

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique



Université Saad Dahlab, Blida
USDB

Faculté des Sciences
Département Informatique

Mémoire pour l'Obtention
Du Diplôme de Master en Informatique
Option : Ingénierie du Logiciel

Sujet:

Développement d'une application Web client pour le
contrôle d'un robot manipulateur mobile :
Application sur RobuTER/ULM

MA-004-76-1

Présenté par:
Belkacem Khiter

Promoteurs :
- Abedelfetah Hentout
- Mahdia Azouz

Soutenu devant :
Hadj-Yahia : Président
Gahbghoub : Examineur
Boumahdi : Examineur

Organisme d'accueil : Centre de Développement des Technologies Avancées

A la Mémoire de Mon Cher Père

A ma Mère, ma Femme et ma Petit fille Malak qui je Vie pour son Bonheur

A Toutes ma Famille

Remerciements

Mes premiers remerciements iront à mon encadreur, M^r *HENTOUT A*, qui a accepté de me diriger dans la réalisation de ce mémoire et qui m'a témoigné son soutien et sa confiance tout au long de cette année. J'exprime également ma sincère reconnaissance aux autres membres du jury, qui ont donné de leur temps et de leur sueur pour évaluer ce modeste travail.

Je tiens à remercier mes amis, *DRIOUCHE A*, *BOUTELAA H*, *MELAHI M* et *BERRANDJIA L* qui m'ont encouragé tout au long de cette année.

Je ne terminerai pas mes remerciements sans avoir une pensée forte pour ma femme et ma mère, qui m'ont supporté, dans tous les sens du terme et en tous moments.

Résumé

De nos jours, le besoin de contrôle des robots à distance s'avère indispensable, surtout lorsque ces robots agissent dans des environnements hostiles ou dangereux aux êtres humains. Dans ce contexte, nous nous sommes fixés comme objectif le *développement d'une application de télérobotique, via internet, d'un robot manipulateur mobile*. Ce dernier consiste en une base mobile de type *RobuTER* surmontée d'un manipulateur ultraléger *ULM*.

L'architecture de télérobotique consiste en deux parties principales. La première partie se compose de six agents hybrides, en l'occurrence (i) *Agent Superviseur*, (ii) *Agent Robot Mobile Local*, (iii) *Agent Robot Manipulateur Local*, (iv) *Agent Système de Vision*, (v) *Agent Robot Mobile Distant* et (vi) *Agent Robot Manipulateur Distant*. Cette partie est dédiée au contrôle local du robot manipulateur mobile. La deuxième partie consiste en l'application web de télérobotique.

Le travail réalisé constitue une plateforme de télérobotique du *RobuTER/ULM*. Cette dernière sera exploitée pour le développement de différentes applications telles que la saisie d'objets, le suivi de trajectoires, l'ouverture de porte, etc.

Mots clés : Télérobotique, Internet, Manipulateur mobile, *RobuTER/ULM*.

Abstract

Currently, telerobotics is indispensable especially when robots are acting in hostile or dangerous environments. In this context, we propose to develop a telerobotic application, via Internet, for remote control of mobile manipulator. This latter consists of a *RobuTER* mobile base carrying a *ULM* manipulator.

The telerobotic architecture consists of two principal parts. The first one consists of six hybrid agents (i) *Supervisory Agent*, (ii) *Local Mobile Robot Agent*, (iii) *Local Manipulator Robot Agent*, (iv) *Vision System Agent*, (v) *Remote Mobile Robot Agent* and (vi) *Remote Manipulator Robot Agent*. This part is dedicated to the local control of the mobile manipulator robot. The second part is the web application of telerobotics.

The done work consists of a platform for the remote control of the *RobuTER/ULM* mobile manipulator. It will be exploited for the development of various applications such as the grasping of objects, the trajectory following, the door opening, etc.

Keywords : Telerobotics, Internet, Mobile Manipulator, *RobuTER/ULM*.

ملخص

تعتبر الحاجة اليوم في التحكم عن بعد في الروبوتات أمر ضروري خاصة عندما يتعلق الأمر بمحيط عمل خطير بالنسبة للإنسان. في هذا السياق، فقد وضعنا نصب أعيننا هدف تطوير تطبيق للتحكم عن بعد، عبر الإنترنت، في ذراع آلي متحرك. هذا الأخير مكون من قاعدة متنقلة من نوع *RobuTER* يعلوها ذراع آلي من نوع *ULM*. هندسة التحكم عن بعد المقترحة تتكون من جزئين رئيسيين. الجزء الأول يتكون من ستة وكلاء هجينة، وهم (أ) الوكيل المشرف، (ب) وكيل قاعدة متحركة محلي، (ت) وكيل ذراع آلي محلي، (ث) وكيل نظام الرؤية، (ج) وكيل قاعدة متحركة بعيد و (ح) وكيل ذراع آلي بعيد. هذا الجزء مكرس للتحكم المحلي في الذراع الآلي المتحرك. الجزء الثاني هو تطبيق الويب في الذراع الآلي المتحرك.

العمل المنجز هو منبر لتطبيقات التحكم عن بعد في الذراع الآلي المتحرك *RobuTER/ULM*. وسيتم استغلال هذا الأخير من أجل تطوير تطبيقات مختلفة كالتقاط الأشياء، تتبع المسارات، فتح الأبواب، الخ.

كلمات مفتاحية: التحكم عن بعد في الروبوتات، الإنترنت، ذراع آلي متحرك، *RobuTER/ULM*.

Table des Matières

Introduction Générale	1
------------------------------------	---

Chapitre I

1. Introduction	3
2. Caractéristiques principales des manipulateurs mobiles	4
2.1. Non-holonomie.....	4
2.2. Redondance mécanique.....	4
2.3. Capteurs utilisés et leurs positions	4
2.4. Nombre des manipulateurs installés et leurs emplacements sur la base mobile	4
2.5. Nombre et position des roues motrices et des roues folles de la base mobile.....	4
2.6. Taille du manipulateur mobile	5
2.7. Mode de Manipulation	5
3. Domaines d'utilisation des manipulateurs mobiles.....	5
3.1. Domaine spatial.....	5
3.1.1. Mars Exploration Rover (MER).....	5
3.1.2. Sky Worker	6
3.2. Domaine manufacturier.....	6
3.3. Domaine médical.....	6
3.3.1. Domaine médical.....	6
3.3.2. Hippocrate	7
3.3.3. Robot Da-Vinci	7
3.4. Domaine agricole	8
3.5. Robotique de service	8
3.5.1. Robots domestiques.....	8
3.5.2. Robots artistes	9
3.5.3. Ouverture de porte.....	9
3.5.4. Saisie d'objet.....	10
4. Conclusion.....	10

Chapitre II

1. Introduction	11
2. Téléoopération.....	12
2.1. Téléoopération Assistée par Ordinateur (TAO).....	13
2.2. Téléoopération à Désignation d'Objectif (supervisée) (TDO)	13

2.3.	Téléopération Semi Autonome (TSA)	13
3.	Télérobotique	14
4.	Télérobotique basée sur Internet	14
5.	Architectures pour la télérobotique	14
5.1.	Architecture de référence pour la télérobotique	15
6.	Conclusion.....	20

Chapitre 3

1.	Introduction	21
2.	Modèle globale de l'architecture de télérobotique	22
3.	Modèle détaillé de l'architecture de contrôle proposée	23
3.1.	Agent Superviseur	23
3.2.	Agent Robot Mobile Local.....	24
3.3.	Agent Robot Manipulateur Local.....	25
3.4.	Agent Robot Mobile Distant	25
3.5.	Agent Robot Manipulateur Distant	26
3.6.	Extension de l'architecture à la télérobotique via <i>Internet</i>	26
4.	Modélisation du système de contrôle	27
4.1.	Modélisation des besoins	27
4.1.1.	Description détaillée du cas d'utilisation « Déplacement de la base Mobile ».....	28
4.1.2.	Description détaillée de cas d'utilisation « Mouvoir le bras Manipulateur »	28
4.2.	Structure et les liens entre les objets	29
4.3.	Diagramme d'activité.....	29
4.4.	Diagrammes de séquence	30
4.4.1.	Authentification.....	30
4.4.2.	Déplacer la base Mobile	31
4.4.3.	Mouvoir le bras manipulateur	31
4.4.4.	Communication inter-agents	32
5.	Conclusion.....	33

Chapitre IV

1.	Introduction	34
2.	Aspects d'implémentation.....	35
3.	Environnement de travail	35
4.	Protocole de communication.....	35
4.1.	Connexion entre le serveur et le robot	35
4.2.	Structure des messages échangés dans le système	35

Liste des Figures

Chapitre I

Figure 1: Manipulateur mobile <i>Spirit</i> avec son Schéma.....	5
Figure 2: Manipulateurs mobiles du projet <i>Sky Worker</i>	6
Figure 3: Manipulateur mobile <i>Kamro</i>	6
Figure 4: Robot <i>Dermarob</i>	7
Figure 5: Robot <i>Hippocrate</i>	7
Figure 6: Robot <i>Da-Vinci</i>	8
Figure 7: Manipulateur mobile <i>AgroBot</i>	8
Figure 8: Les robots <i>iRobo Q</i> , <i>EMIEW 2</i> , <i>Aibo</i> & <i>Asimo</i>	9
Figure 9: Manipulateur mobile <i>Robographe</i>	9
Figure 10: Robot <i>FANUC ROBOTICS</i>	9
Figure 11: Manipulateur mobile <i>Lias</i>	10
Figure 12: Manipulateur mobile <i>EL-E</i>	10

Chapitre II

Figure 1: Illustration de l'architecture générale d'un système de téléopération.....	12
Figure 2: Illustration d'un système de téléprésence avec un robot mobile	13
Figure 3: Classement de la TAO, TDO et TSA selon le degré d'autonomie du dispositif esclave	14
Figure 4: Architecture de référence pour la téléopération.....	15
Figure 5: Architecture du contrôle d'un manipulateur via le World Wide Web.....	16
Figure 6: Interface Utilisateur de <i>WebDriver</i>	17
Figure 7: Le robot <i>DanteII</i> dans le système <i>VEVI</i>	19
Figure 8: Interface <i>Viz</i> pour la mission <i>Mars Polar Lander</i>	19

Chapitre III

Figure 1: Vue globale d'architecture de télérobotique.....	22
Figure 2: Extension de l'architecture de contrôle proposée dans pour la télérobotique des manipulateurs mobiles via <i>Internet</i>	23
Figure 3: Agent Superviseur.....	24
Figure 4: Agent Robot Mobile/Manipulateur Local.....	25
Figure 5: Agent Robot Manipulateur Distant.....	26
Figure 6: Diagramme de Cas d'Utilisation du système de contrôle.....	27

Chapitre IV

Figure 1 : Connexion entre le serveur et le robot	35
Figure 2: Structure d'un message	36
Figure 3: Légende des différents composants	37
Figure 4: Architecture modulaire de l'ARMoL/l'ARMaL.....	38
Figure 5: Architecture modulaire de l'AS.....	40

Chapitre V

Figure 1: Manipulateur mobile <i>RobuTER/ULM</i>	43
Figure 2: page d'authentification	43
Figure 3: Interface opérateur pour le contrôle du <i>RobuTER/ULM</i>	44
Figure 4: Partie concernant la base mobile.....	44
Figure 5: Partie concernant le bras manipulateur	45
Figure 6: Temps de connexion selon le type du réseau utilisé	46
Figure 7: Les variations des coordonnées articulaires du manipulateur <i>ULM</i>	46
Figure 9: Trajectoire réelle suivie par la base mobile	47

<i>Figure 8</i> : Variations articulaires du manipulateur ULM	47
<i>Figure 10</i> : Trajectoire réelle suivie par la base mobile ainsi que l'évitement des obstacles présents.....	48

Liste des tableaux

Chapitre III

Tableau 1: Description détaillée de cas d'utilisation « Déplacement de la base Mobile »..... 28

Tableau 2 : Description Détaillée de cas d'utilisation « Mouvoir du bras manipulateur» 28

Chapitre IV

Tableau 1: Différents messages échangés entre les agents37

Chapitre V

Tableau 1: Temps de connexion selon le type du réseau utilisé45

Introduction Générale

L'histoire de la téléopération commence avec le manipulateur maître-esclave, développé par Goertz R. à l'*Argonne National Laboratory* (ANL) en 1948 [1a]. Dans ce système, conçu pour la manipulation des substances toxiques, le manipulateur maître possédait la même structure mécanique et les mêmes propriétés cinématiques que le manipulateur esclave (donc les limitations physiques de l'esclave étaient perçues naturellement par l'utilisateur). En 1954, l'ANL développa la deuxième génération de télémanipulateurs électriques à retour d'effort [1b].

Au début des années 60, les champs d'application de la téléopération s'étendent à l'exploration spatiale, sous-marine, au domaine nucléaire, etc.

L'apparition des ordinateurs et le progrès en matière de technologie de communication, notamment l'internet, ont permis le développement d'une partie majeure de la téléopération, qui est la télérobotique. Cette dernière consiste à appliquer le principe de la téléopération sur les robots.

Aujourd'hui, on fait appel à la télérobotique pour la réalisation des tâches complexes, risquées, voire impossibles (cas de l'exploration de l'espace). Ceci permet de pallier aux éventuels risques fatals. Citons, comme exemple de ces cas, l'explosion de la centrale nucléaire du *Fukushima* au JAPON en mois de mars 2011, où des robots contrôlés à distances ont été utilisés pour l'inspection et la décontamination des sites concernés. Le besoin de contrôle des robots s'avère indispensable dans de telles situations catastrophiques.

Dans ce contexte, et afin de donner des solutions appropriées au besoin précité, nous fixons comme objectif de ce travail le développement d'une *application web client pour le contrôle d'un robot manipulateur mobile*. Le robot sur lequel nous avons travaillé consiste en une base mobile de type *RobuTER* surmontée d'un manipulateur ultraléger *ULM*. Le robot est contrôlé par un PC industriel embarqué (on-board) et par un PC hôte (off-board).

Nous avons utilisé, à cet effet, une architecture multi-agents pour le contrôle du manipulateur mobile *RobuTER/ULM*. L'architecture se compose de six agents hybrides (i) *Agent Superviseur*, (ii) *Agent Robot Mobile Local*, (iii) *Agent Robot Manipulateur Local*, (iv) *Agent Système de Vision*, (v) *Agent Robot Mobile Distant* et (vi) *Agent Robot Manipulateur Distant*. Les quatre premiers agents sont installés sur le PC off-board, alors que les deux derniers sont installés sur le PC on-board.


Pour la réalisation de ce travail, nous avons étendu cette architecture à l'utilisation via Internet en développant une interface homme/robot conviviale et en intégrant un serveur web de l'architecture de contrôle.

