de défaillance du message *Contrôler_robot* est présenté figure 5.25. Afin de faire le lien avec les moyens de maîtrise du risque, on estime le risque à la plus forte valeur (H) et l'on propose des moyens pour le réduire. L'omission ou le retard du message provoque l'établissement d'un signal, *Perte connexion*, que nous introduisons dans

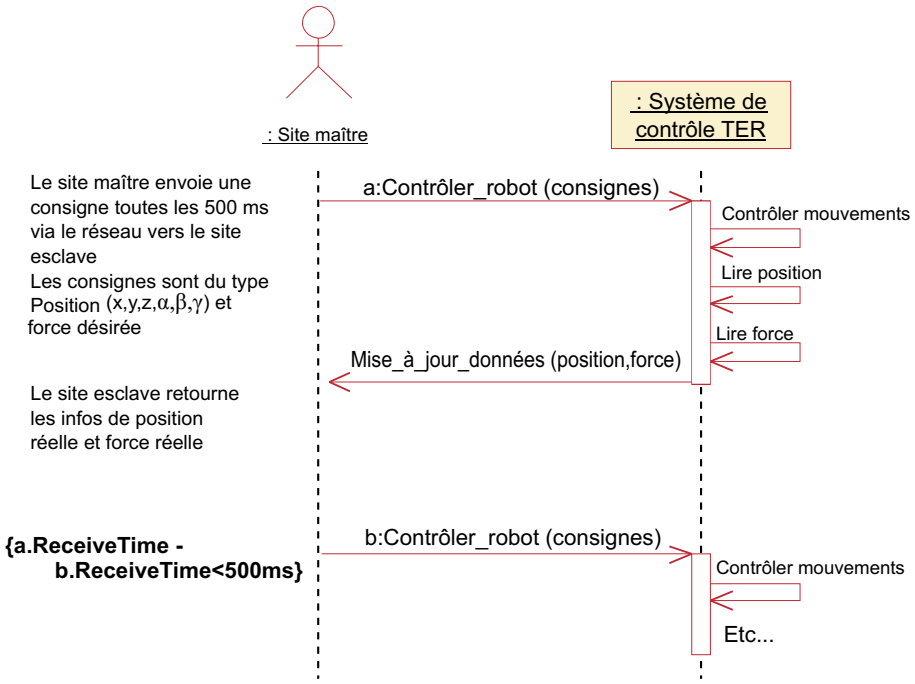

FIG. 5.23 – Diagramme de séquence principal du cas d'utilisation *Réaliser tâche*

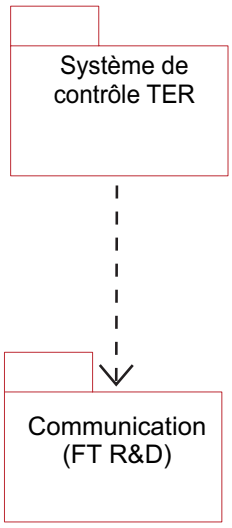

FIG. 5.24 – Utilisation du *package Communication* par le *Système de contrôle TER*

Composant / Fonction/ Message	**Mode de défaillance** (erreur)	**Cause de la défaillance**	**Effets** : a. Niveau local b. Niveau supérieur c. Niveau système	Risque			**Moyens de détection possibles (en ligne)** a. Mode défaillance b. Effets	**Solutions possibles** a. Prévention b. Protection c. Autres actions d. Remarques
				Gravité	**Occurrence**	**Risque**		
Contrôler_robot (consignes)	Trop tard ou omission	Faute `Site Maître` ou faute lien	a. Les consignes ne sont pas mises à jour b. Contrôle des mouvements sur anciennes valeurs c. Déplacements par saccades	4	F	H	a. Utilisation d'une tempo, établir signal Perte connexion.	a. Utilisation protocole (fournit par France Telecom R&D) b. Figer le robot , si le délai expire mettre en position initiale c. Informer Opérateur
	Trop tôt	Faute `Site Maître`	a. Le contrôle des mouvements n'est pas terminé b. Commande interrompue c. Déplacement brusque	3	F	H	a. Le contrôleur n'a pas terminé le contrôle de position	a. Utiliser des tâches indépendantes au sein du contrôleur b. Le contrôleur valide la fin du contrôle d'un mouvement
	Ordre incorrect du message dans la séquence	Faute `Site Maître`	a. Comportement inconnu	1	P	H	a. Le contrôleur n'est pas en état d'attente de ce message	b. Ignorer le message c. Informer Opérateur
	Arguments erronés (nombre, type ou valeur)	Faute `Site Maître` ou faute lien	c. Mouvement non désiré	1	F	H	a. Filtrage numérique	a. Protocole de communication (cheksum, etc.) b. Ignorer les valeurs, si trop de valeurs fausses à la suite mettre au repos c. Informer Opérateur

FIG. 5.25 – AMDEC du message *Contrôler_robot* émit par le *Site maître*

le diagramme d'états de la figure 5.26. Par rapport au diagramme d'états présenté précédemment (cf. figure 5.18), on introduit un nouvel état temporaire, *Pause*, qui permet de mettre le système en attente si le message subit un simple retard, ou s'il existe une coupure momentanée. Si un délai (à fixer par la suite) est expiré alors le système met le robot dans la position initiale, et attend soit une reconnection, soit une annulation de la tâche par l'*Opérateur*. Les diagrammes d'états impliquent que les changements d'état sont effectués uniquement lorsque les événements spécifiés arrivent. Cela signifie que si l'on se trouve dans l'état *Mise en fonctionnement*, et que l'événement *Lancer tâche d'échographie* arrive, alors il n'y aura pas de changement d'état. Ainsi, le modèle ne permettant pas de changement d'états non maîtrisé, il convient ensuite de concevoir notamment au moyen de *patterns* d'implantation des machines à états respectant cette contrainte (Douglass, 1999).

Les solutions proposées dans l'AMDEC concernant les tâches et leur synchronisation, impliquent une connaissance de la conception du logiciel, alors que nous nous situons encore dans la spécification. Au niveau actuel, ceci peut être vu comme une exigence pour le logiciel, qui est la nécessité d'utiliser un noyau temps réel pour la gestion des tâches. On ajoute alors cette information au cas d'utilisation *Réaliser tâche*. Ceci illustre bien le fait qu'une analyse du risque qui s'effectue en parallèle du processus de développement, permet de proposer des solutions, et donc de guider certains choix technologiques.

Les points abordés ci-dessus sont très courants dans de tels systèmes. Les temporisations dans la perte de connexion, et l'exigence d'un noyau temps réel pour le contrôleur, sont des

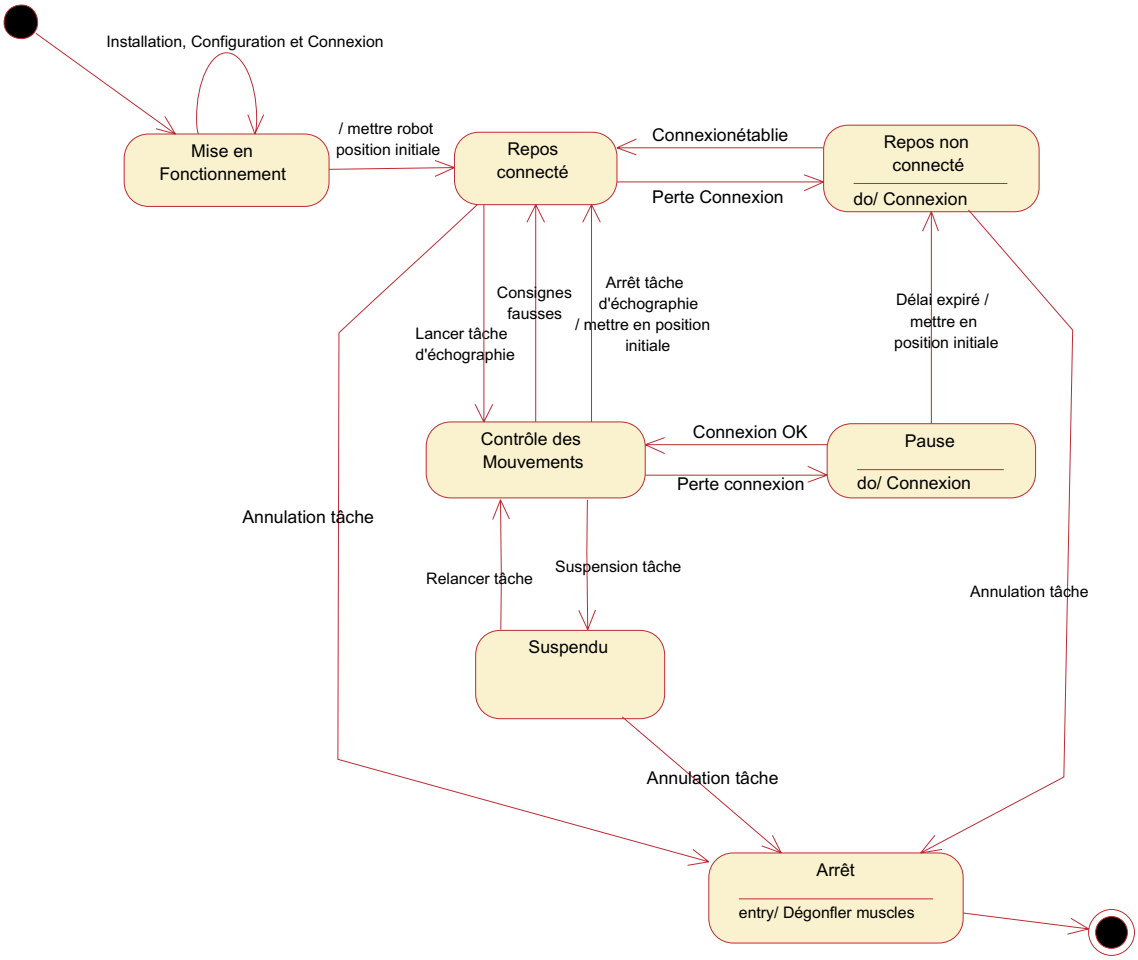

FIG. 5.26 – Diagramme d'états du Système de contrôle TER prenant en compte les modes de défaillance du message *Contrôler_robot*

solutions très classiques. Cependant, un processus de développement cohérent doit pouvoir justifier l'utilisation de telles techniques comme cela est présenté dans cette démarche.

5.4.2.2 Modes de défaillance de l'*Opérateur*

Lorsqu'un acteur est un humain, l'analyse des modes de défaillance de cet acteur correspond à l'analyse des erreurs humaines. Comme présenté dans le chapitre 4, nous basons cette démarche sur les descriptions des scénarios (pour chaque cas d'utilisation), les descriptions des interfaces lorsque cela est possible, et des connaissances sur le comportement et les performances des acteurs considérés. L'analyse de l'AMDEC se base sur le diagramme de séquence de la figure 5.15, qui correspond à la configuration du robot, et celui de la figure 5.13 qui décrit le contrôle de l'exécution de la tâche. Cependant, ces diagrammes ne sont pas suffisamment détaillés pour effectuer une analyse. En effet, il convient de décrire précisément chacune des tâches exprimées par des messages. Ainsi, le message *Installer* de la figure 5.15, entre l'*Opérateur* et le *Système de contrôle TER*, exprime différentes actions comme les branchements des tuyaux d'alimentation en air et des câbles d'alimentation électrique. Toutes ces actions peuvent être décrites par des diagrammes de séquence, détaillés par des documents textuels.

Les diagrammes UML permettent alors de guider les différentes analyses des erreurs humaines. Dans les diagrammes de séquence présentés précédemment, les interfaces ne sont pas précisées et elles peuvent être matérielles ou logicielles. C'est la nature même du message qui guide l'analyse, et ce sont les spécialistes de chaque domaine qui peuvent proposer des solutions. Ainsi, pour l'action *Mettre la pression*, il est possible d'effectuer l'analyse présentée sur la figure 5.27, où les erreurs principales sont présentées à partir du modèle d'erreur du chapitre 3. Dans ce tableau, nous ne faisons pas apparaître les causes des erreurs humaines car elles sont multiples, et complexes. Le but n'est pas ici d'effectuer une analyse sur la production de ces erreurs qui concerne les sciences cognitives, mais d'intégrer dans le système les moyens pour maîtriser ces erreurs.

À titre d'exemple, une solution possible est alors d'utiliser le système présenté à la figure 5.28. Ainsi, l'*Opérateur* établit la pression progressivement, et consulte un manomètre pour surveiller visuellement la pression. Il est possible d'installer un sous-système logiciel de surveillance avec un capteur de pression, et de créer au niveau du logiciel des interblocages pour que l'ordre des messages soit respecté.

Les autres messages, *Installer*, *Calibrer*, et *Mettre en position initiale*, après avoir été détaillés ont fait l'objet d'une même analyse. Puis, les cas d'utilisation *Contrôler l'exécution de la tâche* et *Gestion patient* en relation avec l'*Opérateur* ont été analysés de cette manière. Afin de nous limiter dans ce mémoire, l'ensemble des modifications des exigences, de l'utilisation et de la conception, n'est pas présenté ici.

Composant / Fonction/ Message	**Mode de défaillance** (erreur)	**Effets** : a. Niveau local b. Niveau supérieur c. Niveau système	Risque			**Moyens de détection possibles (en ligne)** a. Mode défaillance b. Effets	**Solutions possibles** : a. Prévention b. Protection c. Autres actions d. Remarques
			Gravité	Occurrence	Risque		
Mettre la pression	Omission	a. Pas d'énergie dans les muscles artificiels b. Nécessité de recommencer initialisation c. Attente du patient (stress)	4	P	I	a. Opérateur lit le manomètre	a. Guide d'utilisation, formation, étapes détaillées à l'écran b. Faire un test de gonflage des muscles lors de l'initialisation c.
	Ordre incorrect : avant mise sous tension	Avant la fin de l'initialisation électronique les sorties ne sont pas à zéro c. Comportement inconnu (mouvements brusques des muscles)	5	P	I		a. Guide d'utilisation, formation.
	Pression réglée trop forte	a. Destruction convertisseurs I/P	2	O	H	a. Opérateur lit le manomètre	d. La limite maximum étant fixée par la pression d'alimentation, il convient de déterminer la pression max d'entrée des convertisseurs (voir fabricant)
	Pression réglée trop faible (inférieure à alimentation des convertisseursI/P = 6bar)	a. Pression insuffisante dans les muscles b. Pas assez de pression sur le patient c. Fonctionnement non dangereux mais chaotique (stress patient)	4	F	H	a. Opérateur lit le manomètre b. Capteur de pression	a. Indication sur le manomètre b. Faire un test de gonflage des muscles à l'initialisation et vérifier la pression par le logiciel.

FIG. 5.27 – AMDEC du message *Mettre la pression*

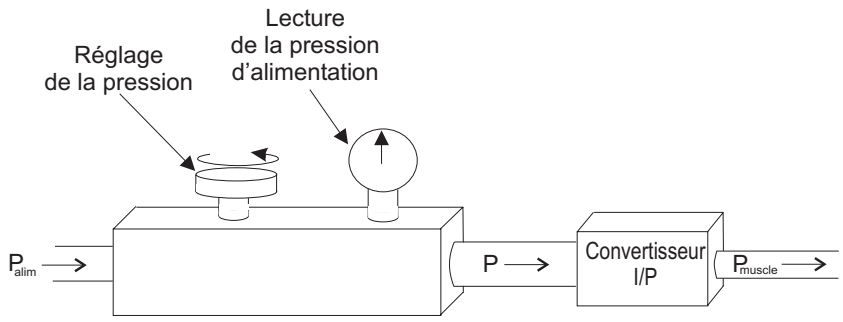

FIG. 5.28 – Système d'alimentation en air

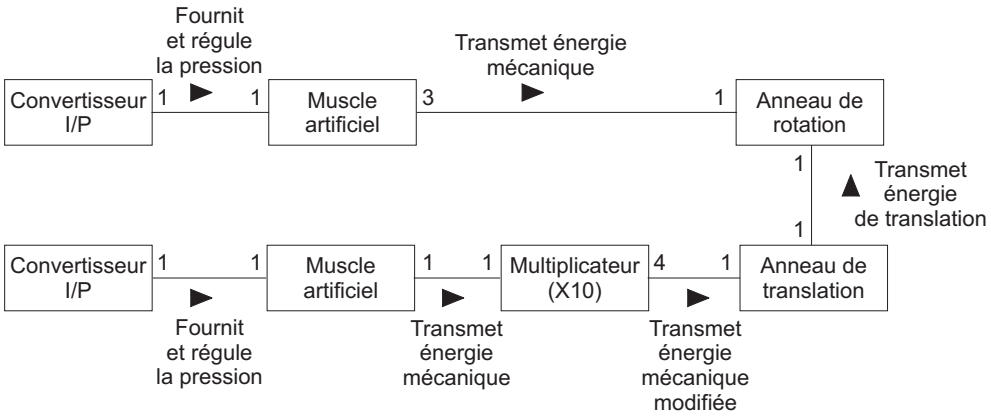

FIG. 5.29 – Principaux composants mécaniques pour l'actionnement des anneaux de déplacement

5.4.3 Analyse des modes de défaillance des composants matériels

Dans le cadre de la robotique, l'analyse des modes de défaillance des composants matériels correspond à l'étude mécanique de l'acteur *Robot*. Le spécialiste du domaine, choisit les technologies et les constructions en fonction des exigences exprimées par les cas d'utilisation. Pour une analyse du risque système, il est évident que la démarche n'est pas de remettre en cause les choix fondamentaux de la mécanique, mais de redéfinir certaines exigences, et d'identifier les effets des défaillances sur le système. Ces défaillances sont spécifiées par le spécialiste du domaine mécanique mais les effets sont calculés grâce à une connaissance du système global. Ainsi, connaître exactement la composition de la mécanique ne permettrait pas une analyse synthétique. La modélisation, que l'on peut effectuer en diagramme de classes, est présentée sur la figure 5.29. Une description plus détaillée de la mécanique est donnée par Vilchis et al. (2001a), et une photo de la conception a été donnée en figure 5.10. Sur la base d'un tel diagramme il est possible d'effectuer une AMDEC classique comme celle de la figure 5.30, et d'en déduire des moyens possibles de réduction du risque. On peut noter que si certains éléments ne sont pas suffisamment définis, il sont mis en évidence au sein de l'AMDEC. Ainsi, la question de la stabilité du robot en cas de crevaison d'un muscle est une question importante pour la sécurité qu'il convient d'étudier. C'est une situation dangereuse qui peut apparaître dans certaines configurations du robot.