Pour assurer une meilleure présentation du travail effectué et garantir la clarté du mémoire, outre cette introduction générale, ce mémoire est divisé en cinq chapitres et deux annexes. Chacun met en évidence une contribution particulière du travail :

- Le premier chapitre est dédié à un tour de monde sur les manipulateurs mobiles qui existent de nos jours. Il cite aussi les différents domaines d'application de ces robots.
- Le deuxième chapitre présente un état de l'art sur les aspects liés à la téléopération, à la télérobotique et aux différentes architectures de télérobotique des manipulateurs mobiles existant dans la littérature.
- Le chapitre suivant est dédié à la description de l'extension de l'architecture de contrôle, précédemment proposée, à la télérobotique via un réseau local ou internet des manipulateurs mobiles.
- L'implémentation et mise en œuvre de l'architecture télérobotique des manipulateurs mobiles sont décrites dans le chapitre quatre.
- Le dernier chapitre présente et discute les principaux résultats expérimentaux obtenus.
- Nous terminons par une conclusion du travail réalisé et nous traçons les perspectives à accomplir dans le futur.
- Le document est complété par une série d'annexes qui donnent plus de détails sur certaines parties, à savoir, les agents et système multi-agents, et l'installation et publication de l'application web sous *IIS*.

Chapitre 1

Présentation des manipulateurs mobiles



1. Introduction

Depuis plusieurs décennies, le robot manipulateur mobile suscite beaucoup d'intérêts et trouve son emploi dans diverses disciplines ou activités de la vie moderne (robots industriels, robots destinés à l'exploration de l'univers, robots agricole, robots de service, etc.). Il pose, par contre, plusieurs problèmes et soulève divers défis relatifs à l'autonomie du déplacement dans un environnement non prédéfini ou changeant et à l'exécution de tâches spécifiques demandées.

Ce premier chapitre répertorie les différentes caractéristiques particulières relatives aux manipulateurs mobiles. Le chapitre fait un tour du monde des manipulateurs mobiles réels disponibles au niveau de certains laboratoires de recherches et universités ou sur le marché (en production).

2. Caractéristiques principales des manipulateurs mobiles

Un certain nombre de caractéristiques relatives aux manipulateurs mobiles ont pu être constatées lors de cette recherche bibliographique.

2.1. Non-holonomie

Un système mécanique non-holonome est un système soumis à des contraintes sur les vitesses des différents corps qui le composent. L'exemple typique de ce genre de système est la voiture. Les roues d'une voiture doivent respecter une contrainte de roulement sans glissement sur le sol ce qui contraint leurs vitesses respectives à ne pas pouvoir évoluer de manière indépendante les unes des autres. Ces contraintes sont très fortes puisqu'elles permettent de limiter, à chaque instant, les directions du mouvement de chacun des corps du système [01].

2.2. Redondance mécanique

Un robot est redondant lorsque le nombre de degrés de liberté (ddl) de l'effecteur est supérieur au nombre de ddl de l'espace articulaire (nombre d'articulations motorisées). Cette propriété permet d'augmenter le volume du domaine accessible du robot et de préserver les capacités de déplacement de l'effecteur en présence d'obstacles.

Les manipulateurs mobiles sont, la plupart du temps, des systèmes mécaniquement redondants dans le sens où le nombre total de ddl qu'ils possèdent excède le nombre nécessaire à l'exécution des tâches de manipulation [01].

2.3. Capteurs utilisés et leurs positions

Les types de capteurs utilisés et leurs positions représentent un critère important dans le choix d'un manipulateur mobile.

La base mobile non-holonome est dotée de divers moyens sensoriels, comme le télémètre laser (se trouvant à l'arrière ou l'avant), ainsi qu'une ceinture de capteurs à ultrasons se trouvant tout autour. Ces deux types de capteurs permettent au robot de détecter les obstacles et de se déplacer dans ses environnements.

La base mobile est surmontée d'un manipulateur qui comporte, en général, une caméra au niveau de l'effecteur, un capteur d'effort ainsi que des capteurs de positions des axes du manipulateur [01].

2.4. Nombre des manipulateurs installés et leurs emplacements sur la base mobile

En général, un manipulateur mobile est composé d'une base mobile surmontée d'un seul manipulateur. Toutefois, ils existent des manipulateurs mobiles qui disposent de deux manipulateurs (Dual-arm) comme c'est le cas du robot *Kamro* [02] ou une version modifiée du robot *Target* [03].

Un autre critère très important est l'emplacement du manipulateur au dessus de la base mobile qui doit être placé de sorte à permettre l'équilibre de l'ensemble. De plus et selon le type de la tâche à accomplir par le manipulateur mobile, l'emplacement du manipulateur peut changer :

- Le manipulateur peut se trouver à l'avant de la base mobile dans le cas d'une tâche d'ouverture d'une porte par exemple [04].
- Le manipulateur peut se trouver de côté dans le cas d'une tâche de saisie d'un objet avec la base mobile en mouvement [05].

2.5. Nombre et position des roues motrices et des roues folles de la base mobile

Pour faire en sorte que le robot se déplace correctement selon la consigne, la position et le type des roues doivent être connus.

Quelques manipulateurs mobiles ont comme particularité l'emplacement des roues différentielles (latérales, situées au milieu, etc.) motrices et directrices. Des plus, deux ou quatre roues libres peuvent

être placées de telle sorte à équilibrer la base mobile comme pour le manipulateur mobile *H2bis* du LAAS [06], etc.

2.6. Taille du manipulateur mobile

La taille des robots représente un critère important. Dans des environnements fortement encombrés, les petits robots peuvent être d'un grand secours par exemple le robot M3 (Manipulateur Mobile Miniature). Par contre, s'il y a nécessité de transport d'objets lourds, les grands robots sont très intéressants.

Les robots de grande taille comportent des bases mobiles omnidirectionnelles car, ajouter une base mobile non-holonome à un robot gigantesque rend la tâche plus complexe à réaliser.

Il peut exister des robots gigantesques pour des opérations qui doivent s'effectuer en hauteur, et pour lesquels un poids assez important pourrait être soulevé.

2.7. Mode de Manipulation

Dans les définitions classiques des manipulateurs mobiles, la locomotion est assurée par un véhicule et la manipulation se fait grâce à un ou plusieurs manipulateurs. Cependant, il existe un cas particulier où ce sont les roues qui effectuent la tâche de manipulation cas du mobipulator [01].

3. Domaines d'utilisation des manipulateurs mobiles

Les manipulateurs mobiles prennent aujourd'hui des formes diverses tant du point de vue de leur mode de locomotion que de leur taille ou encore de la fonction attribuée aux manipulateurs.

On peut classer les différents manipulateurs mobiles selon les domaines d'applications comme suit :

3.1. Domaine spatial

3.1.1. Mars Exploration Rover (MER)

C'est une mission de la NASA composée de deux robots mobiles ayant pour objectif d'étudier la géologie de la planète Mars et en particulier, le rôle joué par l'eau dans l'histoire de la planète. Ces deux robots, MER-A renommé *Spirit* et MER-B renommé *Opportunity* (Figure 2) ont été lancés au début de l'été 2003 et se sont posés en janvier 2004 sur deux sites martiens susceptibles d'avoir conservé des traces de l'action de l'eau dans leur sol. Chaque rover ou astromobile, piloté par un opérateur depuis la Terre, a alors entamé une tournée en utilisant une batterie d'instruments embarqués pour analyser les roches les plus intéressantes.

Chaque rover se déplace sur 6 roues mues par l'énergie électrique fournie par des panneaux solaires. Il est équipé de 3 paires de caméras utilisées pour la navigation et de plusieurs instruments scientifiques.

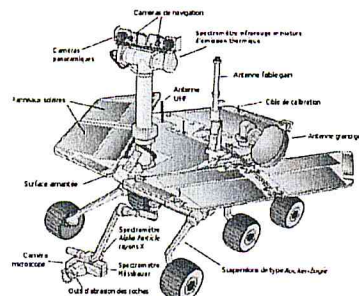


Figure 1: Manipulateur mobile *Spirit* avec son Schéma

3.1.2. Sky Worker

Initié par la NASA, consiste à créer une équipe de manipulateurs mobiles capables de se déplacer sur des panneaux solaires et d'y réaliser des tâches d'assemblage, d'inspection et de maintenance dépassant les capacités des astronautes (Figure 3) [07].

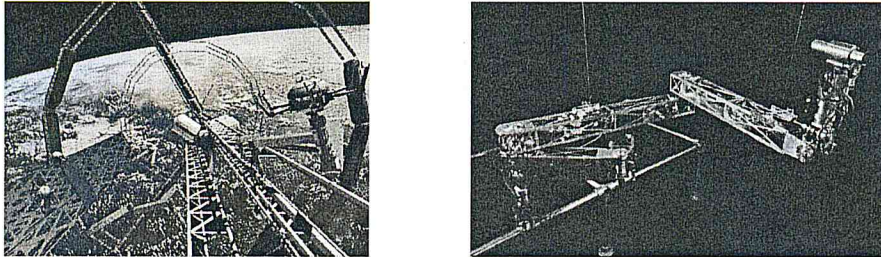


Figure 2: Manipulateurs mobiles du projet *Sky Worker*

3.2. Domaine manufacturier

Les manipulateurs mobiles sont intensivement utilisés dans l'industrie où ils effectuent, sans relâche, des tâches répétitives et avec rigueur. Dans les chaînes de montage de l'industrie automobile, ils y remplacent les ouvriers dans les tâches pénibles et dangereuses (peinture, soudage, emboutissage, etc.). Les robots industriels sont souvent munis de systèmes de vision qui leur procurent une souplesse d'exécution et des moyens de vérifier la qualité des produits fabriqués.

Le robot *Kamro* (Figure 4), est un manipulateur mobile destiné à des tâches autonomes en environnement industriel. Il est constitué d'une base mobile omnidirectionnelle (permettant une grande facilité de mouvements dans des espaces confinés) et de deux manipulateurs PUMA200 de type 6R. *Kamro* est équipé des capteurs d'efforts six axes pour la manipulation, d'un système de vision (deux caméras se trouvant sur chacun des deux effecteurs) pour la reconnaissance d'objets et des capteurs à ultrasons pour la navigation. Le robot peut prendre en charge le transport, la fabrication, la maintenance et le contrôle de processus dans une usine par exemple [08].

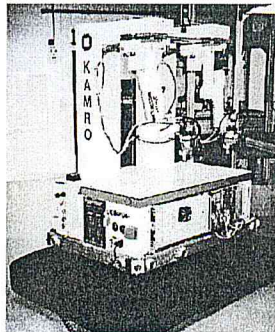


Figure 3: Manipulateur mobile *Kamro*

3.3. Domaine médical

Il existe un grand nombre de robots qui concerne le domaine de la médecine. On distingue les robots en tant qu'outils chirurgicaux, et ceux d'aide au diagnostic.

3.3.1. Domaine médical

On peut citer pour l'exemple du robot *Dermarob* (Figure 5) qui utilise un capteur d'effort permettant de réaliser des tâches qui consistent à faire glisser un instrument sur une surface en y exerçant une pression prédéterminée (prélèvement de peau nécessaire à la greffe de peau, etc.). Ce

geste, difficile à maîtriser pour un médecin, nécessite de faire avancer le *dermatome*¹ à une vitesse et une pression constante [09].



Figure 4: Robot *Dermarob*

3.3.2. Hippocrate

Un des principaux problèmes pour les médecins est de pouvoir localiser l'endroit malade du patient. Le médecin peut désormais voir ce qui constitue l'organisme, mais aussi l'aspect physiologique de ce dernier, et donc de déceler immédiatement si la zone est malade grâce aux robots. Le robot *Hippocrate* [10] (Figure 6) répond à cette demande et permet l'acquisition des images échographiques pour la reconstruction 3D des artères.

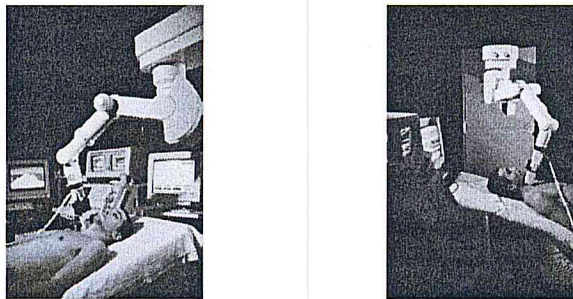


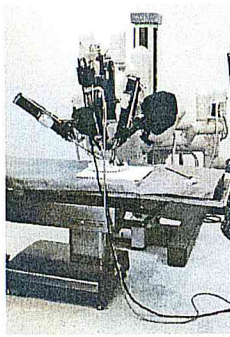
Figure 5: Robot *Hippocrate*

3.3.3. Robot Da-Vinci

Le robot *Da-Vinci* [11] (Figure 7) est un robot médical, et plus précisément une machine dirigée par un chirurgien pour réaliser des opérations. Près de 1400 exemplaires avaient été vendus début 2010, dont plus de 1000 aux États-Unis.

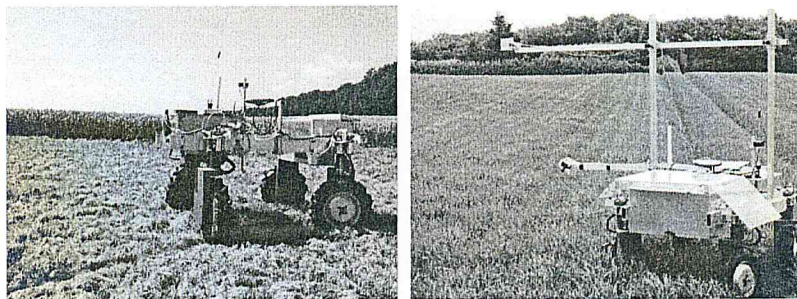
L'utilisation du robot *Da-Vinci* par un chirurgien apporte plusieurs avantages. D'une part, ses bras manipulateurs possèdent 7 ddl, contrairement au poignet humain qui n'en possède que trois, permettent une plus grande précision, car les petits mouvements (tremblements) du chirurgien n'ont pas d'effet sur eux. D'autre part, une opération réalisée à l'aide du *Da-Vinci* est moins invasive que si elle est pratiquée directement par le chirurgien, il y a donc moins de risques de complications postopératoires et les temps d'hospitalisation sont plus courts.

¹ Rasoir qui permet de prélever des fragments cutanés, plus ou moins épais, dans un but d'effectuer des greffes de peau.

Figure 6: Robot *Da-Vinci*

3.4. Domaine agricole

Le robot *AgroBot* [12] (Figure 8) est composé d'un véhicule transportant un bras de 6 ddl, une pince, une tête de 2 ddl et un contrôleur de système. Un système de vision stéréoscopique couleur est utilisé pour piloter le bras. Le robot est capable de naviguer entre les rangées de plants de tomates, effectuer certaines opérations importantes comme la pulvérisation des fleurs et des fruits et la cueillette des tomates mûres. Le système de vision, basé sur un capteur optique, permet de contrôler la direction du mouvement du manipulateur et maintient la base mobile au centre de la voie libre. Les capteurs permettent au robot de faire un choix basé sur des paramètres tels que la couleur, la taille du fruit, le calcul de la distance entre le bras et la cueillette de tomate.

Figure 7: Manipulateur mobile *AgroBot*

3.5. Robotique de service

Les robots de service (supports techniques) permettent aux personnes âgées et handicapées de vivre d'une façon autonome et soutenue dans leurs lieux de résidence privée (maisons, etc.) [13].

3.5.1. Robots domestiques

Les robots domestiques sont des robots au service des particuliers, pour les aider dans leurs tâches ménagères ou de loisir. Il existe par exemple des robots aspirateurs, des robots tondeuse à gazon, etc.

Le premier robot domestique commercialisé fut *Aibo*, le chien artificiel créé par Sony en 1999. Il reconnaît son maître, se relève tout seul fait la joie des enfants. Et avec l'encouragement de l'état japonais « le Project for the Practical Application of Next-Generation Robots organisé par le NEDO » qui a fait de la robotique d'assistance une priorité, de nombreuses sociétés japonaises développent des robots capables de prêter assistance aux humains, à domicile ou au travail, du fait que le pays accuse une pénurie de main d'œuvre en raison du vieillissement rapide de la population. Les robots d'assistance tels *iRobo Q*, *Asimo* et *EMIEW 2* intéressent beaucoup les Japonais, tant à titre privé que professionnel.

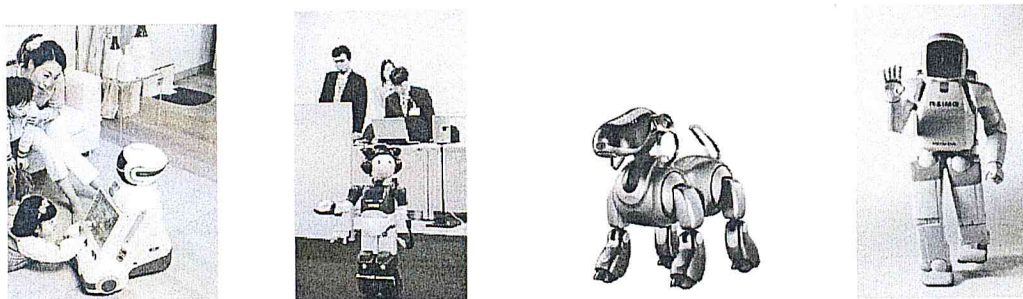


Figure 8: Les robots *iRobo Q*, *EMIEW 2*, *Aibo* & *Asimo*

3.5.2. Robots artistes

On peut voir de nos jours des procédés innovants pour programmer des robots industriels en utilisant la captation de mouvements. A l'aide d'une tablette tactile, l'opérateur effectue son dessin. La gestuelle de sa main est alors captée et renvoyée au *Robographe* [14] (Figure 10) qui restitue fidèlement le dessin dans l'ordre chronologique.

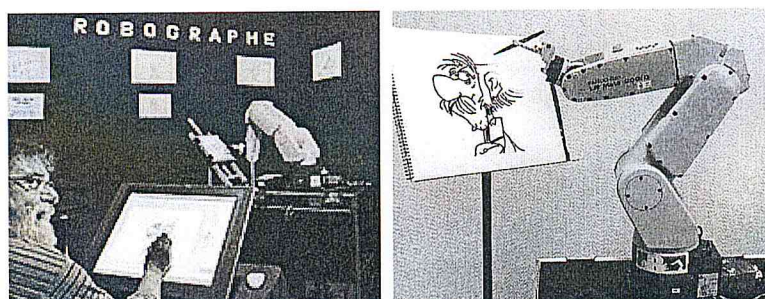


Figure 9: Manipulateur mobile *Robographe*

Le 17 mai 2008 un robot industriel *FANUC ROBOTICS* [15] (Figure 11) a dirigé deux morceaux classiques face à un ensemble instrumental à cordes à paris à la cité des sciences et de l'industrie.

Le procédé utilisé a été la captation de mouvements. Ce procédé est déjà utilisé dans le monde du jeu vidéo et du cinéma. Ainsi, le robot Manipulateur reproduit les mouvements du chef d'orchestre préalablement enregistrés.

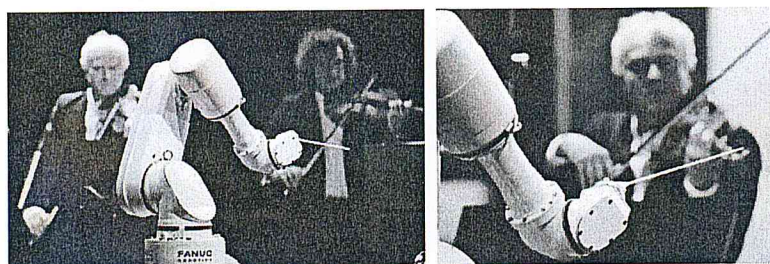


Figure 10: Robot *FANUC ROBOTICS*

3.5.3. Ouverture de porte

Le robot *Lias* (*Leuven Intelligent Autonomous System*) (Figure 12), comporte une base mobile non holonome de type *RobuTER* équipée de nombreux capteurs (un laser rotatif, des odomètres, etc.). La manipulation est accomplie grâce au manipulateur industriel *CRSA465* à six liaisons rotoïdes, comprenant un capteur d'effort, une pince, une caméra assurant la stéréo vision [16]. Ce robot a servi de base expérimentale pour effectuer un mouvement spécifique d'ouverture d'une porte par approche réactive.

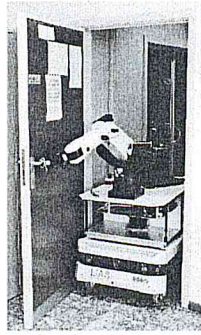


Figure 11: Manipulateur mobile *Lias*

3.5.4. Saisie d'objet

Le robot *EL-E* [17] (Figure 13) apporte les objets désignés à l'aide d'un pointeur laser pour aider les personnes handicapées ou immobilisées. Ses caméras lui permettent d'agir dans un environnement domestique, de repérer des personnes et de saisir un objet se trouvant sur le sol, sur une table ou sur une étagère.



Figure 12: Manipulateur mobile *EL-E*

4. Conclusion

Un manipulateur mobile se compose de deux sous-systèmes mécaniques hétérogènes conçus de manières distinctes (base mobile, manipulateur) et réagissant différemment aux influences extérieures.


Un tour de monde sur les manipulateurs mobiles qui existent de nos jours a été effectué dans ce chapitre. Il cite aussi les différents domaines d'application de ces robots.

Dans le but de contrôler et de superviser un manipulateur mobile quelconque, il faut construire une architecture de contrôle du robot. Une architecture de contrôle est définie comme étant l'ensemble de fonctions élémentaires que le robot peut réaliser. Le contrôle est l'entité qui manipule cette architecture.

Le chapitre suivant présente un état de l'art sur les différentes architectures de télérobotique existantes dans la littérature.

Chapitre 2

Architectures de télérobotique



1. Introduction

Ces dernières années, les technologies de l'information et de la communication ont permis, d'un côté pour les activités des prestations, de ne plus concentrer la production du service dans un lieu unique. D'un autre côté, ces technologies ont permis de dispenser la présence de l'homme dans des milieux dangereux (milieu nucléaire, haute température, etc.) ou tout simplement pour des missions d'exploration (milieu sous-marin, spatial, etc.).

La téléopération permet de réduire, pour les opérateurs humains, les risques associés aux travaux se déroulant dans des environnements hostiles. Les premiers systèmes de téléopération ont donc été développés pour la manipulation de produits radioactifs et l'exploration spatiale et sous-marine. Puis, grâce aux progrès en matière de technologie et à l'intégration de fonctions avancées (notamment la locomotion), le champ d'application de la téléopération s'est étendu à plusieurs domaines, l'intervention en sites contaminés, la surveillance, l'aide aux handicapés et le service.

Un système de téléopération actuel peut intégrer des composants technologiques très variés, des interfaces sophistiquées, des robots avec une certaine capacité de décision, des techniques de visualisation avancées (réalité virtuelle) et divers outils (schémas de contrôle partagé, planification). Ceci, et les conditions héritées des premières applications auxquelles sont destinés, font que la construction d'un système de téléopération est coûteuse en ressources et en temps de développement. Depuis plusieurs années, des efforts ont été déployés pour que la téléopération devienne plus accessible. Plusieurs projets cherchent à diminuer le coût et la complexité du développement de ce type de systèmes [18].

On présente dans ce deuxième chapitre les aspects relatifs à la téléopération, la télérobotique et les différentes architectures de télérobotique des manipulateurs mobiles existants dans la littérature.

2. Téléopération

La téléopération désigne les principes et les techniques qui permettent à l'opérateur humain d'accomplir une tâche à distance, à l'aide d'un système robotique d'intervention (dispositif esclave), commandé à partir d'une station de contrôle (dispositif maître), par l'intermédiaire d'un canal de télécommunication (figure 1) [19].



Figure 1: Illustration de l'architecture générale d'un système de téléopération [19]

Elle a pris ses origines dans le besoin de prolonger le geste de l'homme au delà de la main, et se poursuit par l'ambition de se trouver là où on ne se trouve pas (téléprésence) comme illustré dans la figure 2 [20].

De nos jours, la téléopération s'applique aussi bien aux bras redondants et aux diverses sortes de robots mobiles à roues, et à pattes. Cependant, un certain nombre de problèmes de fond et complexes existent encore. Nous citons par exemple, l'inévitable problème de délai dû au canal de transmission entre les deux sites (maître et esclave) qui implique un retard de transmission d'informations sensorielles utiles à l'opérateur. En effet, tout retard de transmission entraîne des instabilités difficilement compensables [21]. D'autres problèmes qui ne cessent de préoccuper l'esprit de nombreux chercheurs dans ce domaine concernent le choix d'un système de commande universel [22] ainsi que la compréhension de nos propres sous-systèmes sensoriels [23].

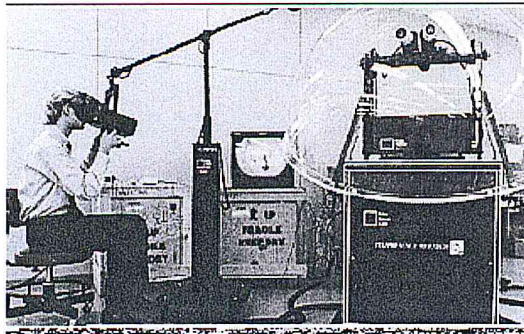


Figure 2: Illustration d'un système de téléprésence avec un robot mobile [20]

Dans les systèmes de téléopération, l'opérateur doit impérativement subir un apprentissage qui peut être long avant d'obtenir une bonne adaptation et une bonne maîtrise du système de téléopération. En plus, l'adaptation obtenue peut ne pas être adéquate pour une autre architecture. La sophistication et le nombre important d'applications téléopérées engendre une augmentation de la charge de travail de l'opérateur, ce qui entraîne une diminution de ses performances et augmente ainsi les risques d'erreurs de téléopération.

L'évolution de l'informatique et des supports de télécommunication numérique est à l'origine de l'évolution de la téléopération. Grâce à ces supports, les robots esclaves ont pu être éloignés à des distances considérables [24]. En effet, initialement l'opérateur commandait le robot esclave dans un niveau assez bas, les trajectoires issues des actions désirées sont envoyées directement via le dispositif maître. Le retour d'information (Images, force de contact, etc.) se fait directement vers l'opérateur. Mais malheureusement, cette approche n'a pas duré très longtemps, car les chercheurs se sont vite rendu compte de la nécessité d'exploiter les possibilités que peuvent offrir des ordinateurs pour apporter une assistance à l'opérateur. Depuis, plusieurs approches et concepts ont été proposés pour améliorer et faire évoluer les systèmes de téléopération. Nous pouvons résumer le classement des différentes approches (détaillées dans [25] et [26]) représentant les modes d'interaction homme/machine que l'on retrouve généralement dans la plupart des systèmes de téléopération :

2.1. Téléopération Assistée par Ordinateur (TAO)

Elle est vue comme une voie intermédiaire entre la téléopération bas niveau et la supervision. L'objectif de la TAO est de réaliser, à chaque instant et pour toute étape de la tâche, une exploitation optimale des ressources d'un ordinateur. La TAO bénéficie actuellement de l'utilisation de la réalité virtuelle et augmentée.

2.2. Téléopération à Désignation d'Objectif (supervisée) (TDO)

Dans ce mode, l'opérateur est plutôt vu comme un superviseur. L'intervention de l'opérateur se limite, dans ce cas, à la désignation d'objectifs qui seront réalisés par le robot.

2.3. Téléopération Semi Autonome (TSA)

La plupart des systèmes de téléopération actuels, s'orientent vers l'utilisation des deux concepts précédents (TAO et TDO). En effet, ils tentent à moderniser la téléopération par une meilleure exploitation simultanée de l'autonomie du robot et des capacités de l'opérateur.

Nous pouvons résumer dans la figure 3, le classement de ces trois concepts selon le degré d'autonomie du dispositif esclave.

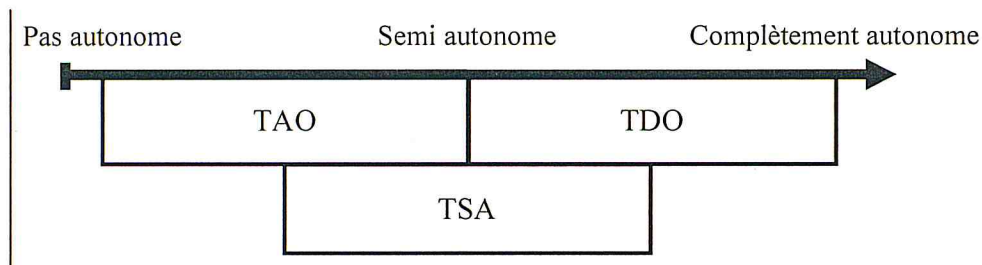


Figure 3: Classement de la TAO, TDO et TSA selon le degré d'autonomie du dispositif esclave

3. Télérobotique

La télérobotique est une forme de téléopération lorsque l'opérateur réalise des tâches à distance en utilisant un robot. Ce dernier peut fonctionner d'une façon autonome. Elle s'applique dans le cadre de la TDO [27].

La télérobotique résulte, en fait, de la fusion des deux domaines séparés qui sont la téléopération et la robotique. En effet, la robotique autonome n'étant pas encore tout à fait au point, le robot doit présentement être opéré à distance par un opérateur humain. On doit donc tenir compte des principes développés en téléopération. Cependant, comme le robot peut exécuter des tâches élémentaires de façon autonome, on parle de télérobotique plutôt que de téléopération. La télérobotique trouve des applications partout où l'homme a des difficultés à travailler directement (milieu hostile, lointain, trop grand ou trop petit) et où les tâches sont suffisamment complexes ou imprévisibles pour rendre difficile une automatisation complète. La condition principale de développement de la télérobotique est sa capacité à concurrencer l'intervention directe d'un homme ou l'utilisation d'un système automatique très spécialisé. Dans le premier cas, l'atout de la télérobotique est tout d'abord le remplacement d'un travail humain pénible ou dangereux par un autre, plus sûr et confortable. Dans le second cas il faut montrer l'intérêt d'un matériel plus versatile que le système automatique dédié à l'application envisagée.

4. Télérobotique basée sur Internet

Les systèmes de télérobotique sont traditionnellement implémentés en utilisant des canaux de communication dédiés. Le déplacement du site esclave et/ou du site maître nécessite le déplacement du matériel ainsi que la reconfiguration réseau du canal de communication reliant ces deux sites. Les personnes sont obligées de se déplacer sur le site client (centre de contrôle afin de travailler et de préparer les missions). De plus, si la machine cliente tombe en panne il devient difficile ou très coûteux de poursuivre la mission puisqu'il s'agit d'une machine dédiée où se trouve l'interface homme/machine.