5.4.4 Analyse des modes de défaillance des composants électroniques

L'analyse des modes de défaillance des messages émis par les composants électroniques peut se faire comme pour les autres domaines. Cependant, par simplification, et du fait de la teneur de ces messages qui ne consistent qu'en un échange de données, nous considérerons chaque composant et ses modes de défaillance. Ainsi, l'unique fonction d'un capteur de position étant de fournir la position lue, nous analyserons les modes défaillance de la fonction de lecture de position, et plus particulièrement les erreurs de la valeur retournée.

Composant / Fonction/ Message	**Mode de défaillance** (erreur)	**Cause de la défaillance**	**Effets** : a. Niveau local b. Niveau supérieur c. Niveau système	Risque			**Moyens de détection possibles (en ligne)** a. Mode défaillance b. Effet c. Cause	**Solutions possibles** : a. Prévention b. Protection c. Autres actions d. Remarques
				Gravité	Occurrence	Risque		
Muscle artificiel	Énergie transmise trop faible (crevaison)	Crevaison	a. Longueur augmente b. Effort sur patient diminue c. Bruit, souffle d'air, désagréments	5	F	I	a. Capteur de pression	a. Maintenance préventive b. Système de maintient du robot, puis suspendre et annuler la tâche
			c. Déterminer si le robot peut basculer	1	?	?		c. Déterminer s'il existe des positions pour lesquelles le robot peut basculer en cas de crevaison d'un muscle
	Energie transmise non prévue (explosion)	Rupture de la tresse ou du sertissage	a. Projections b. Blessures physiques	2	R	I		a. Éloigner au maximum les muscles du patient b. Protection autour des muscles.

FIG. 5.30 – AMDEC d'un muscle artificiel

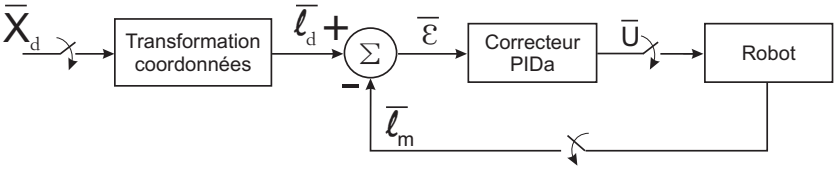

FIG. 5.31 – Commande en boucle fermée avec un régulateur de type PIDa

Une vue d'ensemble des composants a été présentée sur le diagramme de classes de la figure 5.19. Sur cette modélisation, les flots de données sont représentés mais la corrélation entre ces données n'est pas modélisée. Plutôt que de détailler le logiciel du contrôleur, on peut utiliser ici des modèles automatiques pour exprimer l'utilisation des valeurs échangées. Ainsi, on peut modéliser la boucle de rétroaction avec le diagramme de bloc classique de la figure 5.31. Sur ce diagramme sont notés les différents paramètres de la boucle :

- X_d est la position 3D du robot désirée par le Site maître,
- l_d est la longueur des câbles, calculée pour obtenir la position X_d du robot sur le patient,
- l_m est la longueur réelle des câbles,
- ε est l'erreur entre l_d et l_m,
- et U est la commande envoyée aux actionneurs du robot.

Cette commande est un régulateur en position de type PID, avec un terme d'anticipation (Carroll, 1999). La force n'est pas prise en compte dans cette version du contrôleur, elle est transmise à titre d'information au *Site maître*. Cependant, la mesure de force peut être utilisée pour des mécanismes de protection, mais nous n'abordons pas cet aspect ici. Nous considérons le système dans sa version première et purement fonctionnelle (sans aucun mécanisme de protection).

Dans cette démarche, les tableaux de l'AMDEC des composants *Capteur de position*, *Convertisseur Intensité/Pression* et *Contrôleur TER* sont présentés sur les figures 5.32 5.33 et 5.34. Les analyses des composants *Capteur de position* et *Convertisseur Intensité/Pression*, conduisent à modifier la séquence d'initialisation du robot en introduisant une procédure de

Composant / Fonction/ Message	**Mode de défaillance** (erreur)	**Cause de la défaillance**	**Effets** : a. Niveau local b. Niveau supérieur c. Niveau système	Risque			**Moyens de détection possibles (en ligne)** a. Mode défaillance b. Effets	**Solutions possibles** : a. Prévention b. Protection c. Autres actions d. Remarques
				Gravité	Occurrence	Risque		
Capteur de position *Lire la position*	Valeur nulle ou constante	Défaillance interne capteur, ou connectique	a. Calcul erreur faux b. Commande fausse c. Mouvement non désiré	1	P	H	a. b. Capteur de force	a. Test des capteurs à l'initialisation b. Si force>3daN dégonfler les muscles c. Opérateur peut annuler la tâche (au pire enclencher l'arrêt d'urgence)
	Pic ou valeur aléatoire	Défaillance interne du capteur ou bruit sur la connectique	a. Calcul erreur faux b. Commande fausse c. Mouvement non désiré	1	P	H	a. Filtrage numérique (de type Kalman) b. Capteur de force	a. Test des capteurs à l'initialisation b. Ignorer la valeur, si trop d'erreur suspendre la tâche a. Opérateur peut annuler la tâche (au pire enclencher l'arrêt d'urgence)

FIG. 5.32 – AMDEC d'un capteur de position

test décrite sur le diagramme de séquence de la figure 5.35. L'objet principal, le *Contrôleur TER*, est ici considéré comme un simple composant dont le message analysé est l'envoi de la commande aux *Convertisseurs intensité/pression*. Sans détailler les modes de défaillance des différents éléments qui le composent (cf. figure 5.36), il est possible de mener une analyse comparable aux autres composants électroniques comme cela est présenté sur la figure 5.34. Cependant, l'analyse du *Contrôleur TER* concerne aussi le logiciel de l'application qui est un élément critique puisque une défaillance de ce composant peut entraîner des dommages importants (graves). Cet aspect est abordé dans la section suivante.

5.4.5 Analyse des modes de défaillance du logiciel

On décrit le logiciel de la même manière que les autres composants du système grâce à des diagrammes de classes, d'interactions et d'états. L'approche système que nous avons présentée jusqu'ici est entièrement utilisable pour la modélisation du logiciel. En effet, les objets du monde informatique représentent une vue abstraite du monde réel, et l'on modélise des objets tels que les capteurs, les actionneurs, ainsi que des objets dits de contrôle comme un *Contrôleur d'axe*. L'approche objet permet aussi d'utiliser pour la modélisation du logiciel les diagrammes d'états élaborés lors de l'analyse système. En effet, le diagramme d'états présenté précédemment sur la figure 5.26, représente les changements d'états que le contrôleur de mouvements doit intégrer. Ces remarques qui sont propres à l'utilisation d'UML montrent l'apport de ce langage dans la prévention des fautes ; la cohérence et la traçabilité des modèles sont assurées par l'ensemble de ces diagrammes utilisables depuis l'expression des besoins jusqu'à la conception du logiciel.

Le diagramme de classes de la figure 5.37 montre une décomposition du logiciel en classes d'objets. Ce diagramme découle de la modélisation des scénarios de chaque cas d'utilisation, et nous représentons ici les premières classes d'objet que l'on peut identifier. C'est un dia-

Composant / Fonction/ Message	**Mode de défaillance** (erreur)	**Cause de la défaillance**	**Effets** : a. Niveau local b. Niveau supérieur c. Niveau système	Risque			**Moyens de détection possibles (en ligne)** a. Mode défaillance b. Effets	**Solutions possibles** : a. Prévention b. Protection c. Autres actions d. Remarques
				Gravité	Occurrence	Risque		
Convertisseur Intensité/Pression *Fournit et régule la pression*	Pression nulle	Défaillance interne	a. Muscles dégonflés b. Le robot peut basculer s'il se trouve dans certaines configurations (à déterminer) c. Stress patient	3	P	H	a. Capteur de pression	a. Maintenance préventive et séquence de test à l'initialisation b.
	Pression aléatoire (pic) ou dérive	Défaillance interne	a. Mouvement non désiré	1	P	H	a. Capteur de pression b. Capteur de force	a. Maintenance préventive et séquence de test à l'initialisation b. Dégonfler muscles c. Arrêt d'urgence par l'opérateur
	Fuite d'air	Défaillance interne pneumatique	a. Chute de la pression b. Mouvements non désiré, mais non dangereux c. Flot d'air et bruit	3	P	H	a. Opérateur	a. Maintenance préventive et séquence de test à l'initialisation b. Protection autour du circuit d'air c. Annulation de la tâche, changement du convertisseur
	Régulation instable (oscillations)	Défaillance interne	a. Instabilité commande b. Oscillation de la longueur des muscles c. Tremblements du robot sur le patient, et bruit	3	P	H	a. Opérateur	a. Maintenance préventive et séquence de test à l'initialisation

FIG. 5.33 – AMDEC d'un convertisseur Intensité/Pression

Composant / Fonction/ Message	**Mode de défaillance** (erreur)	**Cause de la défaillance**	**Effets** : a. Niveau local b. Niveau supérieur c. Niveau système	Risque			**Moyens de détection possibles (en ligne)** a. Mode défaillance b. Effets	**Solutions possibles** : a. Prévention b. Protection c. Autres actions d. Remarques
				Gravité	Occurrence	Risque		
Contrôleur *Envoyer commande*	Commande constante (figé)	Interblocage du logiciel ou défaillance du matériel	a. Robot figé b. Stress patiente	4	P	H	a. Contrôle de l'activité à l'écran par l'opérateur ou système de type Watchdog	a. Techniques de prévention pour le développement logiciel b. Dégonfler les muscles et réinitialiser le système par le Watchdog
	Commande aléatoire (pic)	Défaillance interne du logiciel, ou de matériel	a. Mouvements non désirés	1	O	H	a. Système redondant avec voteur (voir le patron double canal)	a. Techniques de prévention de fautes pour le développement logiciel b. Solution dépendante de la technique de redondance choisie c. Arrêt d'urgence par l'opérateur
	Sortie nulle	Problème alimentation, ou défaillance matérielle ou logicielle	a. Muscles dégonflés b. Le robot peut basculer s'il se trouve dans certaines configurations (à déterminer)	4	O	I	a. Capteur de pression	a. Inclure le gonflage des muscles dans la séquence de test lors de l'initialisation b. Modifier la conception pour maintenir le robot en cas de dégonflement des muscles

FIG. 5.34 – AMDEC du Contrôleur TER

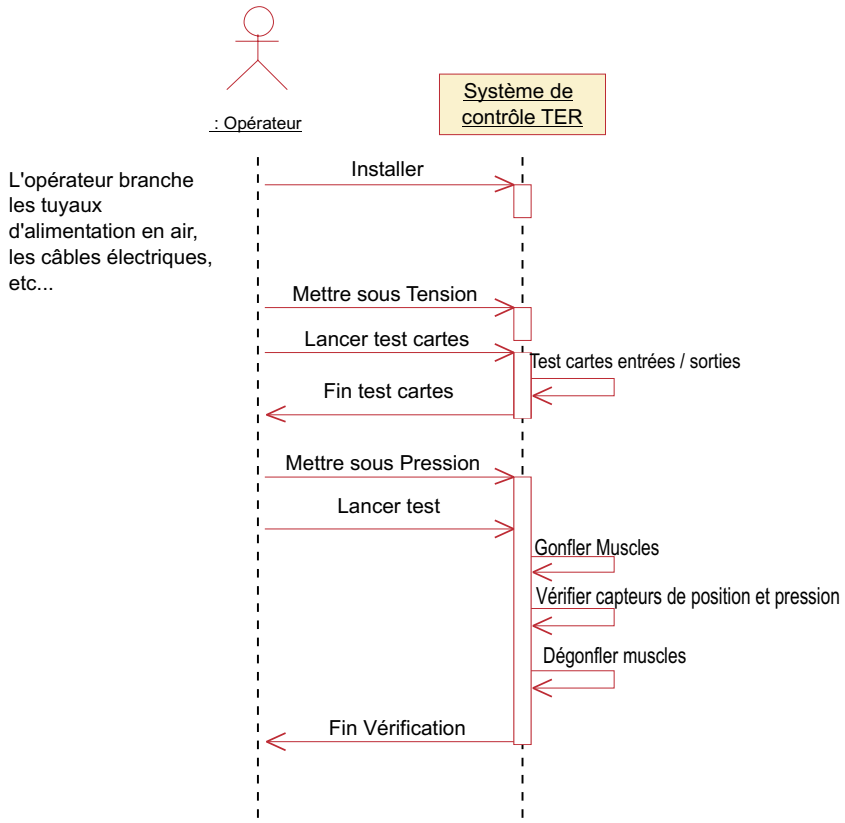

FIG. 5.35 – Diagramme de séquence des tests lors de l'initialisation (cas d'utilisation *Configurer Robot*)

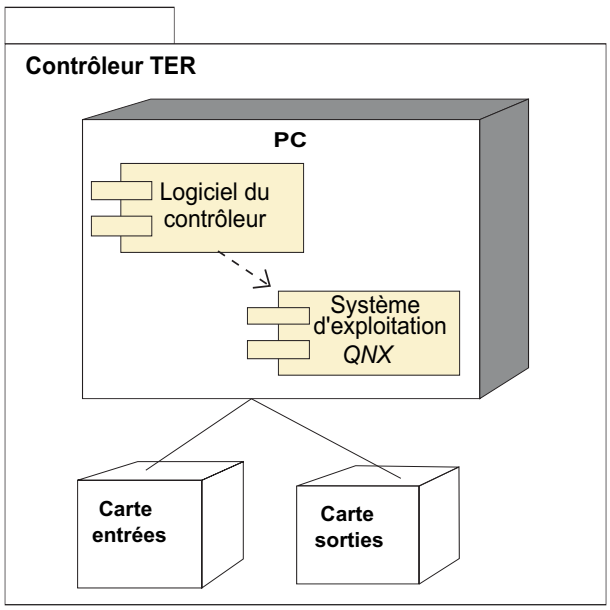

FIG. 5.36 – Contenu du *package Contrôleur TER*

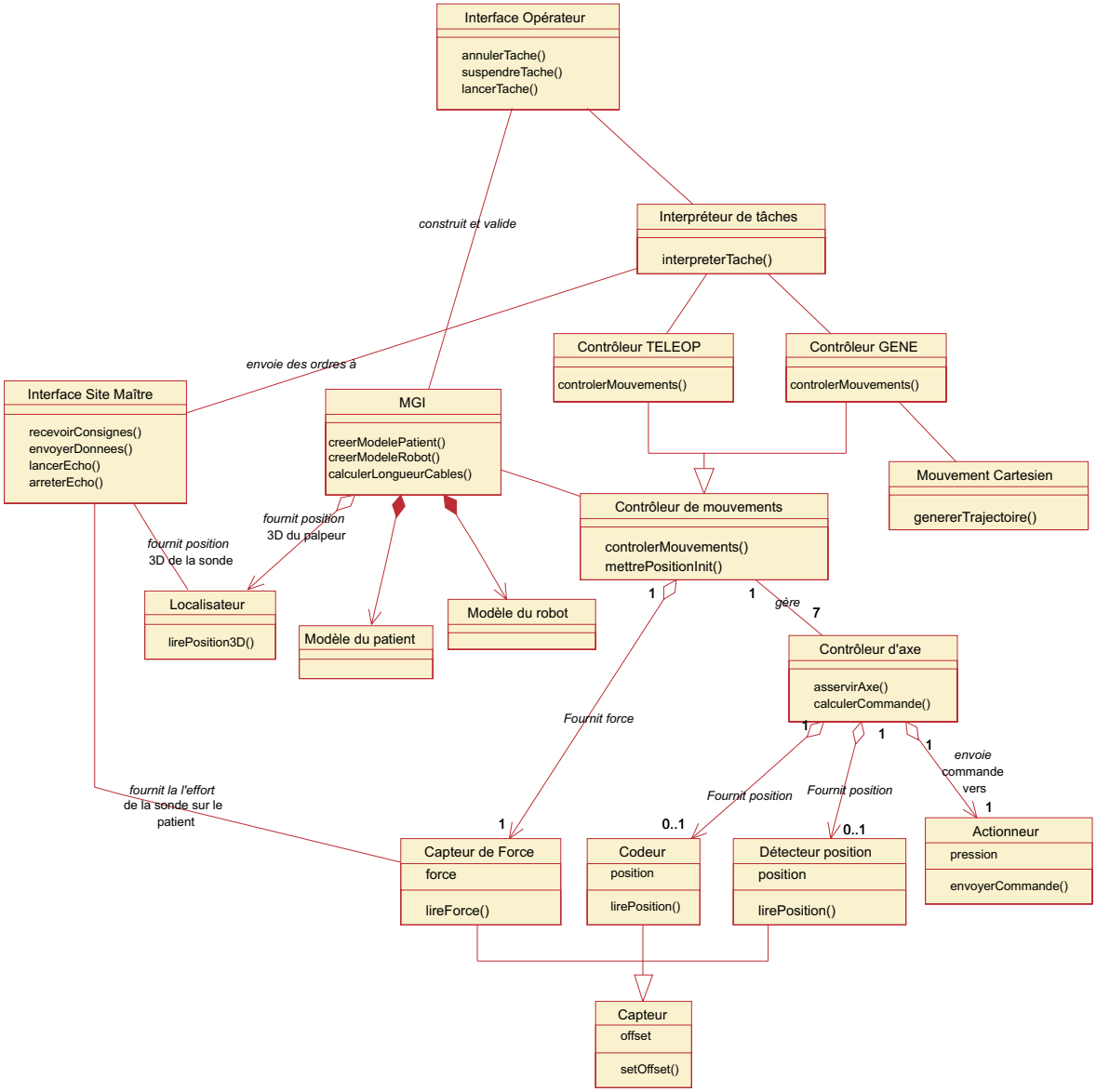

FIG. 5.37 – Diagramme de classes du logiciel

gramme d'analyse, et l'on ne fait pas apparaître les tâches, les choix de synchronisation de ces tâches, etc. Cependant l'analyse présentée dans cette section doit pouvoir être effectuée à un niveau de développement plus avancé comme lors de la conception.