Cependant la télérobotique basée sur Internet, permet une délocalisation facile des opérateurs à faible coût, et ne dépend pas de l'endroit où se trouve le matériel qui permet de contrôler le robot qui, lui, se trouve sur le site esclave généralement distant. Cette idée a été exploitée par la NASA lors de la préparation de la mission *sojourner* sur Mars. Initialement les scientifiques étaient obligés de se déplacer jusqu'au centre de contrôle en Californie afin de travailler et de contrôler le robot. Le développement d'une interface Internet [28], a donné la possibilité aux scientifiques de travailler et de collaborer ensemble pour contrôler le robot de n'importe où dans le monde.

5. Architectures pour la télérobotique

Les premiers sites Internet de la télérobotique sur le Web ont été créés en 1994 par l'américain Goldberg [29] et l'Australien Taylor [30]. Depuis, de nombreux travaux ont eu lieu autour du contrôle de systèmes mécaniques (robots, caméras, machines de productique, etc.) connectés sur l'internet. Nous pouvons citer, sans vouloir être exhaustif, *Telegarden* [31], *PumaPaint* [32], *ARITI* [33], *KhepOnTheWeb* [34], les travaux du *LISYC* [35], du *LIRMM* [36], de *l'INSA* [37], etc.

Depuis que la télérobotique via Internet a vue le jour, un recensement des systèmes fonctionnels a été effectué afin de centraliser l'ensemble des systèmes de télé-contrôle via Internet. En effet, du site Internet de la NASA a recensé 26 projets dont un Français (celui du système *ARITI*) jusqu'à 1999, le projet australien *Telelabs project* a recensé une soixantaine de sites Internet de télé-contrôle dans le monde en mai 2005.

Ces systèmes robotiques opérant à travers le *World Wide Web* requièrent une infrastructure très simple pour leur déploiement et sont accessibles dans le monde entier. De plus, puisque l'interface est facile à comprendre et à maîtriser, l'utilisateur ne nécessite pas d'entraînement. Mais surtout, la téléopération sur internet ouvre de très importantes possibilités de collaboration et de partage de ressources entre les différentes équipes de recherche éparpillées dans le monde.

5.1. Architecture de référence pour la télérobotique

Alvarez et al. [38] ont proposé une architecture de référence (figure 4) pour des systèmes de téléopération. Cette architecture est constituée de plusieurs composants :

- une représentation graphique pour l'affichage du robot et de l'environnement de travail
- un composant de détection de collisions
- l'interface utilisateur pour l'interaction avec l'opérateur
- un composant de communications et le contrôleur, responsable de la gestion et de l'exécution des consignes

Cette architecture de référence a été utilisée dans divers projets, parmi lesquels, un système de contrôle pour un véhicule d'inspection sous-marin et un système robotique pour la maintenance de centrales nucléaires. Actuellement, l'architecture est adaptée pour le développement d'une unité de contrôle pour un robot autonome. Les différentes implantations de l'architecture ont été réalisées avec des bibliothèques commerciales comme *Robcad* et *Paradise*.

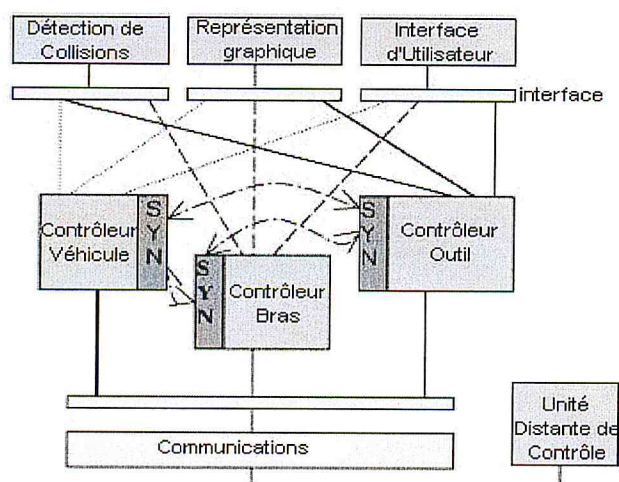


Figure 4: Architecture de référence pour la téléopération [38]

Dans ce qui suit, nous présentons quelques systèmes de téléopération via le réseau Internet qui sont encore fonctionnels. Nous proposons, aussi, de les classer selon les deux catégories de systèmes, à savoir, les systèmes de téléopération non collaboratifs et les systèmes de téléopération multiutilisateur qui proposent une collaboration (via un système de vote) des utilisateurs pour réaliser une tâche, soit par le robot, soit par un autre utilisateur distant.

5.2. Projet Cambridge Coffee Pot

C'est le premier dispositif « distribué » sur internet. Ce système utilisait une webcam pour transmettre des images d'une cafetière placée dans le laboratoire d'informatique de l'Université de Cambridge [39].

5.3. Projet Mercury

Le projet Mercury [40] a été mis en ligne en août 1994. Il permettait aux utilisateurs d'attraper et de manipuler différents objets en utilisant un bras robotique. En septembre 1994, un autre télémanipulateur a été mis en ligne cette fois dans l'Université de Western Australia [41]. Dans la première version de ce système (figure 5), les utilisateurs devaient saisir des coordonnées spatiales pour spécifier les mouvements du manipulateur. Plusieurs autres interfaces ont été utilisées ensuite pour simplifier le contrôle du robot [42].

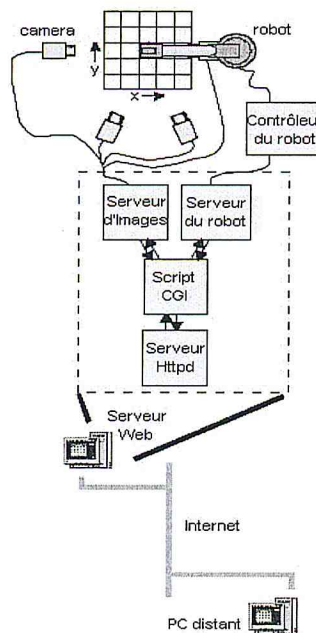


Figure 5: Architecture du contrôle d'un manipulateur via le World Wide Web [41]

5.4. Projet Web Driver

L'idée de rendre accessible un dispositif en utilisant une interface web a été suivie par de nombreux projets, parmi lesquels le télescope robotique de Cox [43], le télérobot mobile « FortyTwo » à Manchester [44], le système DIGIMUSE, conçu pour la visualisation interactive des objets d'art et l'interface web pour la mission Pathfinder de la NASA [45] qui permettait aux chercheurs de collaborer entre eux et travailler dans la mission sans se déplacer au centre de contrôle en Californie. D'autres projets se sont intéressés au contrôle distant de robots mobiles comme les systèmes KhepOnTheWeb [46], le robot Xavier de l'Université Carnegie Mellon [47], le projet Museum Tour-guide [48] et le projet WebPioneer [49]. Ces systèmes permettent à l'utilisateur de contrôler un robot soit dans un environnement statique, soit dans l'exploration d'environnements dynamiques.

La plupart des systèmes de contrôle sur internet ont été construits en utilisant des pages HTML statiques et des programmes CGI (Common Gateway Interface : c'est une interface utilisée par les serveurs HTTP). La page HTML permet à l'utilisateur de connaître l'état du système et de générer les ordres à exécuter. Les ordres sont traités et exécutés par le programme CGI qui contrôle le robot, sans supervision ni intervention de l'utilisateur. Enfin, lorsque l'ordre est achevé ou qu'une erreur s'est produite, une nouvelle page HTML est générée. L'interaction avec l'utilisateur est donc limitée car il

est impossible de récupérer les consignes de l'utilisateur et de lui présenter l'information de retour en même temps. Pour résoudre ce problème, l'utilisation de plusieurs programmes CGI associés aux différents cadres d'une page HTML a été proposée. Cependant, puisqu'un programme CGI établit une nouvelle connexion à chaque fois qu'il est invoqué, leur multiplication augmentait le temps de réponse du système. L'alternative adoptée pour la gestion de l'interface a été l'utilisation d'applets Java à la place de HTML/CGI. Avec une applet Java, la connexion entre l'interface Web et le contrôleur du robot est établie une seule fois et la transmission de données peut donc se réaliser à tout moment dans les deux directions.

Le système *WebDriver* [50] est un exemple d'application de cette technologie. Étant le successeur du système *WebPioneer*, *WebDriver* est conçu pour la téléopération de robots mobiles évoluant dans des environnements dynamiques inconnus. L'interface utilisateur de *WebDriver* (figure 6) est une applet Java qui reçoit les commandes de l'utilisateur et affiche l'information des capteurs du robot.

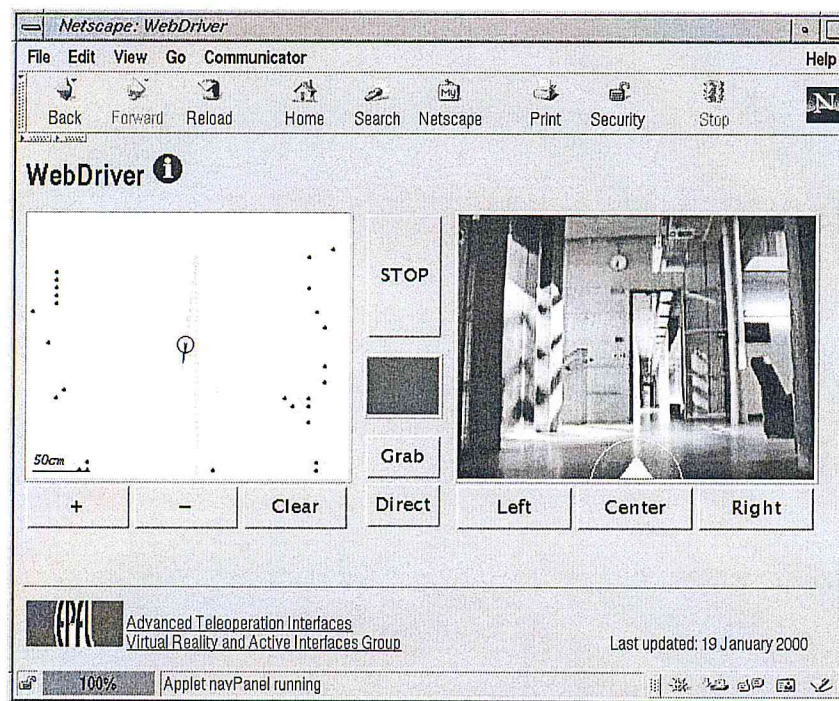


Figure 6: Interface Utilisateur de *WebDriver* [50]

WebDriver met à jour les images de l'interface utilisateur lorsque certains événements (comme la détection d'un obstacle) se produisent. Ce schéma permet de réduire le trafic du réseau généré par un serveur vidéo très utilisé car facile à mettre en œuvre. Cependant, en limitant l'information du site de travail aux seules images vidéo, l'utilisateur a une perception limitée de l'environnement. Pour améliorer l'information de retour et gérer les délais de transmission, il est possible de réaliser une simulation graphique (2D ou 3D) de l'action souhaitée. La simulation est en suite superposée aux images réelles pour aider l'utilisateur à réaliser sa tâche [51].

5.5. Projet CTS

L'USARC (Universities Space Automation / Robotics Consortium) a développé TCS (Telerobotics Construction Set), un outil de construction de systèmes distribués. Ce système utilise un modèle de distribution des objets pour permettre la mise en œuvre de systèmes Télérobotiques [52]. Les interfaces fournies par le TCS pour le contrôle du robot et la gestion de la vidéo et des données de l'application, permettent l'intégration de ressources hétérogènes dans le système. Un mécanisme appelé TSM

(Telerobotics Session Manager) permet à l'opérateur de reconfigurer dynamiquement les flux de données et les interconnexions du système développé.

5.6. Projet Telerobotic Test Bed

Les architectures de contrôle les plus significatives ont été proposées par le Jet Propulsion Laboratory. Un des résultats de leurs recherches a été le Telerobotic Test bed, conçu pour le développement des systèmes autonomes, multi-robot, pour la révision de satellites [53]. Le Telerobotic Test bed était un système très complexe dont la structure logicielle n'a abouti à la fin du projet. Plus tard, plusieurs projets de recherche sur des manipulateurs en orbite ont été mis en place : le projet Remote Surface Inspection écrit en C sur VxWorks, le projet Satellite Servicing développé en C et assembleur, et le projet MOTES écrit en Ada sur VxWorks [54]. Ces projets ont été implantés séparément, sans partage de code, même s'ils fournissaient une fonctionnalité similaire. Le même scénario s'est déroulé avec les projets JPL contrôlant des robots mobiles. Les différences d'architecture et d'outils de développement ont empêché l'utilisation de ces nombreux projets dans d'autres missions. Par exemple, le véhicule *sojourner* participant à la mission Pathfinder a été programmé avec un système entièrement nouveau. Après le *sojourner*, la recherche d'une infrastructure logicielle modulaire, reconfigurable et réutilisable a commencé au JPL. Plusieurs systèmes avec cet objectif ont été produits : Control Shell, FIDO, DARPA TMR, Nano rover [55]. Pour recentrer les efforts et profiter des expériences acquises sur les schémas de contrôle et l'autonomie dans les projets RAX (Remote Agent Experiment), réalisé en collaboration avec le Centre de Recherche Ames de la NASA [56], et le projet MDS (Mission Data System), le projet CLARAty a été créé. CLARAty (Coupled Layer Architecture for Robotic Autonomy) est une architecture conçue pour le contrôle de robots autonomes [57]. La structure de CLARAty contient une couche de Décision et une couche Fonctionnelle. La couche Fonctionnelle est une interface à toute l'architecture matérielle du système et de ses services. La couche de Décision, elle, utilise les capacités de la couche Fonctionnelle pour réaliser les activités lui permettant d'atteindre ses objectifs. Une première implantation de cette architecture est en train d'être réalisée afin de l'intégrer dans les plates formes de recherche des véhicules d'exploration de Mars, Rocky 7 et 8. En outre, une collaboration avec la NASA a été instaurée dans le but d'avoir une implantation complète de l'architecture dans le courant 2007.

5.7. Projet VEVI

Le système VEVI (Virtual Environment Vehicle Interface) (figure 7), développé par le Centre de Recherche Ames de la NASA, est une interface utilisateur modulaire pour la téléopération directe et le contrôle supervisé de véhicules robotiques [58]. VEVI présente une scène 3D constituée par les modèles du véhicule contrôlé et de l'environnement dans lequel il opère. L'interface permet de connaître l'état du système, de planifier et d'examiner des actions, et d'envoyer les consignes devant être exécutées par le véhicule. VEVI a été conçu et implanté pour être modulaire, distribué et facile à opérer. VEVI peut être configuré en ajoutant des modules d'entrée/sortie qui communiquent avec le mécanisme robotique à contrôler. La communication est réalisée suivant un protocole de communication spécifique. Le mécanisme robotique est constitué d'un ensemble d'objets affichés dans l'environnement 3D de VEVI. L'information géométrique de ces objets et leurs relations cinématiques sont décrites dans les fichiers de configuration de VEVI. Le système VEVI a été utilisé dans plusieurs projets de recherche, parmi lesquels le robot sous-marin TROV (Telepresence Remotely Operated Vehicle) travaillant dans l'Antarctique [59] et le robot à 8 pattes DantelII employé pour la recherche scientifique dans le cratère du volcan *Mount Spurr* en Alaska [60].

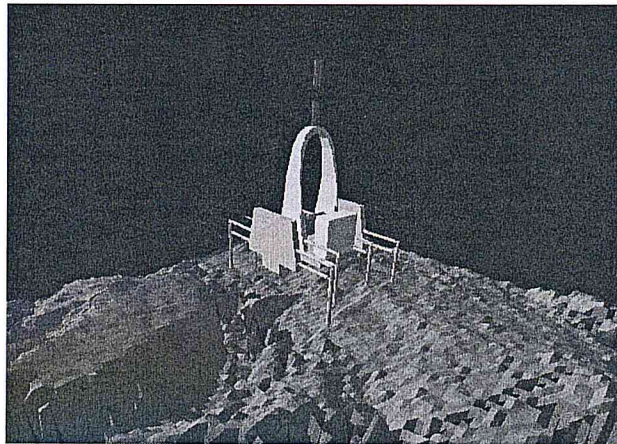


Figure 7: Le robot DanteII dans le système VEVI [58]

5.8. Projet VIZ

L'expérience acquise dans le développement du système VEVI a été utilisée dans la mise en œuvre du projet Viz. Viz est une interface de réalité virtuelle, modulaire, flexible et facilement configurable [61]. Viz a été conçu pour l'intégration simple des systèmes commerciaux existants et des technologies développées par la NASA. Cette stratégie cherche principalement à réduire le cycle de développement et permettre ainsi la construction rapide de prototypes.

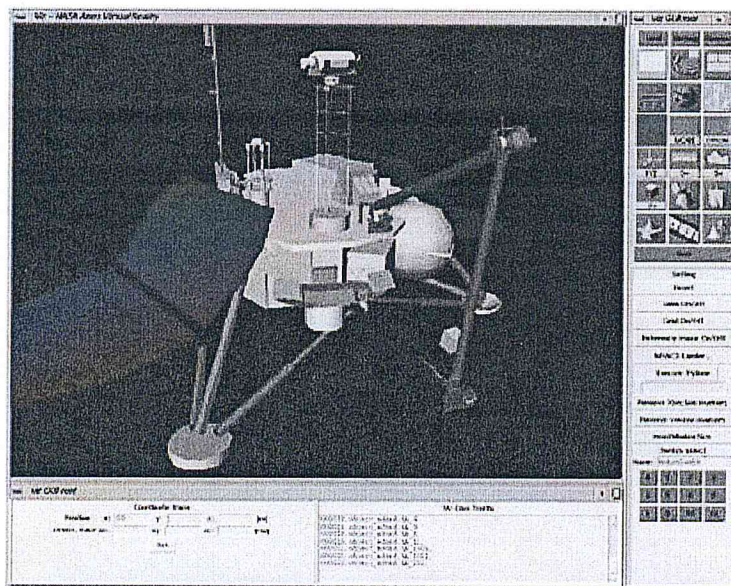


Figure 8: Interface Viz pour la mission Mars Polar Lander [61]

Le module 3D de Viz est un serveur générique auquel les clients se connectent via le protocole TCP/IP. Une fois que le client est connecté au serveur, il peut modifier la scène dynamiquement en envoyant des messages. Le serveur interprète les messages des clients et réalise le rendu de l'environnement virtuel. En outre, une API pour l'envoi de messages est générée automatiquement en Java, C++ et Python à partir de la spécification des messages. En utilisant cette API, il est possible de transformer un module externe en un client de Viz. Viz peut être exécuté sur les plates-formes SGI et Linux, et la visualisation 3D du système est basée sur la bibliothèque graphique Open Inventor.

Le système Viz a été évalué dans la préparation de la mission Mars Polar Lander (Figure 8) pour intégrer dans une même interface les différents outils de planification, simulation et traitement de données développés par la NASA et le JPL.

5.9. Projet VEMI

Les projets VEVI et Viz ont réalisé avec succès l'intégration d'interfaces de réalité virtuelle dans le contrôle de mécanismes robotiques. Cependant le contexte d'application de ces systèmes (exploration spatiale) fait que les besoins de performance, de sophistication et donc de coût des composants impliqués, sont très élevés. Pour cette raison, plusieurs projets proposant des plateformes de test pour la téléopération à faible coût, ont été développés. Par exemple, Kaber et al [62] à l'Université de l'Etat du Mississippi, a développé une plate-forme de test pour la téléopération sur Windows NT/PC. Ce système, appelé VEMI (Virtual Environment Manipulator Interface) utilise un environnement virtuel pour contrôler un robot PUMA 560. Un modèle mathématique a été intégré dans VEMI pour que les mouvements du modèle virtuel correspondent à ceux du manipulateur réel. VEMI communique avec le contrôleur du robot en utilisant le protocole TCP/IP, ce qui permet de contrôler le robot à travers d'un réseau local ou d'Internet. Des plates-formes de test pour des robots mobiles ont aussi été implantées, comme le système KITE (Khepera Integrated Testing Environment) développé pour l'évaluation d'algorithmes de localisation et navigation [63], et l'environnement de test pour robots autonomes réalisé par Finke [64].

5.10. Projet CINEGEN

D'autres projets, comme le projet CINEGEN [65] travaillent sur le prototypage rapide de robots manipulateurs. CINEGEN propose un outil, simple d'utilisation et rapide à mettre en œuvre, pour la simulation cinématique de robots manipulateurs à structure quelconque dans un environnement de type réalité virtuelle. Ce projet cherche à faciliter, pour les non-spécialistes, les aspects de simulation ou de programmation hors-ligne de robots. La mise en œuvre rapide d'une nouvelle simulation s'effectue grâce à la description dans un fichier texte, des paramètres géométriques élémentaires de la structure du robot et les différentes contraintes que la structure comporte. Le fichier de description est analysé par le programme qui construit de manière automatique un modèle numérique de la cinématique du robot en vue de satisfaire toutes les contraintes. En fonction des consignes données en mode interactif par l'utilisateur, le "moteur" cinématique calcule les mouvements que le robot doit effectuer tout en respectant les contraintes externes. L'utilisation d'un environnement virtuel permet l'interaction entre l'utilisateur et le modèle contrôlé pour la définition de tâches et l'étude intuitive du comportement du robot.

6. Conclusion

Dans ce chapitre, nous avons présenté l'état de recherches dans le domaine de la télérobotique. Nous avons vu comment l'utilisation d'Internet et des interfaces Web a réussi à rendre accessibles de nombreux systèmes robotiques, permettant, ainsi, la collaboration entre les équipes de recherche et l'ouverture de la robotique à un large public. Les interfaces de réalité virtuelle pour la téléopération ont aussi été présentées.

La recherche sur les architectures pour la télérobotique a aussi été étudiée. La leçon à retenir de cette étude est le manque d'un environnement de construction d'applications prenant en compte tous les aspects impliqués dans le développement d'un système de télérobotique et non seulement au niveau du contrôle de dispositifs effecteurs.

Le chapitre suivant présente l'architecture multi-agents de télérobotique du manipulateur mobile *RobuTER/ULM*. Les détails sur les agents et les systèmes multi-agents sont donnée en annexe A.

Chapitre 3

Extension de l'architecture de contrôle à la télérobotique des manipulateurs mobiles via Internet

1. Introduction

L'objectif visé dans ce chapitre consiste à étendre l'architecture multi-agents proposée dans [66] pour le contrôle des manipulateurs mobiles, pour la télérobotique de ce type de robots via un réseau local ou internet. Cette architecture est indépendante de l'architecture matérielle et logicielle du robot ce qui va permettre l'ajout de nouvelles ressources ou équipements aux robots.

L'architecture de contrôle d'un manipulateur mobile doit être perçue comme une entité unique et globale même s'il présente un ensemble de sous-systèmes autonomes (sous système de contrôle de la base mobile, sous système de contrôle du bras manipulateur, sous système de contrôle des capteurs, etc.). Cette globalisation introduit des besoins de communication, d'interaction, de partage de connaissances et de partage de ressources.

2. Modèle globale de l'architecture de télérobotique

La figure 1 représente une vue globale de l'architecture de télérobotique d'un manipulateur mobile via un réseau local ou internet.

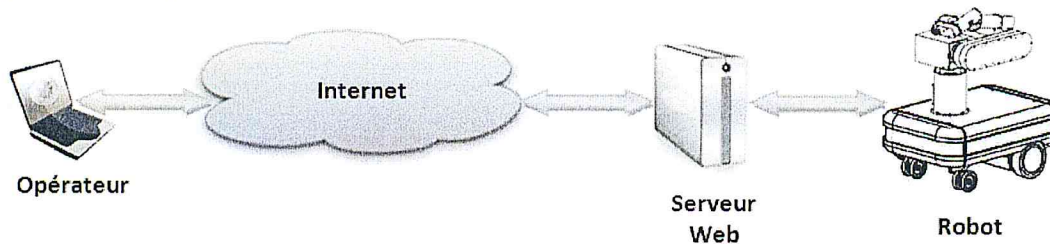


Figure 1: Vue globale d'architecture de télérobotique

L'architecture de contrôle proposée dans [66] pour le contrôle des manipulateurs mobiles est constituée d'un ensemble d'agents hybrides qui pilotent l'ensemble de ressources hétérogènes du robot (base mobile, bras manipulateurs et système de vision) : *Agent Superviseur*, *Agent Robot Manipulateur Local*, *Agent Robot Mobile Local*, *Agent de Vision*, *Agent Robot Manipulateur Distant*, *Agent Robot Mobile Distant*.

L'*Agent Robot Mobile Local* et l'*Agent Robot Manipulateur Local* contrôlent respectivement, la base mobile et le bras manipulateur du robot à partir d'un site local (off-board). L'*Agent Superviseur* a pour rôle la gestion des missions et la répartition des plans d'opérations sur les différents agents ainsi que la communication et la négociation avec ces derniers.

Les agents *Superviseur*, *Robot Mobile Local* et *Robot Manipulateur Local*, possèdent un niveau substantiel d'autonomie à travers des capacités de perception de l'environnement qui est réalisée à travers la communication avec les agents distants et l'échange d'informations entre les agents. Ils possèdent aussi des connaissances suffisantes pour une prise de décision autonome.

Deux autres agents ont été introduits afin d'assurer la communication entre les agents locaux et les ressources matérielles, ainsi que d'autres fonctionnalités. Ces agents sont l'*Agent Robot Mobile Distant* pour le contrôle de la base mobile, et l'*Agent Robot Manipulateur Distant* pour le contrôle du bras manipulateur.

Les caractéristiques que doit posséder cette architecture multi-agents sont [66] :

- *Modularité* : La modularité de l'architecture des agents doit s'articuler autour de la décomposition en modules qui peuvent être développés et mis en œuvre. Cette caractéristique permet à tout système de contrôle, d'évoluer en lui rajoutant de nouvelles fonctionnalités.
- *Réactivité à l'environnement* : Les agents doivent être capables de gérer en temps réel des événements extérieurs asynchrones afin de respecter la dynamique de l'environnement et celle de l'architecture. Un événement extérieur peut avoir plusieurs origines : présence d'un obstacle imprévu, panne inopinée, requête en provenance d'un autre agent, etc. La réactivité implique généralement un traitement en temps réel de ces événements. Ainsi, le système de contrôle doit intégrer la notion de priorité et d'urgence de traitement des événements.
- *Comportements intelligents* : L'intelligence des agents se traduit par des capacités de perception, de communication, de raisonnement et d'action.

L'extension de cette architecture à la télérobotique via Internet consiste à développer une interface homme/robot conviviale et d'un serveur web à installer au niveau du PC off-bord (voir Figure 2).

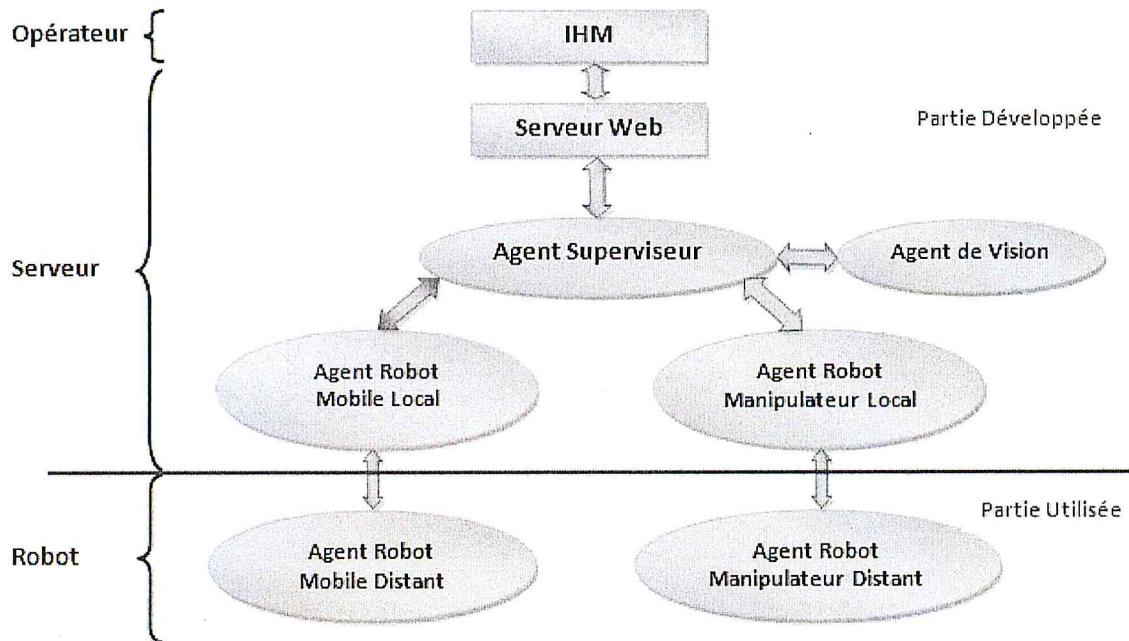


Figure 2: Extension de l'architecture de contrôle proposée dans [66] pour la télérobotique des manipulateurs mobiles via Internet

3. Modèle détaillé de l'architecture de contrôle proposée dans [66]

Un modèle détaillé de l'architecture de contrôle des manipulateurs mobiles est proposé dans ce qui suit :

3.1. Agent Superviseur

L'Agent Superviseur, donné par figure 3, a pour rôle de recevoir une mission de l'opérateur et de vérifier si elle a été exécuté auparavant ou pas.

Dans le premier cas il charge directement le plan correspondant à l'exécution de cette mission. Dans le deuxième, il transmet la mission telle qu'elle est aux autres agents du système. Ensuite, les agents locaux (Robot Mobile Local, Robot Manipulateur Local, Vision) établissent un plan coordonné pour l'exécution de la mission chacun selon ses compétences. A la fin, ils informent le Superviseur des faits déroulés et le transmettent les plans d'opérations générés.