Le diagramme de séquence de la figure 5.38 représente le scénario principal du cas d'utilisation *Réaliser tâche*. On exprime notamment deux contraintes temporelles qui correspondent aux périodes d'échantillonnage de la réception des consignes du site maître et de l'asservissement en position du robot. Ce diagramme montre une partie des messages échangés, et sous-entend notamment l'ensemble des messages entre la classe *Contrôleur d'axe*, et les classes *Capteur* et *Actionneur* que l'on peut représenter sur un diagramme à part (cf. figure 5.39).
La démarche d'une analyse du logiciel proposée ici, n'est pas de trouver les causes de défaillances telles que les dépassements de mémoire, les pointeurs *fous*, etc. Il s'agit de déter-

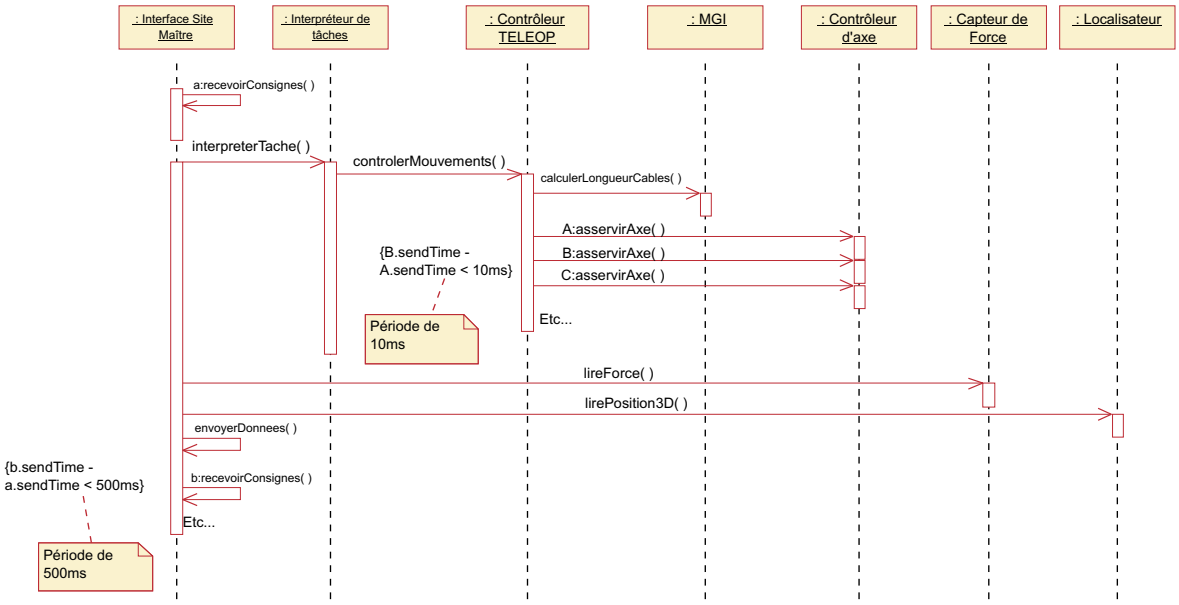

FIG. 5.38 – Diagramme de séquence du logiciel du scénario principal de *Réaliser tâche*

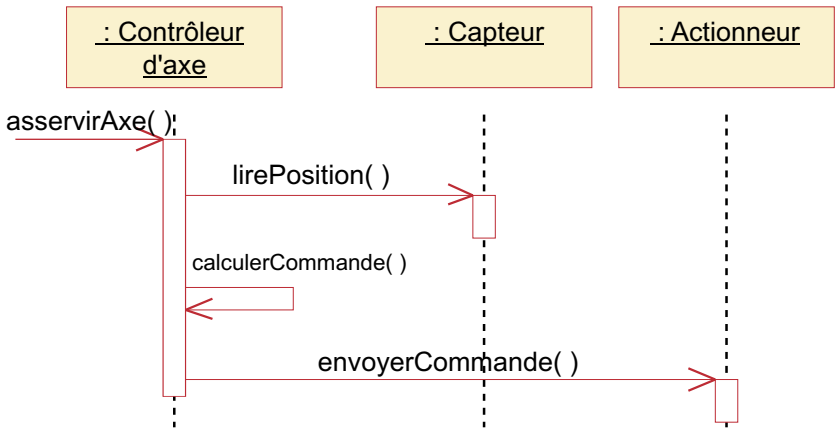

FIG. 5.39 – Diagramme de séquence du logiciel de l'asservissement en position

Composant / Fonction/ Message	**Mode de défaillance** (erreur)	**Effets** : a. Niveau local b. Niveau supérieur c. Niveau système	**Gravité**	**Solutions possibles** : a. Prévention b. Protection c. Autres actions d. Remarques
RecevoirConsignes() *Retourne les consignes*	Trop tôt ou trop tard (dans une limite < 500ms)	a. Consignes pas mises à jour b. Interprétation de la tâche identique à l'itération précédente c. Asservissement dans à la même position	5	d. Ces fluctuations sont sans effet dommageable si l'Interface site maître possède son propre flot d'exécution (une tâche est rattachée à cet objet)
	Omission (blocage)	a. Consignes pas mises à jour b. Asservissement dans la même position c. Robot figé (Stress patient)	4	b. Créer une tâche de haut niveau permettant d'annuler la tâche d'échographie
	Valeurs retournées fausses	a. Consignes fausses a. Mouvement non désiré b. Effort trop important sur le patient	1	b. Autosurveillance de la cohérence des valeurs retournées

FIG. 5.40 – AMDEC du message *recevoirConsigne()*

miner les messages critiques qui devront faire l'objet d'analyses plus approfondies lors de la conception (tests, validations, etc.), ou qui doivent l'objet d'auto-surveillance logicielle. En suivant les consignes présentées dans la section 4.3.4.5, on peut effectuer à titre d'exemple l'analyse des messages du diagramme de séquence de la figure 5.38. Ce diagramme montre les messages échangés pour réaliser le cas d'utilisation *Réaliser tâche*. Ne sont modélisés sur ce diagramme que les messages liés aux besoins fonctionnels de ce cas d'utilisation, et pas ceux liés à une surveillance (comme ceux qui concernent la mesure de la force). Il est évident que ce diagramme est une base, qu'il convient ensuite de compléter, et notamment avec des messages issus des propositions faites lors de l'analyse du risque.

Les tableaux des figures 5.40 et 5.41 montrent une analyse des modes de défaillance de deux messages comportant une contrainte temporelle. La gravité identifiée, de niveau 1, montre que ces messages sont critiques en terme de sécurité. Les moyens proposés sont alors de modifier la conception en introduisant un sous-système de surveillance des valeurs échangées, et de compléter les exigences du logiciel en terme d'ordonnancement des tâches informatiques. Ainsi, l'objet *Interface site maître* doit posséder son propre fil d'exécution, et donc être ce que l'on nomme *objet actif* (OMG, 2001). Cela montre que l'analyse du risque du logiciel permet d'aider aux choix de conception effectués par la suite.

À ce niveau du développement il est encore possible d'être exhaustif en terme d'analyse, cependant lors de la conception, l'explosion du nombre de messages rend cette analyse complexe. Il faut alors se limiter aux messages « à risque », que l'on a présentés dans la section 4.3.4.5.

Composant / Fonction/ Message	**Mode de défaillance** (erreur)	**Effets** : a. Niveau local b. Niveau supérieur c. Niveau système	**Sévérité**	**Solutions possibles** : a. Prévention b. Protection c. Autres actions d. Remarques
EnvoyerCommande() *Envoie une commande en pression*	Réalisation de l'opération trop longue	a. Le message `asservirAxe` arrive alors que `envoyerCommande` n'est pas terminé b. Commande non désirée c. Mouvement non désiré	1	d. Le temps d'exécution des opérations, `EnvoyerCommande`, `lirePosition` et `calculerCommande` doit être inférieur à 10ms (période d'échantillonnage)
	Omission (blocage)	a. Commande figée b. Robot figé c. Stress patient	4	b. Créer une tâche de haut niveau permettant d'annuler la tâche d'échographie
	Valeurs envoyées fausses	a. Commande non désirée b. Mouvement non désiré	1	b. Surveillance des données par l'objet `Actionneur` (vérification limites, écart type, etc.)

FIG. 5.41 – AMDEC du message *envoyerCommande()*

5.4.6 Analyse des arbres de fautes

Nous avons utilisé les arbres de fautes comme une technique permettant de synthétiser les analyses fournies par l'AMDEC. Comme pour l'AMDEC, il serait important d'analyser les interactions entre les événements que l'on trouve dans un arbre de fautes et les modèles UML. Ceci doit faire l'objet de prochaines études et nous présentons ici, qu'une vue partielle de l'utilisation de cette technique.

L'événement racine le plus important est une pression de la sonde sur le patient trop importante. L'arbre de fautes identifié est présenté sur la figure 5.42. Pour analyser l'événement racine, on se base sur le diagramme général de classes d'objets de la figure 5.19. On se place au niveau de l'interaction entre le système et le patient (exprimée par un lien entre la *Sonde* et l'acteur *Patient*) et on remonte la chaîne cinétique, puis électronique, et enfin informatique en identifiant à chaque fois les défaillances possibles. On introduit simultanément les erreurs humaines responsables de certains événements. Dans notre cas, en partant de la sonde, on identifie de manière séquentielle les événements suivants : chute de la sonde, défaillance du robot, défaillance des convertisseurs intensité / pression, envoie d'une mauvaise commande, et une mauvaise installation du système (incluant le *Patient*) de la part de l'*Opérateur*. Puis on raffine les événements qui peuvent résulter d'autres événements comme l'envoie d'un mauvaise commande, en s'appuyant sur les diagrammes modélisant le logiciel. Les événements qui peuvent être raffinés sont décrits dans d'autres arbres comme l'arbre de la figure 5.43 qui illustre deux erreurs possibles que peut faire l'*Opérateur,* et qui peuvent créer l'événement redouté.

L'utilisation des arbres de fautes au niveau de l'analyse ne fait en général apparaître que des portes OU. En effet, on évalue l'ensemble des événements (en général les défaillances) pouvant causer, indépendamment des autres, l'événement racine. Il existe cependant des cas

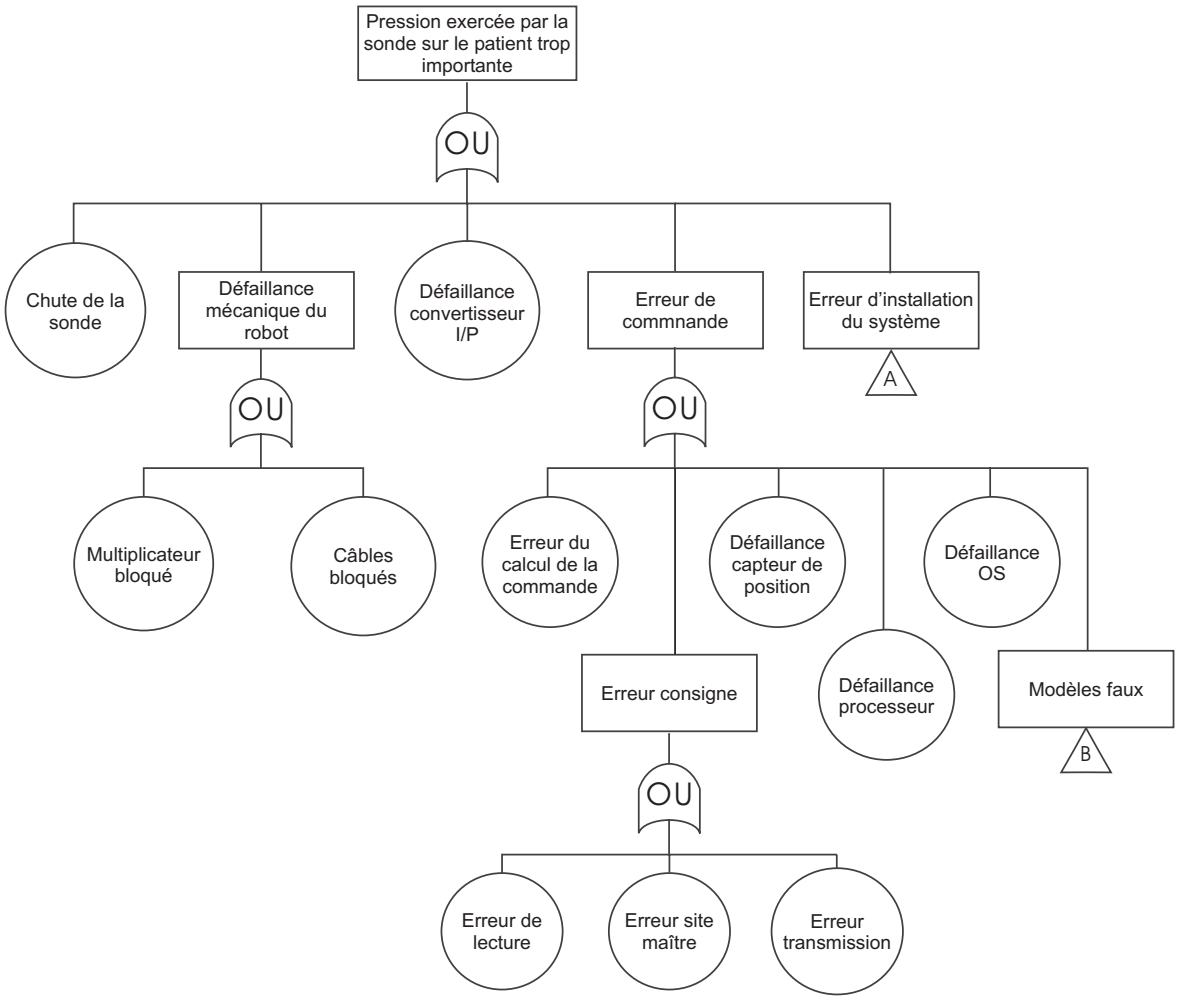

FIG. 5.42 – Arbre de fautes de l'élément racine *Pression trop importante sur le patient*

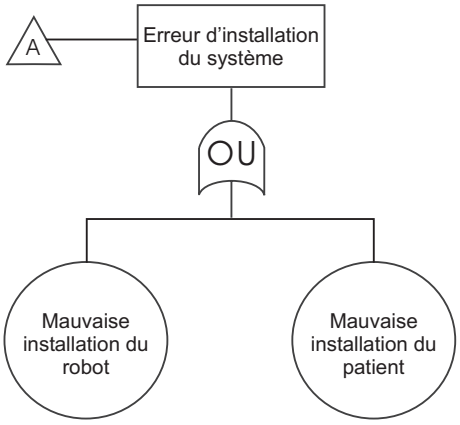

FIG. 5.43 – Arbre de fautes concernant l'installation du système

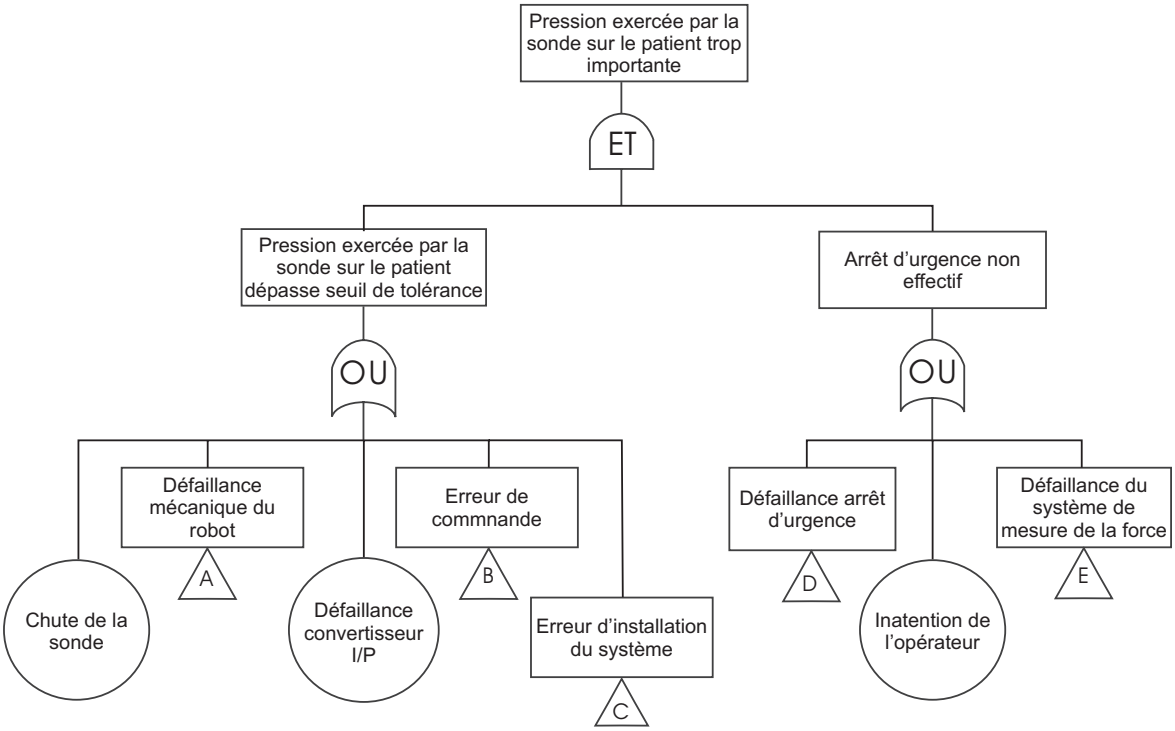

FIG. 5.44 – Arbre de fautes illustrant l'introduction du sous-système d'arrêt d'urgence

où l'on aura besoin d'utiliser la porte ET, pour exprimer la combinaison de deux événements. Dans notre cas, l'arbre que l'on peut identifier ne fait pas apparaître de porte ET. C'est lors de la réduction du risque, et de l'intégration des moyens pour les réduire que l'on fait apparaître les portes ET. À titre d'exemple, puisque le propos n'est pas de traiter de la réduction du risque, on peut exprimer le fait que l'on introduit un arrêt d'urgence que l'*Opérateur* doit enclencher après avoir consulté la force exercée sur le *Patient*. On obtient alors l'arbre de fautes de la figure 5.44. L'arrêt d'urgence peut aussi être utilisé pour d'autres situations, comme des explosions, qui sont décrites par d'autres arbres de fautes. Pour l'événement racine considéré ici, le fait d'introduire un sous-système d'arrêt d'urgence mène à compléter l'arbre de fautes avec une porte ET, et plusieurs événements. Le fait d'introduire une porte ET diminue la probabilité d'occurrence de l'événement *Pression exercée par la sonde sur le patient trop importante*, puisque cela nécessite l'occurrence simultanée des deux événements *Pression exercée par la sonde sur le patient dépasse seuil de tolérance* et *Arrêt d'urgence non effectif*. On peut noter que cette représentation fait référence à plusieurs notions qu'il faut définir pour le système comme le seuil de tolérance de la force exercée sur le *Patient*.

En avançant dans le processus de développement, les arbres de fautes sont enrichis par les sous-systèmes introduits. On peut alors prouver que certains événements ont une probabilité d'occurrence très faible par des preuves formelles (calculs mathématiques), des évidences comme la chute de la sonde mécaniquement impossible, de manière non formelle par des tests, ou en utilisant des données de constructeurs (comme pour les convertisseurs intensité/pression par exemple). Pour chaque événement, il est ensuite possible d'introduire des mécanismes de

protection pour réduire la gravité du dommage, comme l'arrêt d'urgence présenté ci-dessus, mais aussi des moyens de prévention. Comme présenté au chapitre 1, la prévention concerne la réduction de la probabilité d'occurrence d'un dommage. Dans l'analyse des arbres de fautes, il est possible d'appliquer ce principe aux événements induisant le dommage. Ainsi, on peut se poser la question : comment faire pour réduire la probabilité d'occurrence de l'événement *Inattention de l'opérateur* ? Parmi les solutions existent des moyens comme un affichage voyant de la force exercée et de la limite tolérable, des systèmes qui attirent l'attention comme des signaux lumineux ou sonores, etc. Les modifications sont alors introduites dans les modèles UML, et réinjectées dans l'analyse du risque, jusqu'à l'obtention d'un risque acceptable.

5.5 Conclusion

Ce chapitre a présenté une application de la démarche présentée au chapitre 3. L'analyse du risque réalisée a permis de reprendre l'ensemble des éléments du développement du robot esclave TER. Bien que ce chapitre ne présente pas l'analyse dans son ensemble, les principaux résultats ont étés présentés.

Ainsi, à travers une description du système et de son utilisation au moyen des diagrammes de cas d'utilisation, nous avons montré comment définir de manière non ambiguë l'ensemble des tâches allouées aux acteurs. Ceci a permis notamment de définir les capacités des acteurs humains requises pour les différentes tâches et de mettre en valeur la problématique liée à l'opérateur et sa charge de travail. La modélisation a permis par la suite d'effectuer une analyse du risque des différents messages échangés lors de la réalisation des cas d'utilisation. Bien que dans certains cas l'estimation du risque soit complexe, voire impossible, il est cependant possible d'utiliser les modèles d'erreurs présentés au chapitre 4, pour identifier les messages critiques. L'approche a été appliquée aux différents domaines que sont l'électronique, la mécanique, l'informatique, et les facteurs humains. Ces différentes activités ont mené à l'identification de dangers nécessitant un traitement. Certains dangers identifiés ont permis de définir des analyses supplémentaires à effectuer, comme par exemple l'étude de la stabilité du robot lors de la défaillance d'un muscle, ou de redéfinir des exigences de sécurité comme la nécessité d'introduire une coupure de l'alimentation en air. D'autres ont mené à la modification de l'utilisation comme l'élaboration d'une séquence de test impliquant la participation de l'opérateur. Enfin, certains dangers présentant un risque non négligeable ont mené à la modification des modèles de conception comme la présence d'un *watchdog* permettant la réinitialisation du contrôleur. Au regard de l'avancement du projet, ces différents points n'ont pu être tous implantés au sein du robot TER. Il ont donc permis dans un premier temps de valider le fait que le robot dans sa version première (premier prototype) présente un haut niveau de risque, et qu'il sera nécessaire pour la suite d'introduire les différents moyens de réduction du risque.

La principale difficulté pour réaliser cette analyse a été due à la multitude d'acteurs au sein du consortium TER, et à l'impossibilité d'obtenir certains documents de travail inaccessibles ou inexistants. Le fait que ce projet soit innovant a conduit les concepteurs vers un premier objectif, qui n'était pas la sécurité mais la validation de la structure particulière du robot actionné par les muscles artificiels. Cependant, à travers les différentes réunions du consortium TER et du travail réalisé au laboratoire TIMC/GMCAO de Grenoble, l'analyse a permis d'une part de valider notre approche mais aussi de proposer des solutions possibles pour une prochaine version du robot esclave TER.

Bien que l'analyse présentée ici ne représente qu'une itération, les diagrammes et les différents tableaux d'analyse ont été réalisés en plusieurs itérations. De plus, UML étant un langage pouvant être utilisé tout au long du processus de développement, nous pouvons penser que cette démarche d'analyse du risque est utilisable de manière itérative et incrémentale. Cependant, nous nous sommes restreints à des diagrammes de spécification, et donc ne comportant que peu d'objets et de messages par rapport à des diagrammes de conception. Pour une utilisation dans des systèmes plus importants, ou lors de la conception détaillée, l'explosion du nombre de messages rend le travail sur papier long et fastidieux, et peut induire des mises à jour lentes et chaotiques, et ainsi générer des erreurs. Or en se basant sur les diagrammes de séquence et les modèles d'erreur que nous avons proposés, des outils informatiques pourraient aider les concepteurs à gérer cette complexité et à effectuer une telle analyse du risque. Malheureusement, le fait qu'UML soit un langage semi-formel, et l'impossibilité actuelle (version 1.4) d'exécuter les modèles, rend complexe tout développement d'outils informatiques permettant d'effectuer des analyses systématiques.

Synthèse, contributions majeures et perspectives

Synthèse

La complexité grandissante des systèmes de la robotique de service, et le transfert de responsabilités de l'homme vers la machine, posent aujourd'hui le problème de la confiance que l'on peut leur accorder. Cette confiance est notamment liée au concept de sécurité, qui exprime aujourd'hui une sécurité relative. Nous l'avons en effet définie comme l'absence de risque inacceptable. Face aux difficultés pour les concepteurs à appréhender le concept de sécurité, l'objectif de cette thèse était de proposer une démarche de maîtrise de la sécurité au sein d'un processus de développement.

Dans les systèmes développés aujourd'hui, la sécurité est souvent appréhendée par l'application de techniques de la sûreté de fonctionnement. Nous avons proposé une nouvelle approche plus globale basée sur la notion de risque. La maîtrise de la sécurité dépend alors de l'activité de *gestion du risque*, dont le cœur est l'*analyse du risque*. Cette activité centrale consiste à prévoir les dangers, et à estimer les risques de dommage induits par l'utilisation du système. Elle doit être réalisée à partir d'une description du système et de son utilisation, et nous avons pour cela proposé d'utiliser la notation UML. L'utilisation de ce langage a permis de traiter des activités du domaine des facteurs humains que nous avons incluses dans l'analyse du risque. Parmi ces activités, l'allocation et l'analyse des tâches permettent de définir de manière non ambiguë les tâches des acteurs, et de décrire précisément l'utilisation du système. Les diagrammes des cas d'utilisation et de séquence permettent alors de formaliser les résultats de ces analyses, et sont directement utilisables pour les étapes suivantes de la conception du système. Les diagrammes UML permettent par la suite de réaliser une analyse des modes de défaillance des éléments du système. Nous avons montré que les éléments fondamentaux étaient les messages échangés entre tous les objets du système, dont les acteurs. Un premier travail a consisté à définir des modèles d'erreur de ces messages, puis à les appliquer pour effectuer une AMDEC système. Ces analyses ont pu être réalisées sur l'ensemble des messages échangés entre les objets électroniques, mécaniques, informatiques et les utilisateurs modélisés en tant qu'objets de type acteur.

Contributions majeures et perspectives

Cette démarche est réalisée sur la base des modèles UML du système, et elle permet de coupler le processus de développement à l'analyse du risque. Les informations, qui sont développées dans l'une ou l'autre de ces activités, sont donc cohérentes car elles se basent sur les mêmes modèles. L'un des points essentiels des cette approche est de rendre accessible l'analyse du risque aux concepteurs utilisant UML. Bien que la démarche proposée ait été appliquée aux premières phases du développement, il devrait être possible de l'exécuter de manière itérative et incrémentale, de façon à enrichir l'analyse du risque en fonction de l'évolution des modèles UML. Cet aspect reste à valider, et notamment la manière dont les techniques de l'AMDEC et celle des arbres de fautes supportent un tel cycle mérite une attention particulière.

Par rapport aux objectifs fixés, nous avons pu utiliser la terminologie liée au risque sans rencontrer d'ambiguïté, ou de terme non défini. Certains concepts n'ont cependant pas été utilisés, comme *accident* ou *incident*. Ils font en effet référence à des événements concrets, dont l'occurrence a été observée, or le travail présenté ici se place dans un cadre de prévision ; on ne manipule alors que des événements potentiels. Malgré tout, cette terminologie reste complexe, et pouvoir faire la différence entre les concepts[2] demande une certaine « culture du risque ». Or, dans un processus de développement faisant intervenir des acteurs de différents domaines, chacun possède en général son propre langage, et encore plus pour aborder les notions liées au risque qui sont nouvelles. La mise en place d'un langage commun peut alors s'avérer trop coûteux[3].

Grâce aux diagrammes UML et aux modèles d'erreur génériques que nous avons développés, il a été possible d'analyser l'ensemble du système, et notamment les erreurs humaines. C'est en ce sens que l'on peut parler d'un approche *système*. Cela a pu être fait en conservant les modèles UML dans leur forme nominale et sans modifier la notation. L'un des aspects importants du travail réalisé concerne l'utilisation de techniques connues des concepteurs. Le fait de s'appuyer sur la version standard de ce langage, devrait permettre aux concepteurs de comprendre et d'appliquer plus facilement et plus rapidement notre démarche, et cela pour l'ensemble du système.

Un point important qui reste à développer plus en détail est l'analyse du risque du logiciel. Le nombre et la complexité des messages échangés entre les objets informatiques rend en effet difficile l'application des modèles d'erreur développés dans le chapitre 3. En se basant sur des critères de sélection nous avons pu nous concentrer sur les messages dits « critiques » et analyser leurs modes de défaillance ainsi que les effets induits. Cependant, un travail important doit encore être mené sur ce thème ; un axe de recherche peut consister à réaliser des outils informatiques permettant de simuler les modes de défaillance des messages sur la base de

[2] Faire la différence entre estimation et évaluation, ou entre maîtrise et gestion n'est pas naturel...

[3] On peut alors analyser le risque lié à une telle démarche au sein d'un projet, le dommage induit étant une perte de temps et d'argent...

modèles d'erreur comme le font les simulateurs de fautes. Le principal frein à de tels travaux est l'impossibilité d'exécuter les modèles UML (dans la version 1.4), ce qui rend impossible la simulation des effets de ces erreurs.

Le dernier point relatif aux objectifs fixés au départ concerne la traçabilité en vue de la certification. Bien que cette propriété ne soit pas étudiée explicitement tout au long de ce mémoire, elle découle cependant des techniques que nous avons utilisées. En effet, l'utilisation du langage UML permet de décrire, tout au long du processus de développement, l'évolution des choix de conceptions. De plus, les tableaux de l'analyse préliminaire des dangers et de l'AMDEC, et les arbres de fautes, produisent une documentation nécessaire à la traçabilité, que l'on peut enrichir de manière itérative. Il reste cependant de nombreux travaux à faire et des outils à réaliser car les exigences ou les choix de conception à tracer relèvent parfois de propriétés dont la formulation reste un thème d'étude[4].

Maîtriser la sécurité d'une application de robotique de service est une problématique complexe, et malgré l'importance des travaux qu'il reste à mener, cette thèse propose une approche pour appréhender ce concept. En travaillant sur des techniques largement utilisées dans ce domaine, nous espérons que la démarche proposée sera facilement applicable à d'autres projets. Le travail de cette thèse est transversal à de nombreux domaines, et une des difficultés a consisté à ne pas se perdre dans un domaine au détriment des autres. Ceci a été particulièrement difficile pour le domaine des sciences cognitives, qui m'a fortement intéressé. Se maintenir à la frontière de ces domaines est d'autant plus complexe qu'il existe peu de moyens de diffusion de tels travaux, et cela se traduit par l'émergence de nouveaux groupes de travail. Ainsi, une partie de ce mémoire contribue à la mise en place du groupe *UML et Certification* initié par le LESIA[5], qui est aujourd'hui à l'origine de plusieurs thèses en cours et à venir.

[4]Comme par exemple les contraintes temps réel

[5]http ://www.lesia.insa-tlse.fr/UML-certification

Annexe A

La notation UML

Cette annexe présente les principaux diagrammes d'UML. Les notations présentées ici se limitent à ce qui est utilisé dans ce mémoire.

A.1 Concepts transversaux

Les concepts de *package*, de dépendance, de note et de stéréotype peuvent être utilisés dans tous les types de diagramme. Le terme de paquetage est parfois utilisé au lieu de *package*.