Une base de connaissances est dédiée à l'agent dans laquelle sont sauvegardés les paramètres de configuration du système et du robot, un ensemble ordonnancé d'opérations correspondantes à une mission et tout l'historique du fonctionnement du système.

Les capacités comportementales et décisionnelles de cet agent sont réparties sur les modules suivants :

- **Module Communication** : C'est un ensemble de tâches concurrentes dont la fonction globale est, d'une part, la capture des messages et des événements provenant des autres agents ou des méta-agents de l'Agent Superviseur, leurs interprétation et leurs traitements, et d'autre part, la mise en forme, le codage et la transmission des messages à envoyer.

- **Module Gestion de l'interface** : Ce module assure l'interprétation des données introduites par l'opérateur via l'interface graphique et la diffusion des messages provenant des autres agents sur cette même interface.

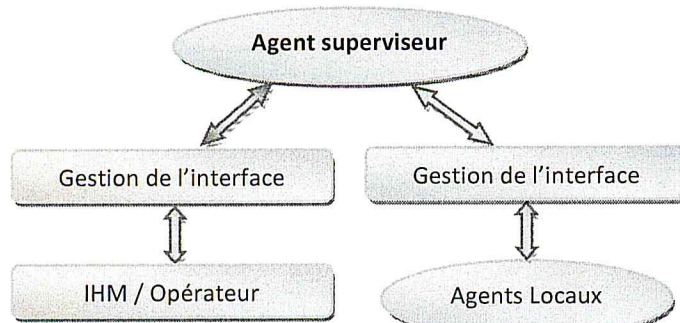


Figure 3: Agent Superviseur

3.2. Agent Robot Mobile Local

Cet agent assure la gestion locale de la base mobile du robot en intégrant les fonctions de contrôle temps réel. L'architecture proposée pour cet agent (figure 4) confère à la base mobile des capacités délibératives pour raisonner sur des situations complexes :

- L'agent doit adopter, suivant sa situation, plusieurs comportements. Il doit mettre en œuvre des mécanismes de raisonnement intelligents sur ses opérations et sur son plan d'opérations.
- L'agent doit contrôler de manière autonome l'ensemble de ses actions en fonction des perceptions de son environnement (perturbations dans le plan d'opérations, présence d'obstacles, etc.), et s'y adapter en élaborant un autre plan.

L'Agent Robot Mobile Local dispose, en outre de la base de connaissances et du module Communication, des compétences suivantes :

- **Module de Configuration** : En plus de permettre à cet agent d'avoir toutes les informations concernant ses Accointances et son Environnement, ce module permet de rajouter des équipements pour pouvoir les contrôler à partir de cet agent ainsi que les Opérations pouvant être exécutées pas ces équipements.
- **Module de Communication** : Ce module est composé des tâches identiques à celles du module de Communication de l'Agent Superviseur. La seule différence est que le Module de communication de cet agent permet la communication avec l'Agent Superviseur ainsi que la communication avec les agents de niveau inférieur (Les Agents Distants).
- **Module de Positionnement** : Dans l'exécution de certaines missions, il est parfois nécessaire de connaître la position de la base mobile par rapport à un repère donné (Repère de la base mobile ou le repère absolu). Ce module a les facultés de calculer la position sur un plan par rapport à n'importe quel repère.
- **Module Surveillance** : Ce module permet de réaliser un Suivi des Opérations avant (pré-contrôle), pendant et après (post-contrôle) l'exécution de chaque opération élémentaire. Une autre tâche compose ce module, il s'agit de la Détection d'Anomalies. Cette tâche se charge de reconnaître les situations qui correspondent à un dysfonctionnement de l'agent. Dans le cas où une anomalie est détectée, la tâche Diagnostique s'active afin de charger ces événements dans la base de connaissances de l'agent. A la fin une tâche de Reprise est lancée pour choisir la meilleure solution pour traiter le problème dans le cas de l'existence d'une solution à ce dernier.

- **Module Planification** : Ce module permet la négociation avec les autres agents du système dans le but d'élaborer un plan d'opérations coordonnée pour l'exécution de la mission introduite par l'opérateur.

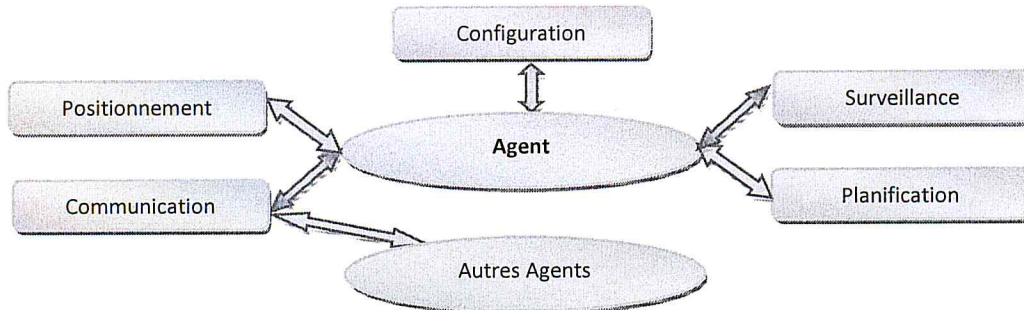


Figure 4: Agent Robot Mobile/Manipulateur Local

3.3. Agent Robot Manipulateur Local

Cet agent (figure 4) dispose des mêmes modules que l'agent précédent. Les modules de planification, surveillance et configuration sont identiques à ceux de l'Agent Robot Mobile Local.

- **Module de Communication** : Une seule fonction différencie ce module et les autres modules de communication, à savoir : la fonction d'interprétation des messages.
- **Module de Positionnement** : Afin de pouvoir contrôler en position le bras manipulateur, un modèle mathématique devrait être élaboré. Ce module contient l'implémentation des modèles du bras manipulateur.

3.4. Agent Robot Mobile Distant

Cet agent (figure 5) a une architecture globale légèrement différente de celles vues précédemment. C'est une sorte d'interface entre l'Agent Robot Mobile Local et les ressources matérielles de la base mobile. Cet agent est constitué des modules suivant :

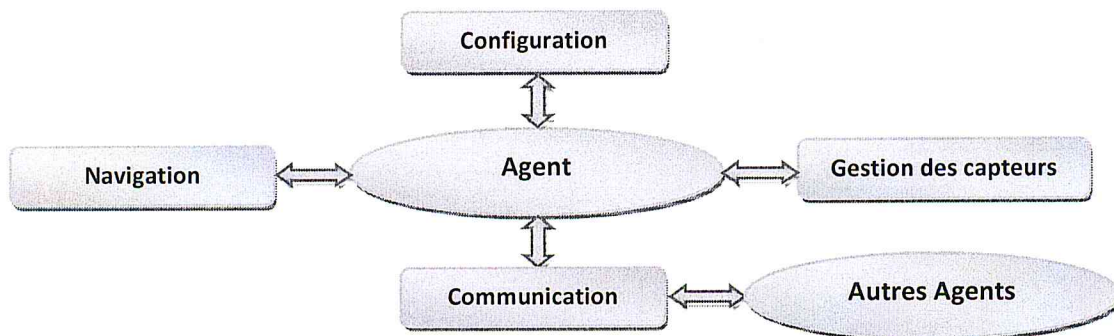


Figure 5: Agent Robot Mobile Distant

- **Module de Communication** : Ce module est conçu de la même manière que le module de communication de l'Agent Robot Mobile Local. Il récolte les informations relatives à l'exécution des opérations, ainsi que l'état de tous les capteurs et les actionneurs et les transmet à l'Agent Robot Mobile Local.
- **Module de Configuration** : il sert à configurer les différents paramètres des équipements avec lesquels l'agent est connecté tels que le mode de lecture des capteurs (télémètre laser LMS, Ultrasons, etc.) ou encore les ports de communication avec les différents agents.
- **Module de Gestion des capteurs** : Ce module permet de récolter toutes les informations concernant les différents capteurs de la base mobile (odomètre, LMS et Ultrasons). Il permet aussi la structuration de ces informations afin qu'elles puissent être exploitées par le Module de Navigation et envoyées à l'Agent Robot Mobile Local.
- **Module de Navigation** : Ce module est le plus important de cet agent. Il calcule les vitesses à envoyer aux actionneurs des roues motrices de la base mobile afin de mouvoir à une position cible tout en évitant les obstacles rencontrés sur sa trajectoire nominale. Pour cela, il a besoin d'exploiter toutes les informations (Données capteurs...) fournies par les autres modules de l'agent.

3.5. Agent Robot Manipulateur Distant

Cet agent (figure 6) est composé de 4 modules. Le Module de Communication et le Module de Configuration ont une conception identique à celle des modules de l'Agent Robot Mobile Distant.

Les autres modules composants cet agent sont donnés dans ce qui suit :

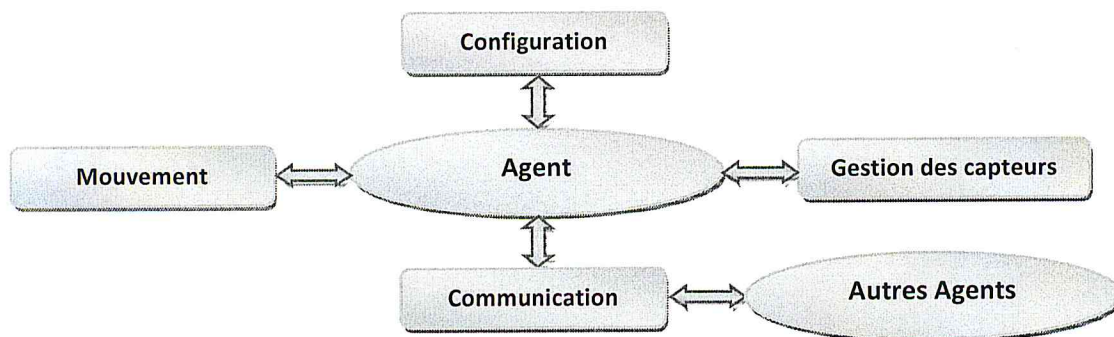


Figure 5: Agent Robot Manipulateur Distant

- **Module de Gestion des capteurs** : Comme le module de gestion des capteurs de l'Agent Robot Mobile Distant, ce module permet la récolte de toutes les informations concernant les capteurs installés sur le bras manipulateur (Capteurs de positions, capteur d'effort, etc.). Ces informations sont par la suite structurées dans un tableau afin de les envoyer à l'Agent Robot Manipulateur Local.
- **Module de Gestion du mouvement** : C'est le module le plus important de cet agent. Ce module calcule les valeurs à envoyer aux actionneurs du bras manipulateur afin de mouvoir à la position cible donnée par (Q_1, \dots, Q_n) avec des vitesses de déplacement de (V_1, \dots, V_n) tels que n est le nombre d'axes du bras manipulateur et les Q_i représentent les variables articulaires.

3.6. Extension de l'architecture à la télérobotique via Internet

L'extension de l'architecture de contrôle consiste à développer un serveur web (IIS) qui va publier l'interface de télérobotique. Cette partie concerne les agents locaux de l'architecture, à savoir,

l'Agent Superviseur, l'Agent Robot Mobile Local, l'Agent Robot Manipulateur Local et l'Agent de Vision telle que montrée par la figure 6.

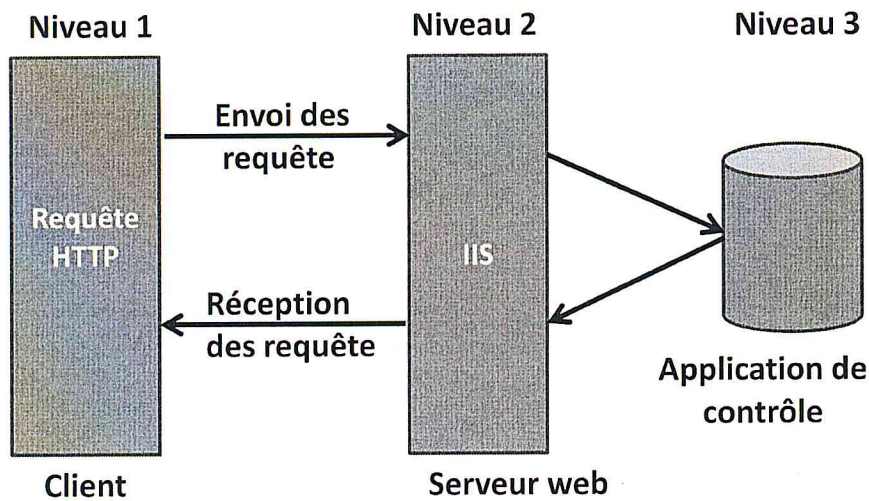


Figure 6: Diagramme de Cas d'Utilisation du système de contrôle

4. Modélisation du système de contrôle

Pour mener à bien le projet, nous avons utilisé le langage *UML (Unified Modeling Language)* qui est considéré comme le standard en matière de modélisation objet. Nous l'utilisons pour décrire de conception du système. UML s'articule autour de 13 diagrammes, chacun d'eux étant dédié à la représentation des concepts particuliers d'un système logiciel du point de vue modélisation. L'ordre de présentation de ces différents diagrammes ne reflète pas un ordre de mise en œuvre dans un projet réel, mais simplement une démarche pédagogique qui essaie de minimiser les pré-requis et les références croisées. Quatre types de diagrammes ont été choisis dans ce travail. Ils sont donnés comme suit :

- **Diagrammes de cas d'utilisation** : permettent de modéliser les besoins du système (A QUOI sert le système, en organisant les interactions possibles avec les acteurs).
- **Diagrammes de classe** : permettent de spécifier la structure et les liens entre les objets dont le système est composé (QUI sera à l'œuvre dans le système pour réaliser les fonctionnalités décrites par les diagrammes de cas d'utilisation).
- **Diagrammes d'activité** : permettent de décrire un processus global ou partiel du système à travers une représentation proche d'un organigramme.
- **Diagrammes de séquences** : permettent de documenter les interactions des différents modules du système (COMMENT les éléments du système interagissent entre eux et avec les acteurs).

4.1. Modélisation des besoins

Le diagramme de cas d'utilisation a été utilisé afin de modéliser les besoins du système. Il sert à définir à quoi sert le système.