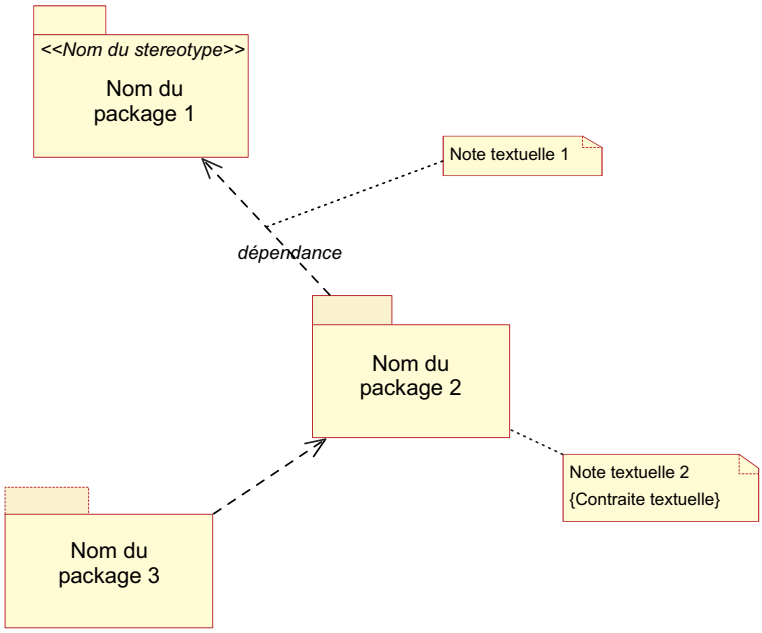

FIG. A.1 – *Package*, dépendance, note et stéréotype

A.2 Diagramme des cas d'utilisation

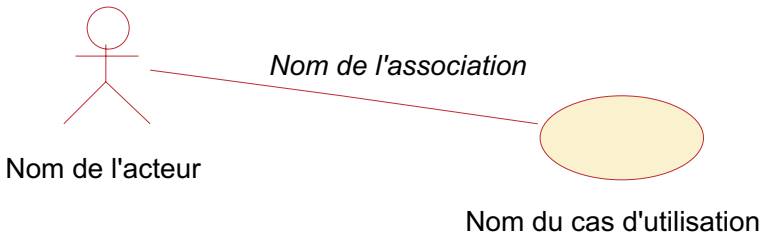

FIG. A.2 – Acteur, cas d'utilisation et association

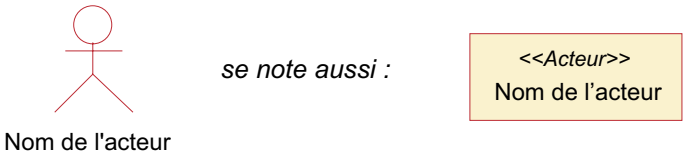

FIG. A.3 – Stéréotype «Acteur»

A.3 Diagramme de classes

FIG. A.4 – Classe

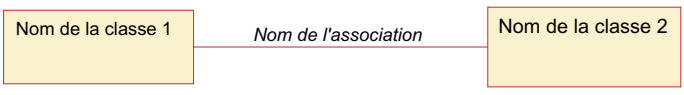

FIG. A.5 – Association

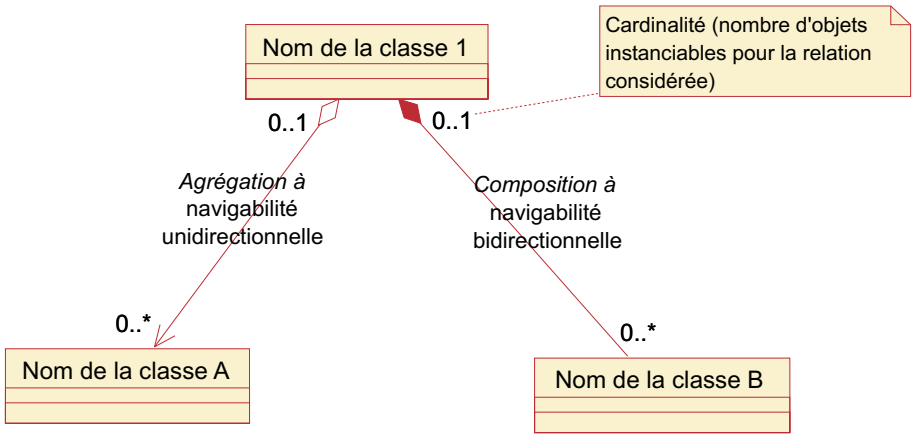

FIG. A.6 – Agrégation, composition et cardinalité

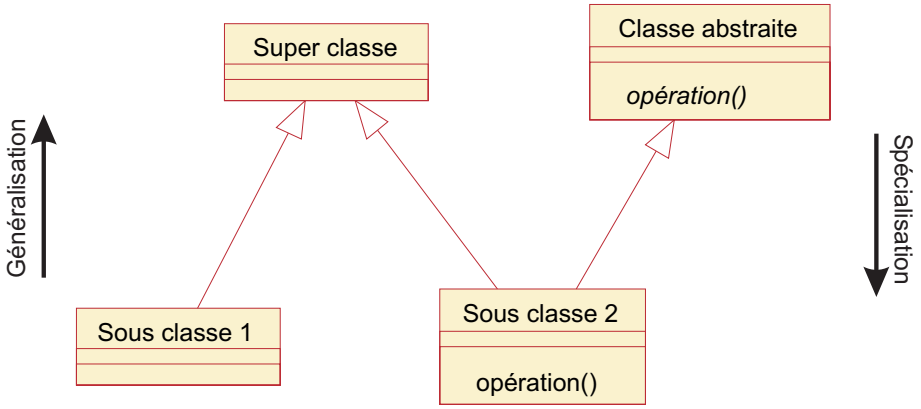

FIG. A.7 – Généralisation et spécialisation

A.4 Diagramme d'états-transitions

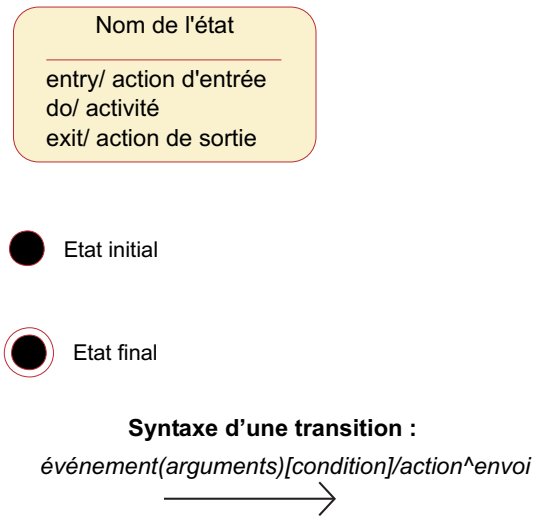

FIG. A.8 – États et transitions

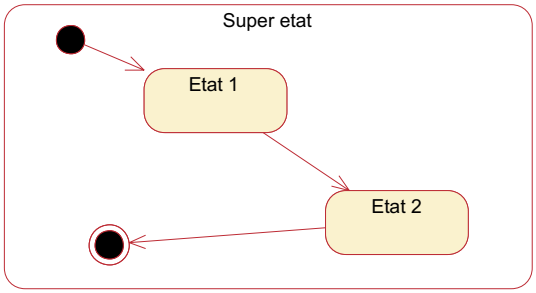

FIG. A.9 – Emboîtement d'états

A.5 Diagrammes d'interaction

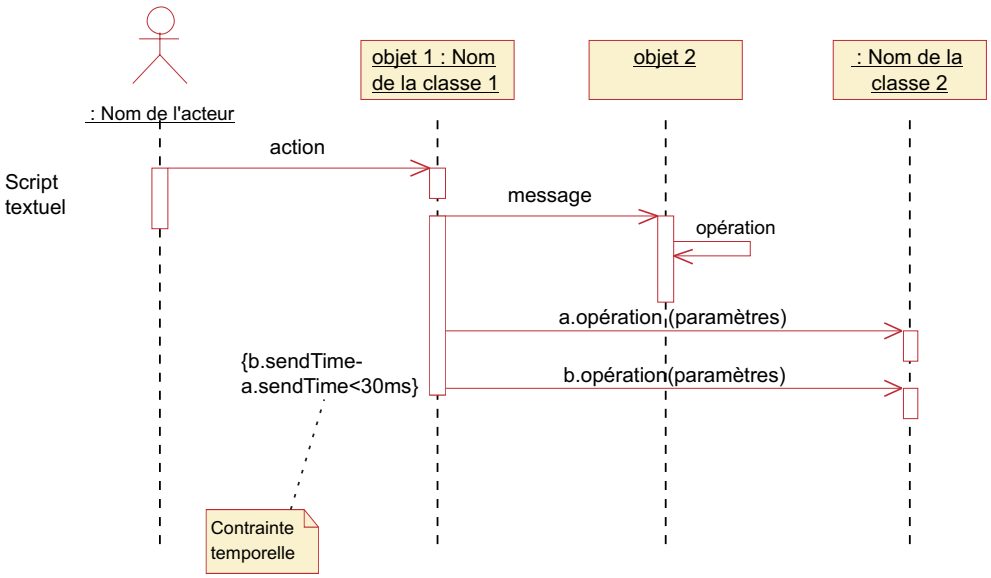

FIG. A.10 – Diagramme de séquence

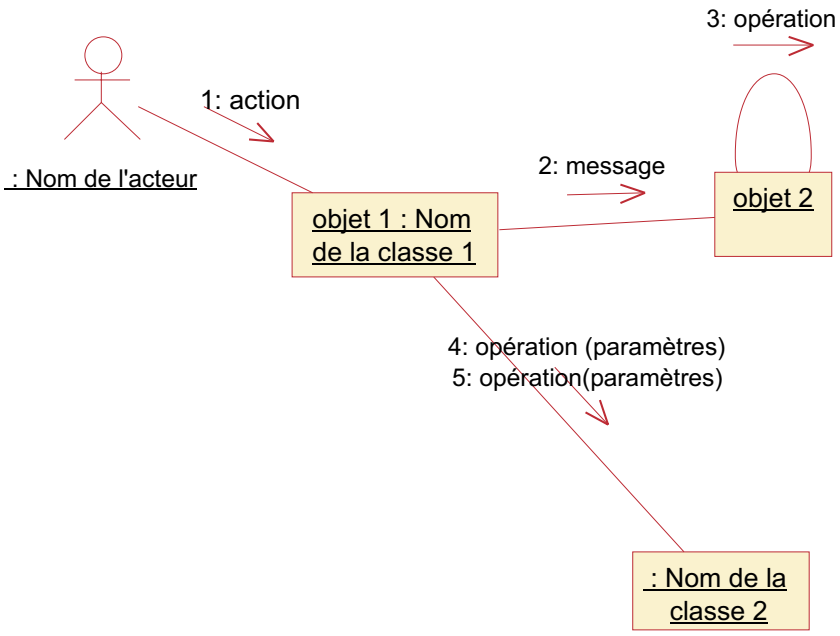

FIG. A.11 – Diagramme de collaboration

A.6 Diagramme de composants

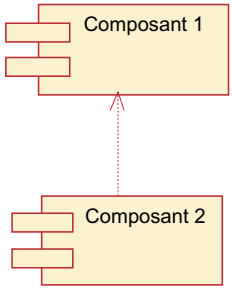

FIG. A.12 – Composant

A.7 Diagramme de déploiement

FIG. A.13 – Nœud et lien

Annexe B

Compléments à l'analyse des modes de défaillance du robot TER

B.1 Diagramme des cas d'utilisation simplifié

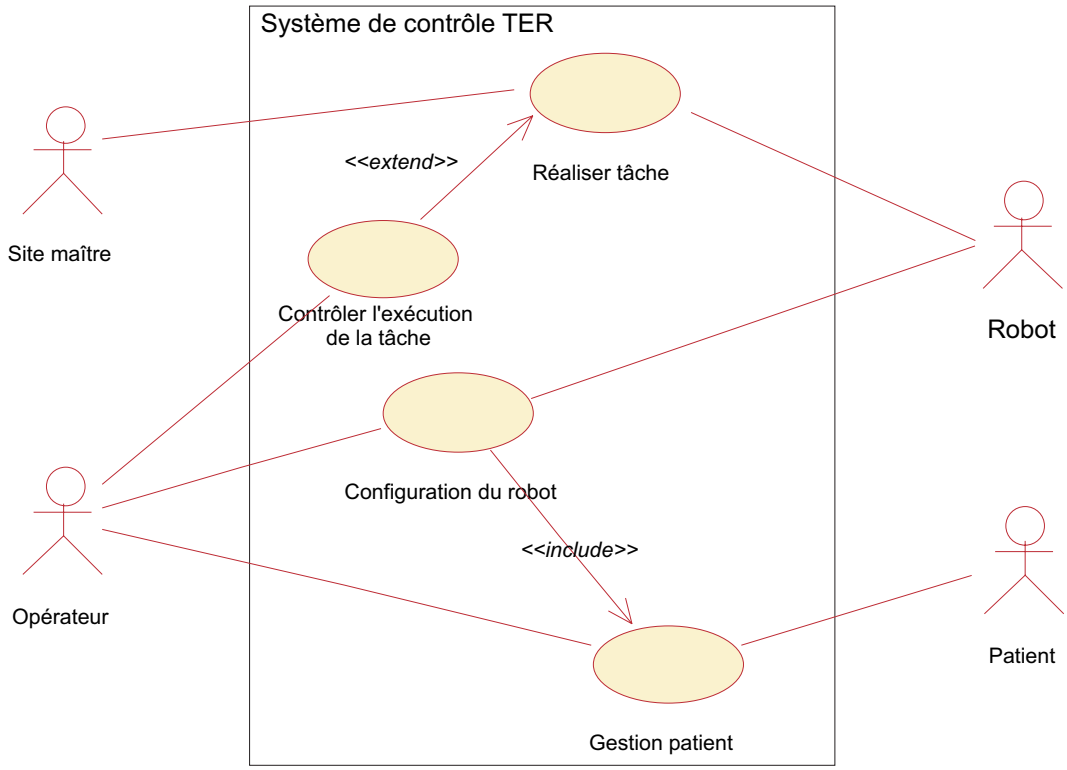

FIG. B.1 – Cas d'utilisation du système de contrôle du site esclave de TER

B.2 Diagramme de séquence principal du cas d'utilisation *Réaliser tâche*

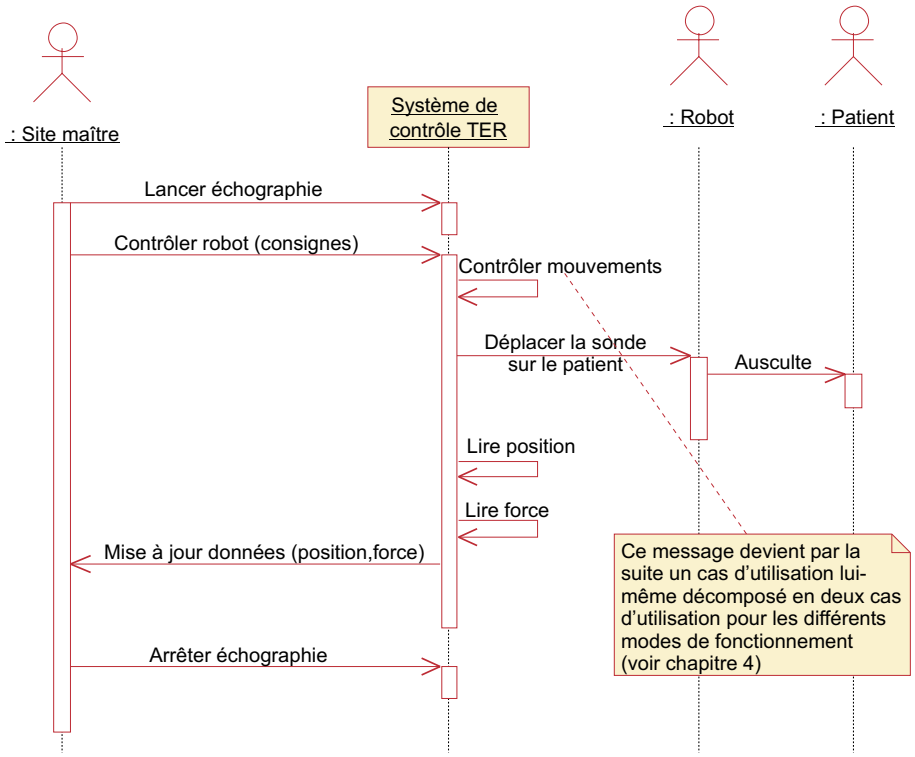

FIG. B.2 – Réalisation de la tâche d'échographie

Date :17/01/2003 **Par** : J. Guiochet			**Projet TER**					
Interaction/ Message	**Mode de défaillance** (erreur)	**Cause de la défaillance**	**Effets** a Niveau local b. Niveau supérieur c. Niveau système	Risque Gravité	Occurrence	Risque	**Moyens de détection possibles** (en ligne) a. Mode de défaillance b. Effets	**Solutions possibles** : a. Prévention b. Protection c. Autres actions d. Remarques
Réalisation de la tâche d'échographie	Mouvement du patient (erreur E.1) ou saisie du robot esclave (erreur E.1)	Mauvaise installation Stress du patient	a. Perturbation importante du mouvement du robot b. Mouvement non désiré	1			a. Surveillance de l'Opérateur b. Erreur importante	a. Consignes au patient b. Mise au repos du robot (muscles dégonflés) si erreur trop importante

FIG. B.3 – AMDEC des messages émis par le patient lors de l'interaction *Réalisation de la tâche d'échographie*

Interaction/ Message	**Mode de défaillance** (erreur)	**Cause de la défaillance**	**Effets** : a. Niveau local b. Niveau supérieur c. Niveau système	Risque Gravité	Occurrence	Risque	**Moyens de détection possibles (en ligne)** a. Mode défaillance b. Effets	**Solutions possibles** a. Prévention b. Protection c. Autres actions d. Remarques
Contrôler_robot (consignes)	Trop tard ou omission (E.3 et E.5)	Faute `Site Maître` ou faute lien	a. Les consignes ne sont pas mises à jour b. Contrôle des mouvements sur anciennes valeurs c. Déplacements par saccades	4	F	H	a. Utilisation d'une tempo, établir signal Perte connexion.	a. Utilisation protocole (fournit par France Telecom R&D) b. Figer le robot , si le délai expire mettre en position initiale c. Informer Opérateur
	Trop tôt (E.5)	Faute `Site Maître`	a. Le contrôle des mouvements n'est pas terminé b. Commande interrompue c. Déplacement brusque	3	F	H	a. Le contrôleur n'a pas terminé le contrôle de position	a. Utiliser des tâches indépendantes au sein du contrôleur b. Le contrôleur valide la fin du contrôle d'un mouvement
	Ordre incorrect du message dans la séquence (E.2)	Faute `Site Maître`	a. Comportement inconnu	1	P	H	a. Le contrôleur n'est pas en état d'attente de ce message	b. Ignorer le message c. Informer Opérateur
	Arguments erronés (nombre, type ou valeur) (E.6,E.7,E.8)	Faute `Site Maître` ou faute lien	c. Mouvement non désiré	1	F	H	a. Filtrage numérique	a. Protocole de communication (cheksum, etc.) b. Ignorer les valeurs, si trop de valeurs fausses à la suite mettre au repos c. Informer Opérateur

FIG. B.4 – AMDEC du message *Contrôler_robot* émis par le site maître

B.3 Diagramme de séquence principal du cas d'utilisation *Contrôler l'exécution de la tâche*

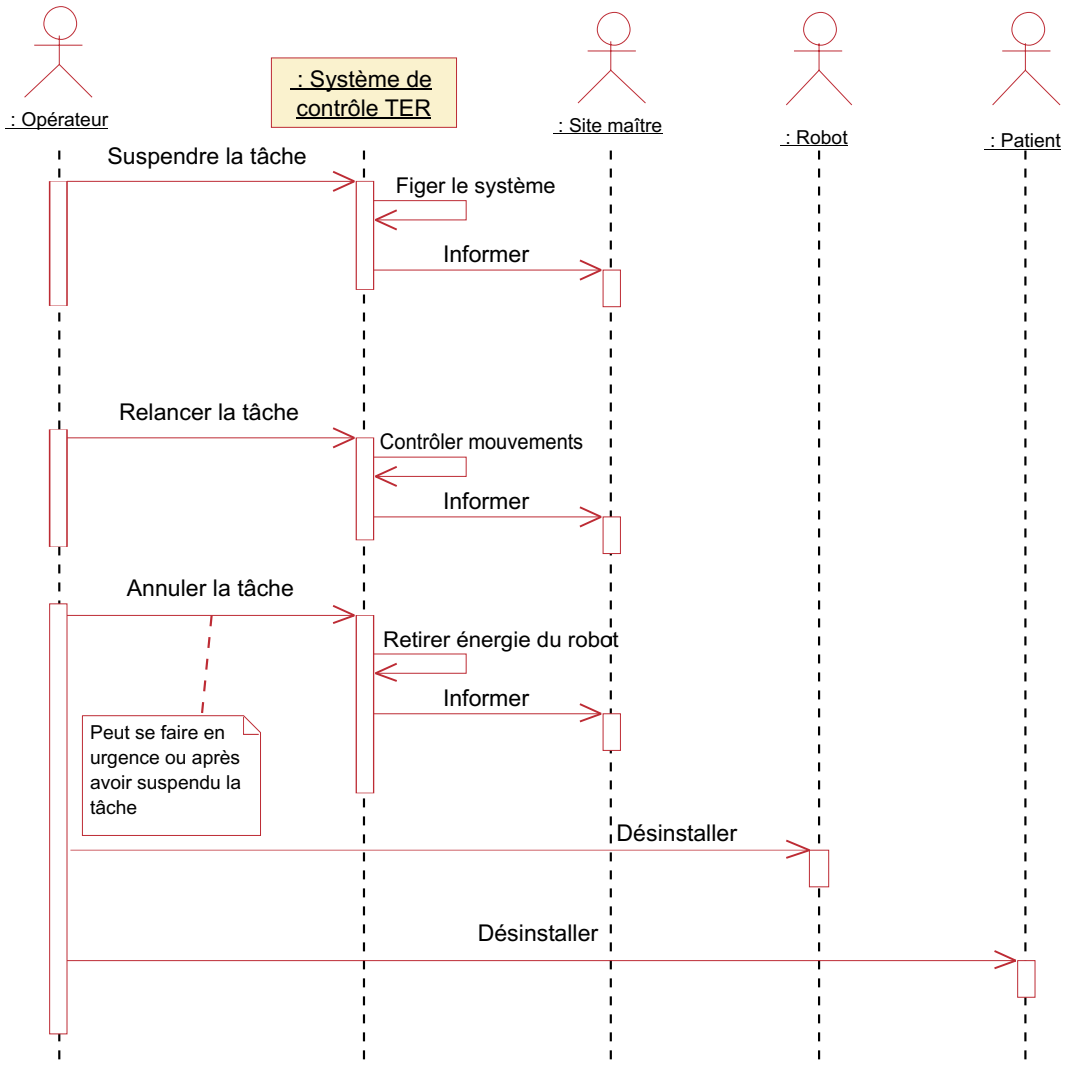

FIG. B.5 – Contrôle de l'exécution de la tâche

B.4 Diagrammes de séquence du cas d'utilisation *Configuration du robot*

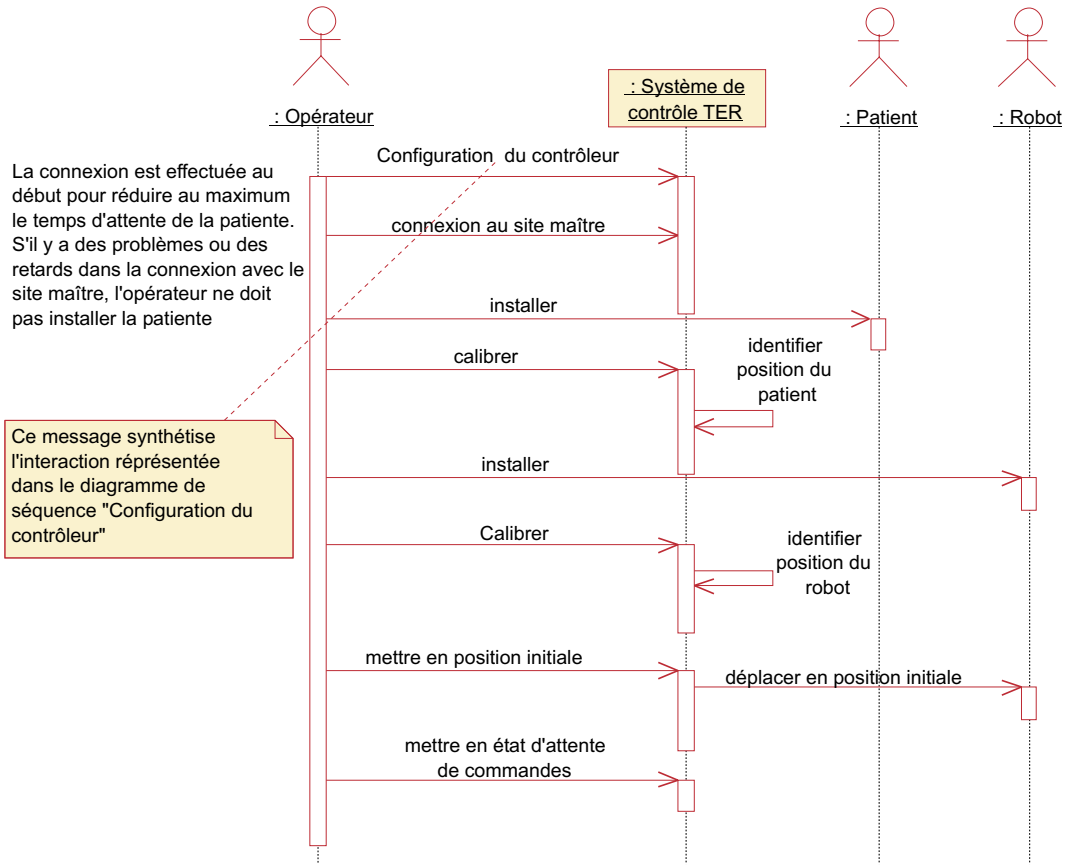

FIG. B.6 – Configuration de la tâche

Date :17/01/2003
Par : J. Guiochet

Projet TER

Interaction/ Message	**Mode de défaillance** (erreur)	**Effets** : a. Niveau local b. Niveau supérieur c. Niveau système	Risque Gravité	Occurrence	Risque	**Moyens de détection possibles** (en ligne) a Mode de défaillance b Effets	**Solutions possibles** a. Prévention b Protection c Autres actions d Remarques
Mettre en position initiale	Ordre incorrect (E.2) : avant de calibrer	a. Contrôleur non calibré b. Mouvement non désiré du robot	1				a. Affichage des étapes pour l'opérateur (validation pas-à-pas) b. Blocage si pas de calibrage
	Omission (E.3) ou la position initiale est incorrecte (E.8)	a. Au démarrage le robot est dans une position quelconque b. Mouvement non désiré du robot	1			b. Robot en dehors d'une zone d'initialisation	a. Définir une zone globale où le robot doit se trouver avant de lancer la tâche c. L'interface doit contraindre l'opérateur à respecter l'interaction

FIG. B.7 – AMDEC du message *Mettre en position initiale*

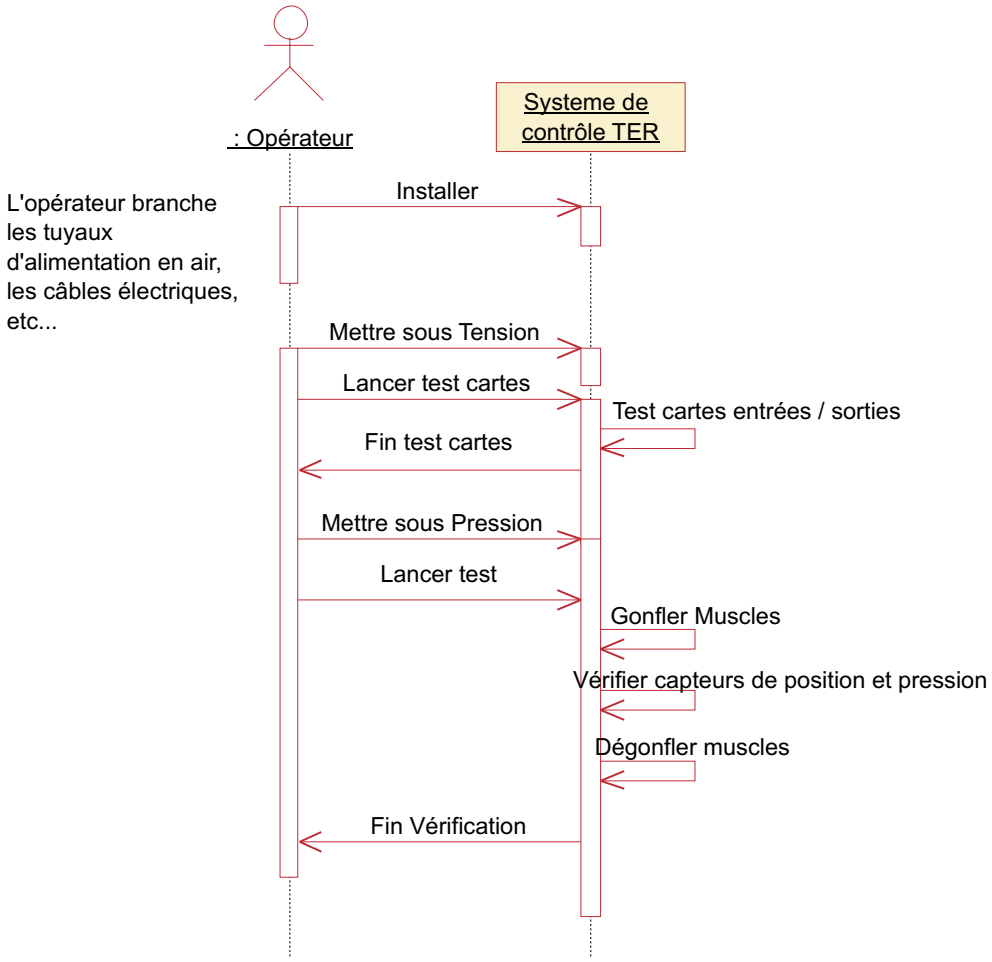

FIG. B.8 – Configuration du contrôleur

Interaction/ Message	**Mode de défaillance** (erreur)	**Effets** : a. Niveau local b. Niveau supérieur c. Niveau système	Risque			**Moyens de détection possibles (en ligne)** a. Mode défaillance b. Effets	**Solutions possibles** : a. Prévention b. Protection c. Autres actions d. Remarques
			Gravité	Occurrence	Risque		
Mettre la pression	Omission (E.3)	a. Pas d'énergie dans les muscles artificiels b. Nécessité de recommencer initialisation c. Attente du patient (stress)	4	P	I	a. Opérateur lit le manomètre	a. Guide d'utilisation, formation, étapes détaillées à l'écran b. Faire un test de gonflage des muscles lors de l'initialisation c.
	Ordre incorrect : avant mise sous tension (E.2)	Avant la fin de l'initialisation électronique les sorties ne sont pas à zéro c. Comportement inconnu (mouvements brusques des muscles)	5	P	I		a. Guide d'utilisation, formation.
	Pression réglée trop forte (E.8)	a. Destruction convertisseurs I/P	2	O	H	a. Opérateur lit le manomètre	d. La limite maximum étant fixée par la pression d'alimentation, il convient de déterminer la pression max d'entrée des convertisseurs (voir fabricant)
	Pression réglée trop faible (inférieure à alimentation des convertisseursI/P = 6bar)(E.8)	a. Pression insuffisante dans les muscles b. Pas assez de pression sur le patient c. Fonctionnement non dangereux mais chaotique (stress patient)	4	F	H	a. Opérateur lit le manomètre b. Capteur de pression	a. Indication sur le manomètre b. Faire un test de gonflage des muscles à l'initialisation et vérifier la pression par le logiciel.

FIG. B.9 – AMEDC du message *Mettre la pression*

Index

Accident, 27–29, 182
Acteur, 85, 116, 117, 140, 145–147, 159
AMDEC, 41, 42, 55, 91, 94, 106, 107, 109, 114
Analyse des arbres de fautes, 41, 94, 130, 134, 175
Analyse du risque, 31, 34, 37, 41, 52, 53, 55, 56, 91, 95, 99, 157, 163, 167
Analyse préliminaire des dangers, 107, 108, 158, 160
Analyse préliminaire du risque, 107

Cas d'utilisation, 85, 88, 90, 186
Certification, 34, 52, 54, 183
Classe, 83, 87, 186
Compliance, 47, 49, 63, 76

Danger, 25, 28, 29, 31
Diagramme d'états, 119, 155, 156, 161, 163, 164, 188
Diagramme d'interaction, 189
Diagramme de classes, 86, 186
Diagramme de collaboration, 189
Diagramme de composants, 190
Diagramme de déploiement, 121, 122, 139, 140, 190
Diagramme de séquence, 105, 112, 117, 118, 189
Disponibilité, 17
Dommage, 22–25, 28–30, 36, 37, 73, 74, 76, 115, 159

Élimination des fautes, 91, 92, 95, 115
Estimation du risque, 26, 36, 41, 43, 55, 106, 107, 114, 118, 121, 158

Événement dommageable, 29, 30
Évaluation du risque, 27, 37, 39–41, 43, 53, 55, 158

Facteur humain, 32
Fiabilité, 17, 27, 32, 48, 84, 92, 94, 95, 104

Gestion du risque, 28, 31–33, 55, 90, 121
Gravité, 23–25, 91

Incident, 29, 73, 182
Incrémental, 88
Itératif, 88

Métamodèle, 86
Maîtrise du risque, 40, 43, 44, 55, 77

Objet, 82, 83
OCL, 86, 111, 113
OMG, 84

Package, 110, 128, 146, 185
Phénomène dangereux, 28–31, 35, 41, 107
Prévention, 23, 27, 40, 73, 77, 78, 114, 115, 120, 128, 178
Prévention des fautes, 84, 91–93, 105
Prévision des fautes, 91, 93, 94
Processus Unifié, 87
Protection, 21, 40, 43, 45, 47, 73, 77, 78, 114, 115, 120, 168, 178

QoS, 92, 102, 104, 108, 129, 141–144, 151, 153, 155, 159

Réduction du risque, 40, 115, 177
Risque, 25, 26, 30, 31, 36–38

Risque acceptable, 26, 27, 37, 38, 40, 53, 158
Risque tolérable, 38

Sécurité, 17, 18, 21–23, 27, 32–34, 45, 47, 48, 50, 53, 92
Sécurité-confidentialité, 17
Sécurité-innocuité, 17
Sévérité, 24
Sûreté de fonctionnement, 17, 18, 24, 33, 90–92, 96, 97, 114
Safety, 17
Safety cases, 54
Security, 17
SIL, 93, 128, 129
Situation dangereuse, 28, 30, 31, 76, 167

Tolérance aux fautes, 40, 91, 95, 128, 132
Traçabilité, 56, 83, 97, 99, 101, 135, 169

UML, 84

Bibliographie

90/385/CEE (1990). Directive du conseil du 20 juin 1990 relative aux dispositifs médicaux implantables actifs. Journal officiel des Communautés européennes (JOCE) N°L189.