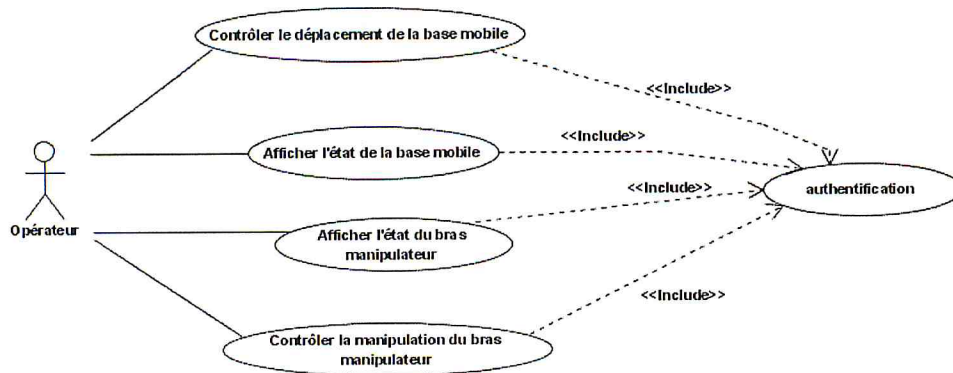


Figure 7: Diagramme de Cas d'Utilisation du système de contrôle

4.1.1. Description détaillée du cas d'utilisation « Déplacement de la base Mobile »

Signification	Élément
Nom	Déplacement de la base Mobile
Description	Ce cas d'utilisation permet de déplacer la base Mobile
Inputs	Coordonnées de la position à attendre
Pré condition	L'Opérateur est s'authentifé La base est connectée
Enchaînements nominaux	Ce cas d'utilisation commence lorsque l'opérateur se connecte au Système 1. Le système affiche un formulaire d'authentification (Login, Password). 2. L'opérateur saisis les informations nécessaires. 3. Le système affiche l'interface de contrôle. 4. L'opérateur connecte la base. 5. L'opérateur saisis les coordonné dans l'interface de contrôle. 6. La base commence le déplacement. 7. Le cas d'utilisation se termine.

Tableau 1: Description détaillée de cas d'utilisation « Déplacement de la base Mobile ».

4.1.2. Description détaillée de cas d'utilisation « Mouvoir le bras Manipulateur »

Signification	Élément
Nom	Mouvoir du bras manipulateur
Description	Ce cas d'utilisation permet la manipulation du bras manipulateur
Inputs	Coordonnées de la position à attendre
Pré condition	L'Opérateur s'est authentifié Le bras est connecté
Enchaînements nominaux	Ce cas d'utilisation commence lorsque l'opérateur se connecte au système 1. Le système affiche un formulaire d'authentification (Login, Password). 2. L'opérateur saisis les informations nécessaires. 3. Le système affiche l'interface de contrôle. 4. L'opérateur connecte le bras. 5. L'opérateur saisis les coordonné dans l'interface de contrôle. 6. Le bras commence la manipulation. 7. Le cas d'utilisation se termine.

Tableau 2 : Description Détaillée de cas d'utilisation « Mouvoir du bras manipulateur »

4.2. Structure et les liens entre les objets

Le diagramme de classe a été utilisé afin de spécifier la structure et les liens entre les objets dont le système est composé.

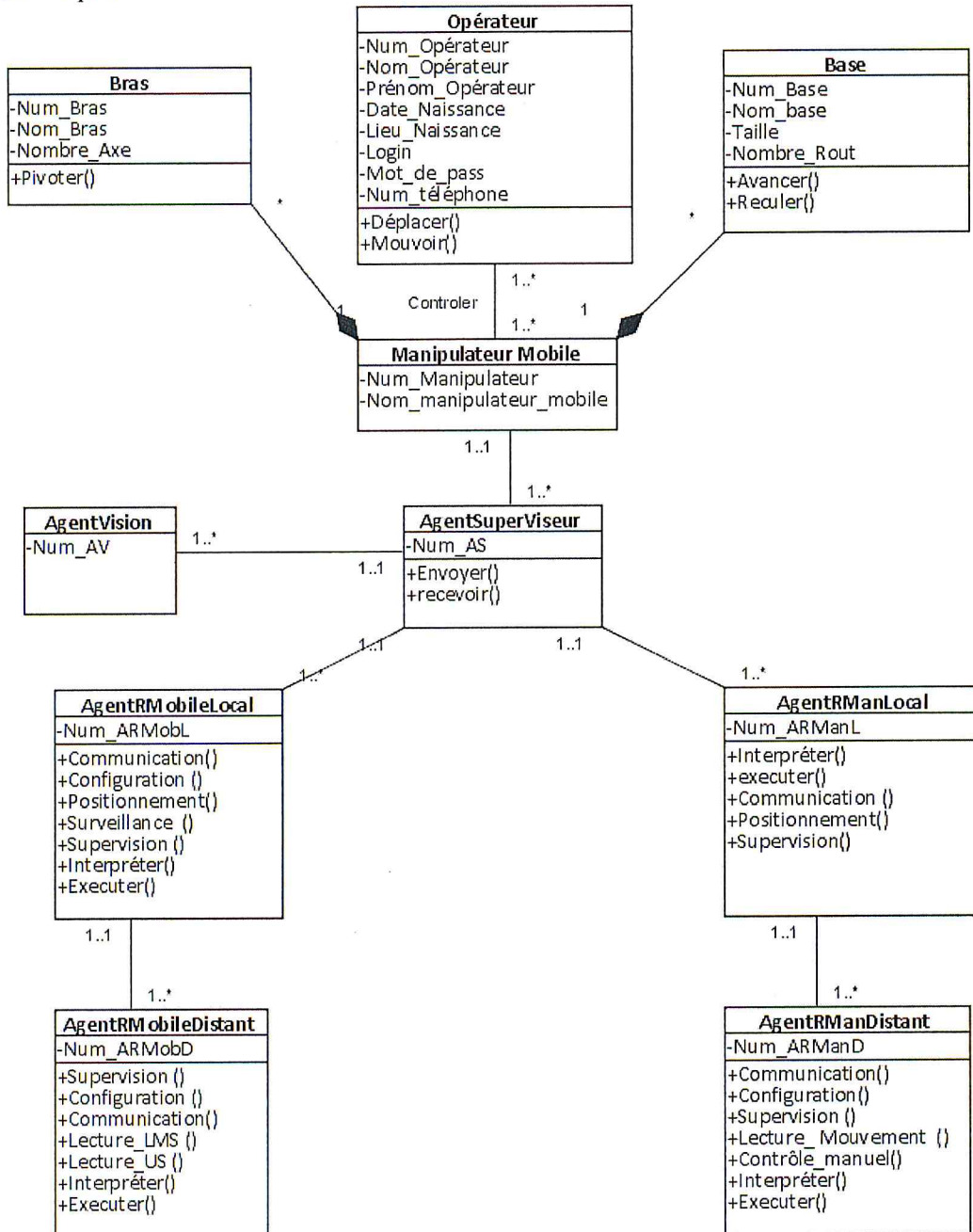


Figure 8: Diagramme de Classe « Contrôler le manipulateur Mobile »

4.3. Diagramme d'activité

Le diagramme de classe a été utilisé afin de spécifier le processus global du système.

4.2. Structure et les liens entre les objets

Le diagramme de classe a été utilisé afin de spécifier la structure et les liens entre les objets dont le système est composé.

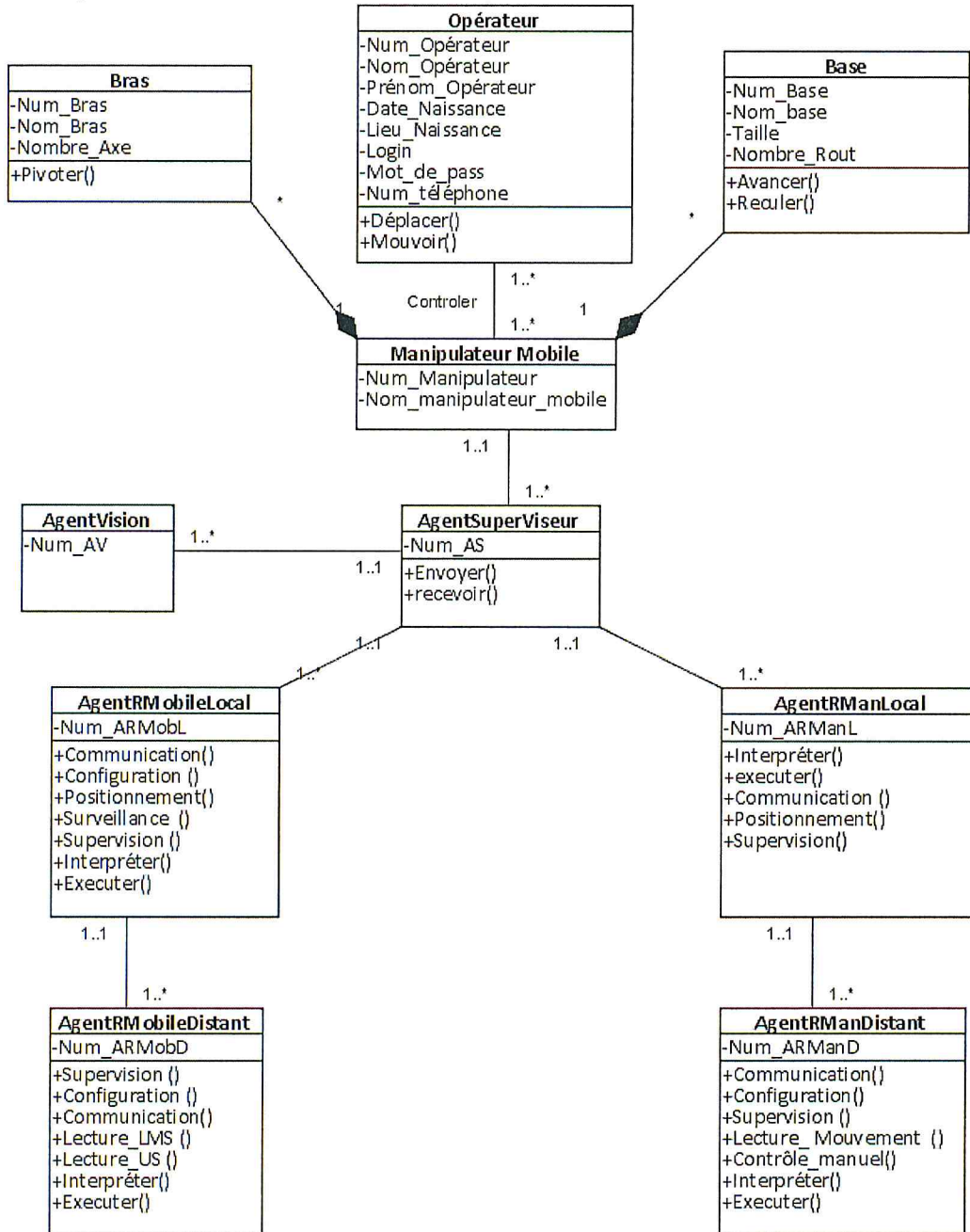


Figure 8: Diagramme de Classe « Contrôler le manipulateur Mobile »

4.3. Diagramme d'activité

Le diagramme de classe a été utilisé afin de spécifier le processus global du système.

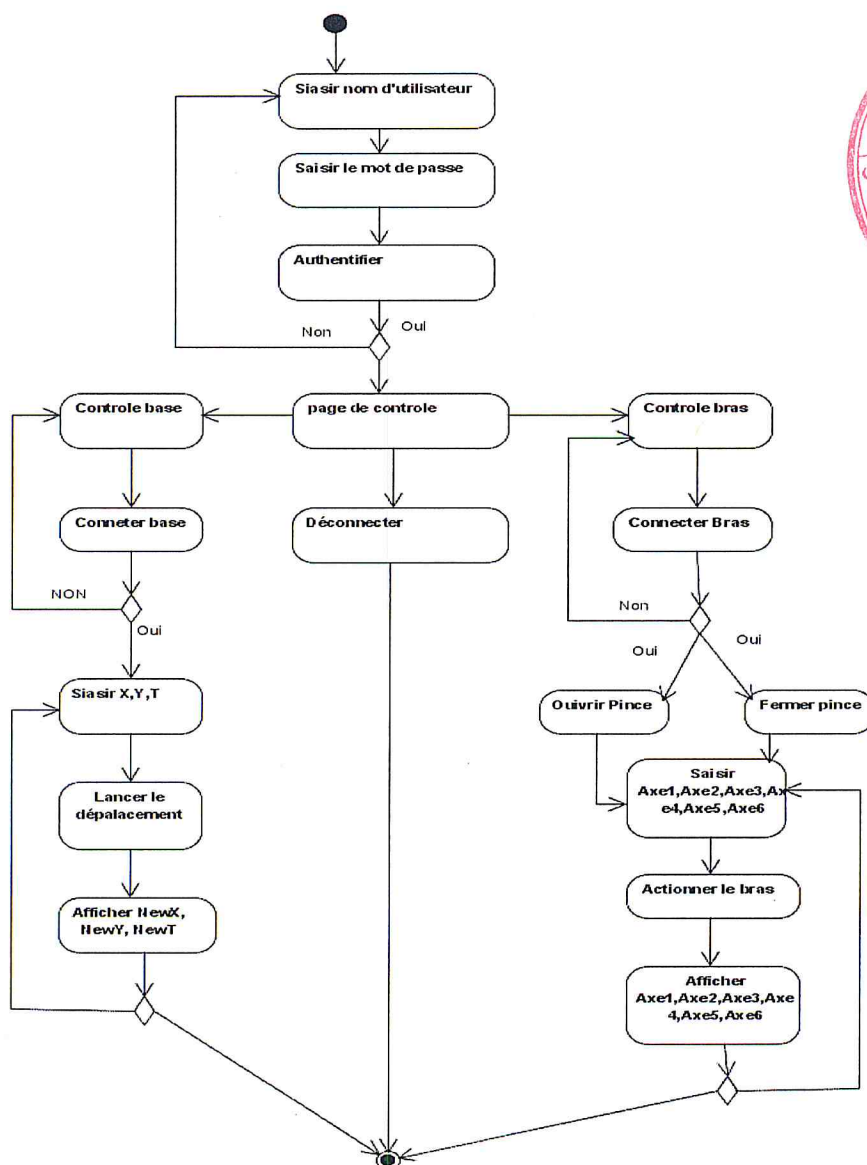


Figure 9: Diagramme d'état de navigation

4.4. Diagrammes de séquence

Dans cette partie, 04 scénarios sont décrits. Ces scénarios correspondent à des fonctionnalités du système. Les scénarios proposés sont les suivants :

- **Scénario 1** : Authentification.
- **Scénario 2** : Déplacer la base Mobile.
- **Scénario 3** : Mouvoir le bras Manipulateur.
- **Scénario 4** : Communication inter-agents.

4.4.1. Authentification

C'est une fonctionnalité qui permet le contrôle d'accès au système.

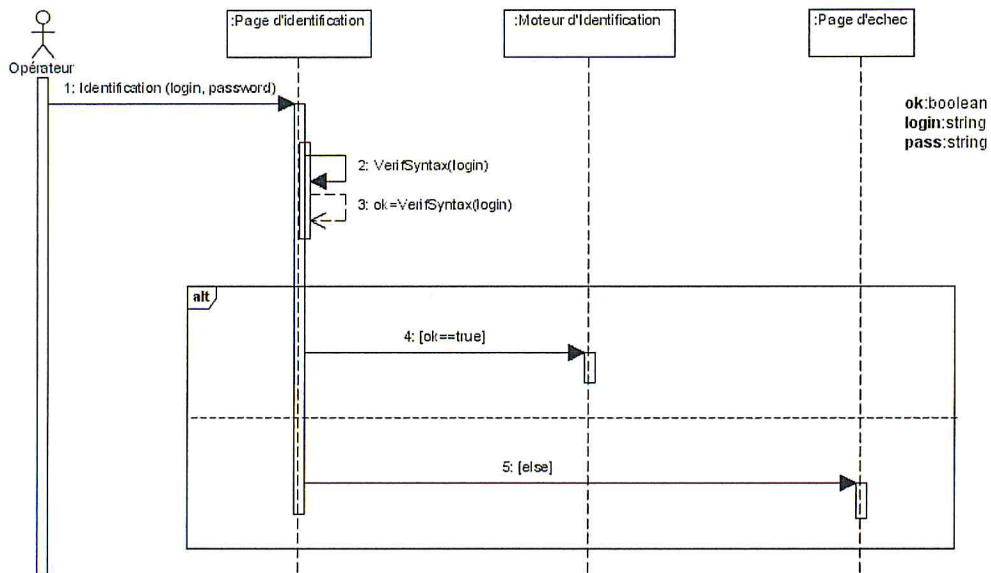


Figure 10: Diagramme de séquence de l'authentification

4.4.2. Déplacer la base Mobile

Cette fonctionnalité permet de déplacer la base mobile a un emplacement donné par les coordonnées X, Y, Théta.