93/42/CEE (1993). Directive du conseil du 14 juin 1993 concernant les dispositifs médicaux. Journal officiel des Communautés européennes (JOCE) N°L169.

98/79/CE (1998). Directive du parlement européen et du conseil du 27 octobre relative aux dispositifs médicaux de diagnostic in vitro. Journal officiel des Communautés européennes (JOCE) N°L220.

AVSI (2002). *A guide to the certification of systems with embedded object-oriented software*. Project AVSI AFE #7 certification issues for embedded object-oriented software Version 1.4, Aerospace Vehicule Systems Institute.

Baerveldt, A. (1992a). Cooperation between man and robot : interface and safety. *IEEE International workshop on Robot and Human Communication*, p. 183–187.

Baerveldt, A. (1992b). A safety system for close interaction between man and robot. *Safety of Computer Control Systems SAFECOMP'92*, p. 25–29.

Bargar, W., Bauer, A., et Borner, M. (1998). Primary and revision total hip replacement using the ROBODOC system. *Joint Surgeons of Sacramento*. `http ://www.jointsurgeons.com/clinical.htm`.

Beevis, D., Bost, R., Döring, B., Nordø, E., Oberman, F., Papin, J.-P., Schuffel, H., et Streets, D. (1994). *Analysis techniques for man-machine systems design*. Rapport technique AC/243(Panel 8)TR/7, NATO, Canada.

Binkley, D. (1995). *C++ in safety critical systems*. Rapport technique NISTIR 5769, National Institute of Standards and Technology. `http ://hissa.ncsl.nist.gov/sw_develop/safety.html`.

Bitsch, F. (2002). Requirements on methods and techniques in perspective to approval process for railway systems. Dans *Second International Workshop on Integration of Specification Techniques for Applications in Engineering (INT 2002), Grenoble, France*.

Boehm, B. (1988). A spiral model of software development and enhancement. *IEEE Computer*, 21(5), p. 61–72.

Boitier, V. (1999). *Mise en œuvre et contrôle d'un robot SCARA à deux degrés de liberté, actionné par des muscles artificiels pneumatiques de McKibben*. Thèse de doctorat, Institut National des Sciences Appliquées de Toulouse, France.

Bondavalli, A., Cin, M. D., Latella, D., Majzik, I., Pataricza, A., et Savoia, G. (2001). Dependability analysis in the early phases of UML based system design. *International Journal of Computer Systems - Science & Engineering*, 16(5), p. 265–275.

Bondavalli, A., Majzik, I., et Mura, I. (1999a). Automated dependability analysis of UML designs. Dans *2nd IEEE International Symposium on Object-Oriented Real-time Distributed Computing (ISORC'99), Saint Malo, France*, p. 139–144. IEEE Computer Society Press.

Bondavalli, A., Majzik, I., et Mura, I. (1999b). Automatic dependability analysis for supporting design decisions in UML. Dans *4th IEEE High Assurance System Engineering Symposium (HASE99) Washington D.C., USA*, p. 64–71. IEEE Computer Society Press.

Booch, G. (1994). *Object-oriented analysis and design with applications*. Menlo Park, CA : Addison-Wesley, 2ème édition.

Booch, G., Rumbaugh, J., et Jacobson, I. (1999). *Unified Modeling Language Users Guide*. Addison Wesley Longman.

Brigeston Corporation (1987). Soft Arm ACFAS Robot System. Tokyo, Japan.

Brigeston Corporation and Taicubo Engeneering (1993). Soft Body : Advanced Painting System Unit. Tokyo, Japan.

Cain, P., Kazanzides, P., Zuhars, J., Mittelstadt, B., et Paul, H. (1993). Safety considerations in a surgical robot. *Biomedical Sciences Instrumentation, Proc. of the 30th Annual Rocky Mountain Bioengineering Symp.*, p. 291–194.

Caroll, L., Tondu, B., Baron, C., et Geffroy, J. (1998). Comparison of two significant development methods applied to the design of real-time robot controllers. Dans *IEEE International Conference on Systems, Man and Cybernetics (SMC'98), La Jolla, USA*, p. 3394–3399.

Carroll, L. (1999). *Vers la maîtrise du développement d'un contrôleur temps réel sûr de fonctionnement pour les robots manipulateurs*. Thèse de doctorat, Institut National des Sciences Appliquées de Toulouse, France.

Carroll, L., Mahout, V., Tondu, B., et Lopez, P. (1997). Sliding mode control of a 2 d.o.f. SCARA robot arm actuated by McKibben artificial muscles. Dans *Conférence IFAC de Commande des Systèmes Industriels (CIS97), Belfort, France*, p. 299–304.

Chevalley, P. et Thévenod-Fosse, P. (2001). Automated generation of statistical test cases from UML state diagrams. Dans *25th Annual International Computer Software and Applications Conference (COMPSAC'01), Chicago, USA*, p. 201–210.

Cichocki, T. et Górski, J. (2000). Failure mode and effect analysis for safety-critical systems with software components. Dans *SAFECOMP 2000*, edité par F. Koornneef et M. van der Meulen, p. 382–394. Springer-Verlag Berlin Heidelberg.

Cinquin, P. (1993). Gestes médicaux-chirurgicaux assistés par ordinateur. *Annales de Radiologie*, 36(6/7), p. 386–406.

Cosgriff, P. (1994). Quality assurance of medical software. *Journal of medical engineering and technology*, 8(1), p. 1–10.

Coste-Manière, E., Adhami, L., Bondyfalat, D., et Dowek, G. (2002). Formal methods for safe integration in medical robotics. Dans *Proc. of the 2nd IARP IEEE/RAS joint workshop on Technical Challenge for Dependable Robots in Human Environments, Toulouse, France*, p. 217–227. `http ://www.laas.fr/drhe02/preprints.pdf`.

Daouk, M. et Leveson, N. (2001). An approach to human-centered design. *Workshop on Human Error and System Development, Linkoping, Suède*. `http ://sunnyday.mit.edu/safety`.

Dario, P., Guglielmelli, E., Allotta, B., et Carrozza, M. (1996). Robotics for medical applications. *IEEE Robotics and Automation Magazine*, 3(3), p. 44–56.

Daugherty, G., Haverkamp, D., Richard, R., Bradford, R., et Statezni, D. (2002). *Response to realtime safety critical RFI*. Rapport technique orbos/2000-01-18, Rockwell Collins Advanced Technology Center.

Davies, B. (1993). Safety of medical robots. *ICAR'93*, p. 311–313.

Davies, B., Harris, S., Lin, W., Hibberd, R., Middleton, R., et Cobb, J. (1997). Active compliance in robotic surgery - the use of control as a dynamic constraint. *Proceedings of the Institution of Mechanical Engineers*, 211(4), p. 285–292.

Degoulange, E., Urbain, L., Caron, P., Boudet, S., Gariepy, J., Pierrot, F., et Dombre, E. (1998). Hippocrate : an intrinsically safe robot for medical applications. *1998 IEEE/RSJ International Conference on Intelligent robots and Systems*, 2, p. 959–964.

Delnondedieu, Y. (1997). *Un robot à sécurité passive en réponse aux problèmes d'ergonomie et de sécurité en Robotique médicale*. Thèse de doctorat, Institut National Polytechnique de Grenoble, France.

Dhillon, B. (1991). *Robot Reliability and Safety*. Springer-Verlag.

Dhillon, B. et Anude, O. (1993). Robot safety and reliability : a review. *Microelectronics and Reliability*, 33(3), p. 413–429.

Dhillon, B. et Fashandi, A. (1997). Safety and reliability assessment techniques in robotics. *Robotica*, 15, p. 701–708.

DO178B/ED-12 Revision B (1992). Software considerations in airbone systems and equipment certification. Requirement and Technical Concepts for Aviation (RTCA) Inc.

Dombre, E., Poignet, P., Pierrot, F., Duchemin, G., et Urbain, L. (2001). Intrinsically safe active robotic systems for medical applications. Dans *Proc. of the 1st IARP/IEEE-RAS Joint Workshop on Technical Challenge for Dependable Robots in Human Environments, Seoul, Korea*.

Douglass, B. (1998). *Safety-critical systems design*. Rapport technique, i-Logix, Inc. `http ://www.nohau.se/articles/pdf/safcritdes.pdf`.

Douglass, B. (1999). *Doing hard time : developping real-time systems with UML, objects, framewoks and patterns*. Object Technology Series. Addison-Wesley.

EN 1441 (1997). Medical devices - risk analysis. CEN, European Committee for standardization.

E.P.W. (1984). Rubber muscles take robotics one step further. *Rubber Development*, 37(4), p. 117–119.

Eriksson, H. et Penker, M. (2000). *Business modeling with UML : business patterns at work*. John Wiley and Sons, Inc.

Eskiizmirliler, S., Forestier, N., Tondu, B., et Darlot, C. (2002). A model for the cerebellar pathways applied to the control of a single-joint robot arm actuated by mc kibben artificial muscles. *Biological Cybernetics*, 86, p. 379–394.

Felciano, R. (1995). Human error : designing for error in medical information systems. Invited talk in the journal club « Human Error as a Tool in Design Health Care Information Systems ». `http ://www.smi.stanford.edu/people/felciano/research/humanerror/`.

Ferreira, L. L. et Rubira, C. M. F. (1998). Reflective design patterns to implement fault tolerance. Dans *OOPSLA'98 Workshop #13, Vancouver, CA*. `http ://www.csg.is.titech.ac.jp/ chiba/oopsla98/proc/`.

Food and Drug Administration (2000). *Medical device use-safety : incorporating human factors engineering into risk management*. Rapport technique, U.S. Departement of Health and Human Service. `http ://www.fda.gov/cdrh/humfac/1497.pdf`.

Gabbar, H., Suzuki, K., et Shimada, Y. (2001). Design of plant safety model in plant enterprise engineering environment. *Reliability Engineering and System Safety*, 73, p. 35–47.

Gamma, E., Helm, R., Johnson, R., et Vlissides, J. (1995). *Design patterns : elements of reusable object-oriented software*. Addison Wesley Longman, MA.

Geffroy, J. et Motet, G. (1998). *Sûreté de fonctionnement des systèmes informatiques*. Paris : InterEditions.

Geffroy, J. et Motet, G. (2002). *Design of Dependable Computing Systems*. Kluwer Academic Publishers.

Glauser, D., Flury, P., Burckhardt, C., et Kassler, M. (1993). Mechanical concept of the neurosurgical robot Minerva. *Robotica*, 11(6), p. 567–575.

Gomaa, H. (2000). *Designing concurrent, distributed, and real-time applications with UML*. Object Technology Series. Addison-Wesley.

Greenhill, S. (1993). The digit muscle. *Indust. Robot*, 20(5), p. 29–30.

Górski, J. et Nowicki, B. (1997). Object oriented safety monitor synthesis. Dans *Third International conference on reliability, quality and safety of software intensive systems (ENCRESS'97), Athens, Greece*, edité par D. Gritzalis, p. 121–133. Chapman and Hall.

Górski, J., Nowicki, B., et Wardzinski, A. (1996). Holistic and partial system models in safety analysis. Dans *International conference on probabilistic safety assessment (PSA'96), Park City, USA*, tome 2, p. 1301–1309.

Guerraz, A. (2002). *Etude du télégeste médical non invasif utilisant un transducteur gestuel à retour d'effort*. Thèse de doctorat, Université Joseph Fourier de Grenoble.

Guiochet, J. (2001). *La sécurité des robots*. Rapport technique 060401, INSA/LESIA, Toulouse, France.

Guiochet, J., Motet, G., Tondu, B., et Baron, C. (2003a). La sécurité des systèmes de la robotique médicale. *Le travail humain*. Soumis en seconde lecture après corrections.

Guiochet, J., Tondu, B., et Baron, C. (2001). Safety analysis and integration for robotic systems - application to medical robot for tele-echography. Dans *Proc. of the IASTED International Conference on Robotics and Applications (RA'01), Tampa, USA*, edité par M. Hamza, p. 158–162. ACTA press.

Guiochet, J., Tondu, B., et Baron, C. (2003b). Integration of UML in human factors analysis for safety of a medical robot for tele-echography. Dans *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Intelligent Robots and Systems for Human Security, Health, and Prosperty IROS 2003*. Accepted.

Guiochet, J. et Vilchis, A. (2002). Safety analysis of a medical robot for tele-echography. Dans *Proc. of the 2nd IARP IEEE/RAS joint workshop on Technical Challenge for Dependable Robots in Human Environments, Toulouse, France*, p. 217–227. `http ://www.laas.fr/drhe02/preprints.pdf`.

Hannaford, B. et Winters, J. (1990). *Multiple muscle systems : biomechanics and movement organization*, chapitre 7, Actuator properties and movement control : biological and technological models, p. 101–120. Springer-Verlag.

Harel, D. (1987). Statecharts : a visual formalism for complex systems. *Science of Computer Programming*, 8, p. 231–274.

Hitz, M. et Kappel, G. (1998). Developing with UML - Some pitfalls and workarounds. Dans *The Unified Modeling Language, UML'98 - Beyond the Notation. First International Workshop, Selected Papers*, edité par J. Bézivin et P.-A. Muller, tome 1618, p. 9–20. Mulhouse, France : Springer.

HSE (1999). *Reducing risk, protecting people*. Rapport technique Discussion Document, Health and Safety Executive, UK. `http ://www.hse.gov.uk`.

HSE (2001a). *Marine risk assessment*. Rapport technique 2001/063, Health and Safety Executive, UK. `http ://www.hse.gov.uk`.

HSE (2001b). *Proposed framework for addressing human factors in IEC 61508*. Rapport technique 373/2001, Health and Safety Executive, UK. `http ://www.hse.gov.uk`.

IEC 60300-3-9 (1995). Dependability management - Part 3 : Application guide - Section 9 : Risk analysis of technological systems. International Electrotechnical Commission.

IEC 61508 (2001). Functional safety of electrical/electronic/programmable electronic safety-related systems. International Electrotechnical Commission.

Ikuta, K. et Ishii, H. (2003). Safety evaluation method of design and control for human-care robot. *The International Journal of Robotics Research*, 22(5), p. 281–297.

Immega, G. (1986). ROMAC muscle powered robots. Dans *Proc. RI-SME Conf. Robotics Research, Scottdale, AZ*, p. 18–21.

Inoue, K. (1988). Rubbertuators and applications for robots. Dans *Proceedings of the fourth Int. Symp. on Robotics Research, Cambridge*, p. 57–63.

INRS (1998). *Sites robotisés - Guide technique de sécurité*. Rapport technique ND 1728-135-89, Institut National de Recherche et de Sécurité, Paris.

ISO 14971 (2000). Dispositifs médicaux - Application de la gestion des risques aux dispositifs médicaux. International Organization for Standardization.

ISO 9000-3 (1997). Partie 3 : Lignes directrices pour l'application de l'ISO 9001 :1994 au développement, à la mise à disposition, à l'installation et à la maintenance du logiciel. International Organization for Standardization.

ISO/CEI Guide 51 (1999). Aspects liés à la sécurité - Principes directeurs pour les inclure dans les normes. International Organization for Standardization.

Jacobson, I. (1992). *Object-oriented software engineering : a use case driven approach*. Addison-Wesley.

Jacobson, I., Booch, G., et Rumbaugh, J. (1999). *The Unified Software Development Process*. Addison Wesley Longman.

Jannin, P., Raimbault, M., Morandi, X., et Gibaud, B. (2001). Modeling surgical procedures for multimodal image-guided neurosurgery. Dans *MICCAI2001, Utrecht, The Netherlands*, tome 2208 de *Lecture Notes in Computer Science*, p. 139–152. Springer Verlag.

Johannessen, P., Grante, C., Alminger, A., Eklund, U., et Torin, J. (2001). Hazard analysis in object oriented design of dependable systems. Dans *2001 International Conference on Dependable Systems and Networks, Göteborg, Sweden*, p. 507–512.

Johnson, W. (1980). *MORT safety assurance systems*. New York : Deker, Marcel Incorporated.

Karlsson, J. (1999). *Les ventes de robots sont en plein essor en Europe et en Amérique du nord mais s'effondrent au Japon et en Asie*. Communiqué de presse ece/stat/99/2, ONU-CEE. `http ://www.unece.org/press/99stat2f.htm`.

Khodabandehloo, K. (1996). Analyses of robot systems using fault and event trees : case studies. *Reliability Engineering and System Safety*, 53, p. 247–264.

Klafter, R., Chmielewski, T., et Negin, M. (1989). *Robotic engineering : an integreted approach*. New Jersey - USA : Prentice Hall.

Klute, G. et Hannaford, B. (1998). Fatigue characteristics of mckibben artificial muscle actuators. Dans *Proc. IROS'98, Victoria B.C., Canada*, p. 1776–1786.

Kruchten, P. (1999). *The Rational Unified Process - an introduction*. Object Technology Series. Addison-Wesley.

Kumamoto, H., Soto, Y., et Inoue, K. (1986). Hazard identification and safety assessment of human-robot systems. *Engineering Risk and Hazard Assessment*, 1, p. 61–80.

Laible, U., Bürger, T., et Pritschow, G. (2001). A fail-safe dual channel robot control for surgery applications. *Proceedings of SAFECOMP01, Springer-Verlag Berlin Heidelberg*, p. 75–85.

Laprie, J.-C. (1992). *Dependability : Basic concepts and terminology in English, French, German, Italian and Japanese*, tome 5 de *Dependable Computing and Fault Tolerance*. Austria : Springer-Verlag.

Laprie, J.-C., Arlat, J., Blanquart, J.-P., Costes, A., Crouzet, Y., Deswarte, Y., Fabre, J.-C., Guillermain, H., Kaâniche, M., Kanoun, K., Mazet, C., Powell, D., Rabéjac, C., et Thévenod, P. (1995). *Guide de la sûreté de fonctionnement*. Toulouse, France : Cépaduès - Éditions.

Leplat, J. (1985). *Erreur humaine, fiabilité humaine dans le travail*. Paris : Armand Colin.

Leveson, N. (1995). *Safeware - System safety and computers*. University of Washington : Addison-Wesley.

Majzik, I. et Bondavalli, A. (2000). Automatic dependability modeling of systems described in UML. Dans *9th International Symposium on Software Reliability Engineering (ISSRE'98), Paderborn, Germany*, edité par R. Chillarege, tome 2, p. 4–7. Th. Illgen, IEEE Computer Society.

Manhes, S. (1998). Les patterns métier : extraction dans l'existant logiciel. Rapport de DEA, Université de Nantes, Faculté des Sciences et des Techniques.

Mersiol, M., Mazet, C., Guillerman, H., et Waeselynck, H. (1998). Human dependability in complex system : an issue of task consistency and task allocation. *International Conference on Probabilistic Safety Assessment and Management (PSAM'4)*, 4, p. 2693–2698.

MIL-STD-1629A (1980). Procedures for performing a Failure Mode, Effects and Criticality Analysis. Military Standard.

Morita, T., Iwata, H., et Sugano, S. (1999). Development of human symbiotic robot : WENDY. *Proceedings 1999 IEEE International Conference on Robotics and Automation*, 4, p. 3183–3184.

Morita, T. et Sugano, S. (1997a). Development of an anthropomorphic force controlled manipulator WAM 10. *8th International Conference on Advanced Robotics*, p. 701–706.

Morita, T. et Sugano, S. (1997b). Double safety measure for human symbiotic manipulator. *IEEE/ASME International Conference on Avanced Intelligent Mechatronics'97*, p. 130.

Muller, P. (2000). *Modélisation objet avec UML*. Eyrolles, 2ème édition.

NF EN 46003 (1999). Systèmes qualité - Dispositifs médicaux - Exigences particulières relatives à l'application de l'EN ISO 9003. International Organization for Standardization.

Ng, W. et Tan, C. (1996). On safety enhancements for medical robots. *Reliable Engineering and System Safety*, 54(1), p. 34–45.

Noritsu, T., Tanaka, T., et Yamanaka, T. (1996). Application of rubber artificial muscle manipulator as a rehabilitation robot. Dans *5th IEEE Int. Workshop on Robot and Human Communication (ROMAN96), Tsukuba, Japan*, p. 112–117.

Nowicki, B. et Górski, J. (1998). Object oriented safety analysis of an extra high voltage substation bay. Dans *SAFECOMP'98*, edité par W. Ehrengerber, p. 306–315. Springer-Verlag.

OMG (2001). *OMG Unified Modeling Language Specification v1.4*. Rapport technique, Object Management Group. `http ://www.omg.org`.

OMG (2002a). *RFP : UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms*. Rapport technique ad/2002-01-07, Object Management Group. `http ://www.omg.org`.

OMG (2002b). *UML profile for schedulability, performance, and time specification*. Rapport technique ptc/02-03-02, Object Management Group. `http ://www.omg.org/`.

OMG (2003a). *2nd revised submission to OMG RFP ad/00-09-02 - Unified Modeling Language : Superstructure - version 2.0*. Rapport technique ad/2003-01-02, Object Management Group. `http ://www.u2-partners.org`.

OMG (2003b). *3rd revised submission to OMG RFP ad/00-09-01 - Unified Modeling Language : Infrastructure - version 2.0*. Rapport technique ad/2003-01-01, Object Management Group. `http ://www.u2-partners.org`.

OMG (2003c). *OMG Unified Modeling Language Specification v1.5*. Rapport technique formal/03-03-01, Object Management Group. `http ://www.omg.org`.

Pack, R., Christopher, J., et Kawamura, K. (1997). A rubbertuator based structure climbing inspection robot. Dans *Proc. IEEE Int. conf. on Robotics ans Automation, Albuberque, NM*, p. 1869–1874.

Pai, G. et Dugan, J. (2002). Automatic synthesis of dynamic fault trees from UML system models. Dans *Proceedings of the 13th International Symposium on Software Reliability Engineering (ISSRE'02)*.

Pap, Z., Majzik, I., Pataricza, A., et Szegi, A. (2001). Completeness and consistency analysis of uml statechart specifications. Dans *Proc. IEEE Design and Diagnostics of Electronic Circuits and Systems Workshop (DDECS'2001), Gyor, Hungary*, p. 83–90.

Papadopoulos, Y. et Maruhn, M. (2001). Model-based automated synthesis of fault trees from matlab-simulink models. Dans *Int. Conf. on Distributed Systems and Networks (DSN'2001), Gothenburg, Sweden*, p. 77–82.

Papadopoulos, Y. et McDermid, J. (1998). The potential for a generic approach to certification of safety critical systems in the transportation sector. *Reliability Engineering and Systems Safety*, 63, p. 47–66.

Pocock, S., Fields, B., Harrison, M., et Wright, P. (2001). *THEA - A Reference guide*. Rapport technique 336, University of York Computer Science. `http ://www.cs.york.ac.uk/ftpdir/reports/`.

Pr ISO 14971 (1999). Projet de norme : Dispositifs médicaux - Application de la gestion des risques aux dispositifs médicaux. International Organization for Standardization.

Pransky, J. (1997). Robodoc - surgical robot success story. *Industrial Robot*, 24(3), p. 231–233.

Prasad, H. (1988). Safety standards. *International Encyclopedia of Robotics : Applications and Automation*, p. 1428–1438. R.C. Dorf and S.Y. Nof(eds), John Wiley.

Reason, J. (1990). *Human Error*. Cambridge University Press.

Reichenspurner, H., Boehm, D., Gulbins, H., Schulze, C., Wildhirt, S., Detter, C., et Reichart, B. (2000). Three-dimensional video and robot-assisted port-access mitral valve operation. *The Society of Thoracic Surgeons*, 69, p. 1176–82.

Rierson, L. (1999). *Object-Oriented Technology (OOT) In Civil Aviation Projects : Certification Concerns*. Rapport technique, Federal Aviation Administration, Washington, D.C. `http ://av-info.faa.gov/software/Other_Documents.htm`.

Rodrigues, J. (1993). What every molder should know about robot safety and maintenance. *Plastics technology*, 39(11), p. 69–72.

Roques, P. et Vallée, F. (2002). *UML en action*. Eyrolles, 2ème édition.

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., et Lorenson, W. (1991). *Object oriented modeling and design*. Englewood Cliffs, NJ : Prentice Hall.

Sackier, J. et Wang, Y. (1994). Robotically assisted laparoscopic surgery. *Surgical Endoscopy*, 8(1), p. 63–66.

Sanchez, A. (2000). *Approche non linéaire de la commande en contraction d'un muscle artificiel pneumatique de McKibben*. Thèse de doctorat, Institut National des Sciences Appliquées de Toulouse.

Schulte, H. (1961). The characteristics of the McKibben artificial muscle. Dans *Appendix H, The Application of External Power in Prosthetics and Orthotics*, 87, p. 94–115. Washington, DC : National Academy of Sciences.

Selic, B. (2000). Fault tolerance techniques for distributed systems. The Rational Edge : e-zine for the rational community. `http ://www.therationaledge.com/content/dec_00/t_fault.html`.

Selic, B., Moore, A., Bjorkander, M., Gerhardt, M., Watson, B., et Douglass, B. (2000). *Response to the OMG RFP for Schedulability, Performance, and Time*. Rapport technique Document version 1.0, Object Management Group. `http ://www.omg.org`.

Simons, A. J. H. et Graham, I. (1999). 30 things that go wrong in object modelling with UML 1.3. Dans *Behavioral Specifications of Businesses and Systems*, edité par H. Kilov, B. Rumpe, et I. Simmonds, p. 221–242. Kluwer, Dordrecht. `http ://citeseer.nj.nec.com/simons99things.html`.

Smith-Guerin, N. (2000). *Contribution à l'aide robotisée au geste chirurgical - Nouvelle approche en ophtalmologie*. Thèse de doctorat, Institut National des Sciences Appliquées de Lyon, France.

Tah, J. et Carr, V. (2001). Towards a framework for project risk knowledge management in the construction supply chain. *Advances in Engineering Software*, 32, p. 835–846.

Tondu, B., Daidié, A., Ippolito, S., et Guiochet, J. (2003). A seven d.o.f. robot arm driven by pneumatic artificial muscles for humanoïd robots. *Advanced Robotics*. Soumis en juillet 2003.

Tondu, B. et Lopez, P. (1997). The McKibben artificial muscle and its use in actuating robot-arms showing similarities with human arm behavior. *Industrial Robot*, 24(6), p. 4432–4339.

Tondu, B. et Lopez, P. (2000). Modeling and control of McKibben artificial muscle robot actuators. *IEEE Control Systems*, 20(2), p. 15–38.

Traon, Y. L., Jéron, T., Jézéquel, J., et Morel, P. (2000). Efficient OO integration and regression testing. *IEEE Transactions on Reliability*, p. 12–25.

Troccaz, J. (1999). La robotique médicale en France. *Journées Nationales de la Recherche en Robotique*, p. 251–255.

Troccaz, J. et Delnondedieu, Y. (1996). Semi-active guiding systems in surgery. A two-DOF prototype of the passive arm with dynamic constraints (PADyC). *Mechatronics*, 6(4).

Urbain, L. et Guiochet, J. (2001). *Guide pour l'analyse de sûreté de fonctionnement du programme de Télé-Echographie Robotisé (TER)*. Rapport Interne Consortium TER 2231-852, SINTERS/LESIA, Toulouse.

Vial, D. (1997). *Etude et commande floue d'actionneurs à muscles artificiels de McKibben à la motorisation de robots-manipulateurs*. Thèse de doctorat, Institut National des Sciences Appliquées de Toulouse, France.

Vilchis, A. (2003). *Télé-échographie Robotisée*. Thèse de doctorat, Institut National Polytechnique de Grenoble.

Vilchis, A., Cinquin, P., Troccaz, J., Guerraz, A., Hennion, B., Pellissier, F., Thorel, P., Courreges, F., Gourdon, A., Poisson, G., Vieyres, P., Caron, P., Mérigeaux, O., Urbain, L., Daimo, C., Lavallée, S., Arbeille, P., Althuser, M., Ayoubi, J.-M., Tondu, B., et Ippolito, S. (2001a). TER : a system for Robotic Tele-Echography. *Lectures Notes in Computer Science, Medical Image Computing and Computer-Assisted Intervention (MICCAI'01)*, p. 326–334.

Vilchis, A., Troccaz, J., Cinquin, P., Courreges, F., Poisson, G., et Tondu, B. (2001b). Robotic tele-ultrasound system (TER) : slave robot control. Dans *IFAC Conference on Telematics Applications in Automation and Robotics, TA'2001*.

Visinsky, L., Cavallero, J., et Walker, I. (1994). Robotic fault detection and fault tolerance : A survey. *Reliability Engineering and System Safety*, 46, p. 139–158.

Wadegaonkar, A., Sunnapwar, V., et Modak, J. (1996). Development of a knowledge-based system for robot work-station safety. *Proceedings of International Manufacturing Engineering, 2000 and beyond*, p. 359–361.

Walker, I. et Cavallero, J. (1996). Failure mode analysis for a hazardous waste clean-up manipulator. *Reliability Engineering and System Safety*, 53, p. 277–290.

Worthington, J. et Burns, W. (1988). Maintenance and repair, robotic. *International Encyclopedia of Robotics : Applications and Automation*, p. 833–840. R.C. Dorf and S.Y. Nof(eds), John Wiley.

Wright, P., Fields, B., et Harrison, M. (1994). Deriving human-error tolerance requirements from tasks. *IEEE International Conference on Requirements Engineering (ICRE'94)*, 1, p. 462–467.

Wu, C. (1988). Compliance. *International Encyclopedia of Robotics : Applications and Automation*, 1, p. 192–202. R.C. Dorf and S.Y. Nof(eds), John Wiley.

Yacoub, S., Ammar, H., et Robinson, T. (2000). A methodology for architectural-level risk analysis. Dans *11th International Symposium on Software Reliability Engineering (ISSRE'2000), San Jose, CA*, p. 210–221.

Yamada, Y., Hirasawa, Y., Huang, S., et Umetani, Y. (1996). Fail-safe human/robot contact in the safety space. *5th IEEE International Workshop on Robot and Human Communication*, p. 59–64.

MAÎTRISE DE LA SÉCURITÉ DES SYSTÈMES DE LA ROBOTIQUE DE SERVICE

APPROCHE UML BASÉE SUR UNE ANALYSE DU RISQUE SYSTÈME

Résumé : Les systèmes de la robotique de service, tels que les robots médicaux, permettent de réaliser des tâches complexes en milieu humain, et s'intègrent à ce titre dans les *systèmes à sécurité critique*. Lors de la conception de ces nouvelles applications, la sécurité est souvent traitée grâce à des techniques de sûreté de fonctionnement. Nous proposons cependant une nouvelle approche plus globale, basée sur la notion de risque. L'objectif de cette thèse est de proposer une démarche aux concepteurs pour appréhender la sécurité de tels systèmes, en intégrant le concept de risque et en se plaçant à un niveau système. La maîtrise de la sécurité dépend alors de l'activité de gestion du risque dont le cœur est l'analyse du risque. Cette activité centrale se décompose en trois étapes : la description du système et de son utilisation, l'identification des dangers, et l'estimation des risques de dommages induits par l'utilisation du système. Nous proposons d'utiliser la notation UML (*Unified Modeling Language*) pour la description du système. Les modèles UML sont alors couplés avec des activités du domaine des facteurs humains, incluses dans l'analyse du risque proposée. Puis, pour les deux étapes suivantes, les interactions entre cette notation et des techniques d'analyse du risque comme l'AMDEC (Analyse des Modes de Défaillance, et de leurs Effets Critiques) et les arbres de fautes sont étudiées. Cette démarche est ensuite appliquée sur le cas concret du développement d'un robot télé-échographe actionné par des muscles artificiels de McKibben.

Mots clés : Sécurité des robots, analyse du risque, UML, robot médical, muscle artificiel.

SAFETY MANAGEMENT OF SERVICE ROBOT SYSTEMS

UML APPROACH BASED ON SYSTEM RISK ANALYSIS

Abstract : Service robot systems, as medical robots, can perform complex tasks and share their working area with humans. Therefore, they belong to safety critical systems. In nowadays development process, safety is often managed by the way of dependability techniques. We propose a new global approach, based on the risk concept in order to guide designers along the safety analysis of such complex systems. Safety depends on risk management activity, which core is risk analysis. This one consists in three steps : system definition, hazard identification and risk estimation. We first propose the use of UML (Unified Modeling Language) as the description language and we integrate human factors activities for the system definition step. Then, for the next steps, interactions of UML and risk analysis techniques such as FMECA (Failure Mode, Effects and Criticality Analysis) and FTA (Fault Tree Analysis) are studied. As an illustration of its potentiality, the proposed approach is then applied to the case study of a system for robotic tele-echography (ultrasound scan examination) actuated by artificial muscles of McKibben.

Keywords : Robot safety, risk analysis, UML, medical robot, artificial muscle.