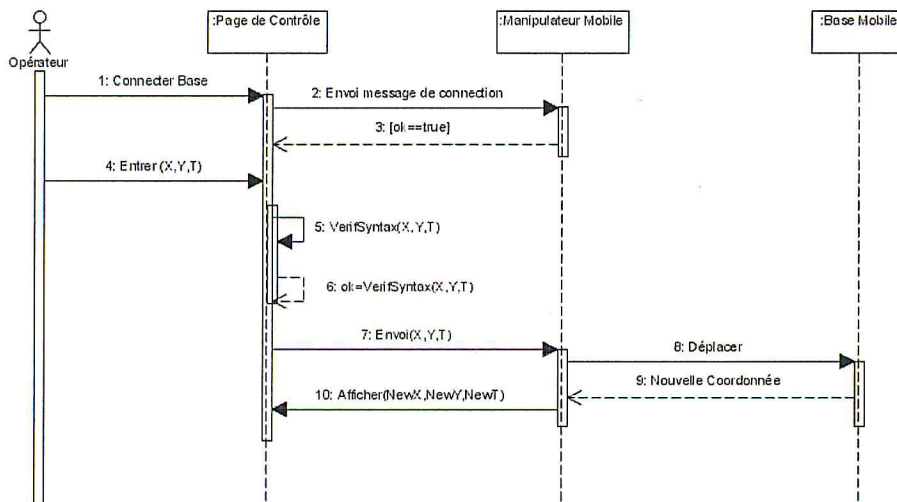


Figure 11: Diagramme de Séquence déplacement de la base mobile

4.4.3. Mouvoir le bras manipulateur

Cette fonctionnalité permet de mouvoir le bras manipulateur selon des axes qui représentent les 06 degrés de liberte.

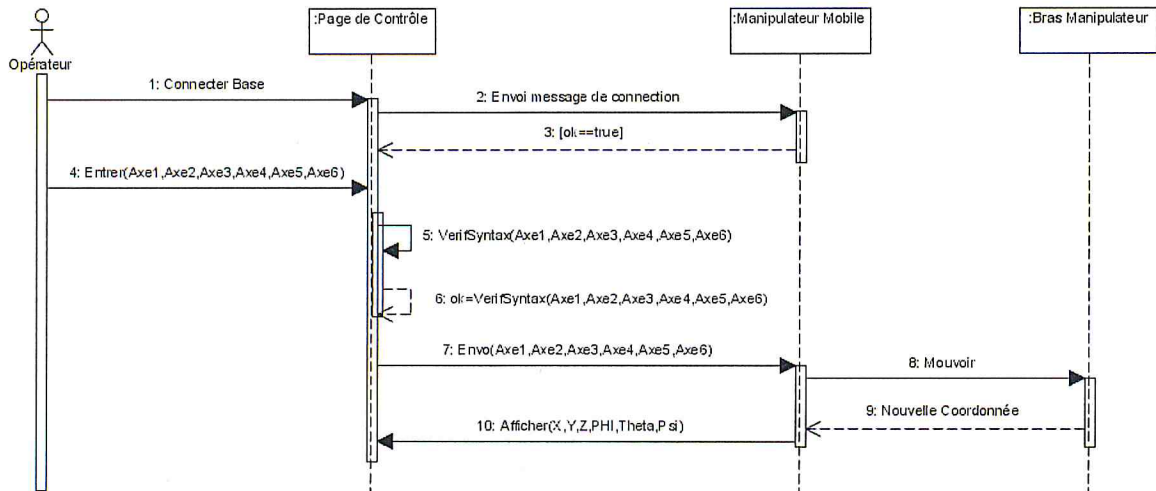


Figure 12: Diagramme de Séquence mouvoir du bras manipulateur

4.4.4. Communication inter-agents

Les agents doivent utiliser un protocole commun pour pouvoir échanger des informations et coopérer pour la résolution d'un problème. Ce protocole permet de structurer et d'uniformiser les interactions inter-agents.

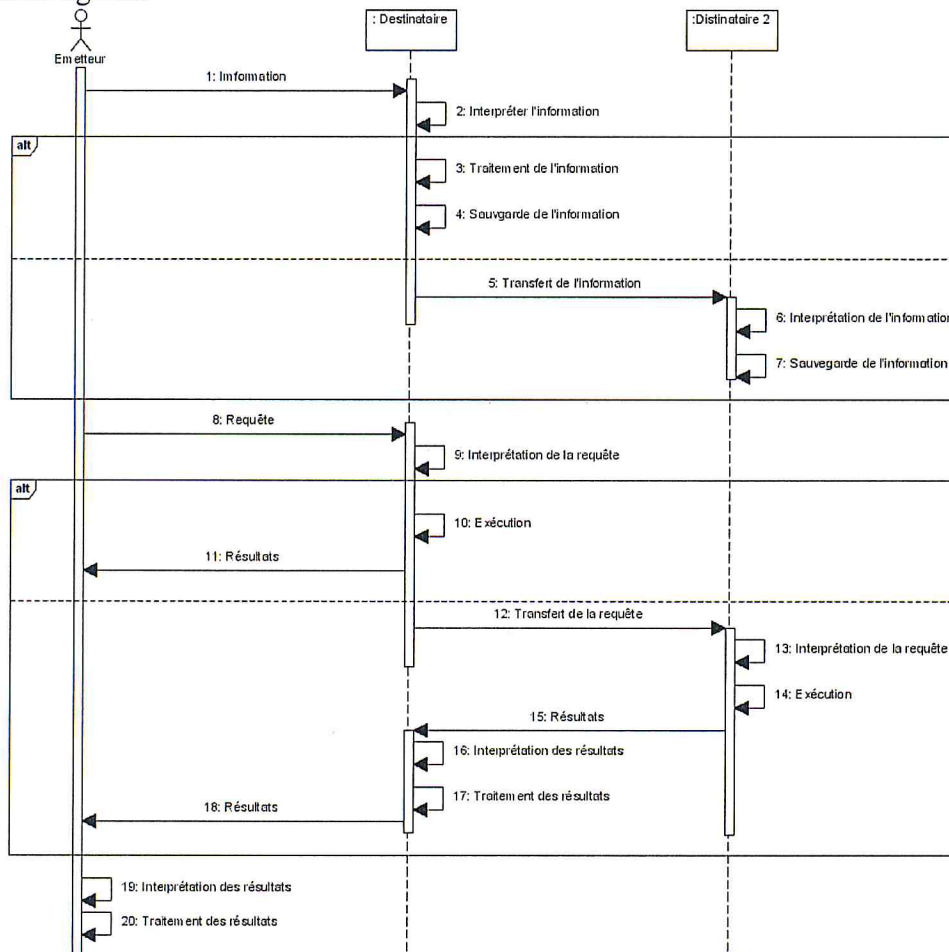


Figure 13: Diagramme de séquence du protocole de communication

5. Conclusion

Ce chapitre a présenté l'architecture proposée pour la télérobotique des robots manipulateurs mobiles, cette architecture est indépendante de l'architecture matérielle et logicielle du robot manipulateur mobile à contrôler.

Les composants du système de contrôle sont modélisés en *UML* en utilisant quelques types de diagrammes, en l'occurrence, le diagramme de cas d'utilisation, le diagramme de classes, le diagramme de composants et le diagramme de séquence.

Le chapitre suivant présente la partie réalisation et mise en œuvre du système de télérobotique du robot manipulateur mobile *RobuTER/ULM*.

Chapitre 4

Implémentation de l'architecture de télérobotique via *Internet*

1. Introduction

Après avoir achevé la partie conception du système de télérobotique, il s'agit de présenter la partie réalisation et mise en œuvre de notre travail. Ce système doit permettre l'introduction des commandes, l'affichage des informations des capteurs en temps réel ainsi que l'analyse des comptes-rendus sur l'exécution des opérations et des commandes envoyées.

Pour cela, nous présentons, l'environnement de travail, les outils de développement utilisés, les différents agents qui composent ce système, leurs implémentations, l'architecture réseau ainsi que l'interface opérateur.

Le système de contrôle se compose de deux parties, une partie embarquée sur le robot manipulateur mobile et une autre partie au niveau local (le serveur web).

2. Aspects d'implémentation

L'une des principales caractéristiques des systèmes multi-agents (SMA) est de permettre de décomposer un problème complexe en sous-problèmes plus facilement implémentable. La conception modulaire des agents facilite considérablement la tâche d'implémentation. Il est, toutefois, nécessaire de préciser qu'aucun agent n'a été développé indépendamment des autres et que la technique du prototypage fonctionnel a été largement utilisée.

3. Environnement de travail

Ce travail a été réalisé au sein de la Division Productique et Robotique du CDTA sur un environnement mixte (Windows/Linux). L'aboutissement du système de contrôle s'est fait en optant pour des choix d'implémentation tels que :

- L'usage de la plateforme Microsoft.Net.
- L'utilisation de l'outil Microsoft Visual Studio 2008 pour le développement de l'application en Asp.Net, Microsoft Silverlight ainsi Visual Paradigm pour la partie Conception en UML.
- SQL serveur comme SGBD.

Remarque : La partie embarquée de l'architecture de télérobotique a été développée avec le langage C/C++ sous RedHat 6.

L'*AS*, l'*ARMoL* et l'*ARMaL* sont installés sur le Serveur Web ou est héberger l'application de contrôle. Ils sont développés sous Visual Studio 2008, tandis que l'*ARMoD* et l'*ARMaD* sont installés sur le PC embarqué. Ces deux derniers agents ont été développés avec le langage C/C++.

4. Protocole de communication

Comme il a été mentionné dans le chapitre 1, un protocole est une stratégie de niveau élevé suivie par des agents logiciels qui interagissent avec d'autres agents. Nombreuses missions exécutées par le robot doivent se dérouler en temps réel. Il est donc nécessaire de mettre en place un protocole de communication simplifié afin de respecter les contraintes d'exécution de certaines missions.

4.1. Connexion entre le serveur et le robot

La connexion est assurée via deux sockets : socket bras pour échanger les commandes du bras et socket base pour échanger les commandes de la base.

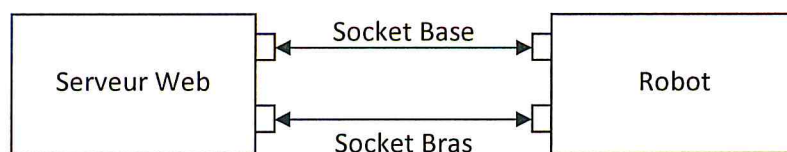


Figure 1 : Connexion entre le serveur et le robot

Une fois le robot est mis en marche, il ouvre les 02 sockets et se mis à l'écoute des requêtes de connexion de l'opérateur.

4.2. Structure des messages échangés dans le système

Les messages échangés entre les différents agents du système sont composés de trois champs : L'émetteur, le destinataire et l'information utile. Cette même information utile est composée de deux champs : Le type et les données utiles. La taille de l'information utile varie d'un message à un

autre. Elle est composée de plusieurs données. Ces données sont séparées par des dièses (#). La figure 1 schématise la structure d'un message.

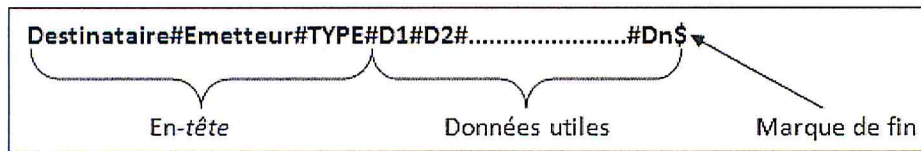


Figure 2: Structure d'un message

Chaque agent dispose de sa propre méthode d'interprétation des messages. Cette interprétation dépend, non seulement du type du message et de la taille de l'information utile, mais aussi de l'émetteur.

Convention : Dans ce qui suit, on se convient d'utiliser les abréviations suivant :

- *AS* : Agent Superviseur
- *ARMaL* : Agent Robot Manipulateur Local
- *ARMoL* : Agent Robot Mobile Local
- *ARMaD* : Agent Robot Manipulateur Distant
- *ARMoD* : Agent Robot Mobile Distant

4.3. Différents messages échangés dans le système

Le tableau qui suit regroupe les différents messages qui peuvent être échangés entre les agents :

En-tête du message	Nombre de données dans l'information	Émetteur	Récepteur	Description de l'information
NAV	3	<i>AS</i>	<i>ARMoL</i>	Requête de positionnement
NAV	3	<i>ARMoL</i>	<i>ARMoD</i>	Requête de positionnement
NAV	3	<i>ARMoD</i>	<i>ARMoL</i>	Position actuelle du robot
NAV	3	<i>ARMoL</i>	<i>AS</i>	Position actuelle du robot
ODO	2	<i>ARMoD</i>	<i>ARMoL</i>	Données de l'odométrie
ODO	2	<i>ARMoL</i>	<i>AS</i>	Données de l'odométrie
US	16	<i>ARMoD</i>	<i>ARMoL</i>	Données des ultrasons
US	16	<i>ARMoL</i>	<i>AS</i>	Données des ultrasons
LMS	181	<i>ARMoD</i>	<i>ARMoL</i>	Données du capteur LMS
LMS	181	<i>ARMoL</i>	<i>AS</i>	Données du capteur LMS
GB	2	<i>AS</i>	<i>ARMoL</i>	Requête pour contrôle manuel
GB	2	<i>ARMoL</i>	<i>ARMoD</i>	Requête pour contrôle manuel
POS	6	<i>AS</i>	<i>ARMaL</i>	Requête de positionnement
POS	6	<i>ARMaL</i>	<i>ARMaD</i>	Requête de positionnement
POS	13	<i>ARMaD</i>	<i>ARMaL</i>	Position Actuelle du bras + Les valeurs des capteurs d'efforts
PS	2	<i>AS</i>	<i>ARMaL</i>	Requête pour contrôle manuel
PS	2	<i>ARMaL</i>	<i>ARMaD</i>	Requête pour contrôle manuel
BRAS	0	<i>ARMaL</i>	<i>ARMaD</i>	Requête pour recevoir les données des capteurs d'efforts
PINCE	1	<i>ARMaL</i>	<i>ARMaD</i>	Ouvrir ou Fermer la pince
O	0	<i>ARMoL</i>	<i>ARMoD</i>	Activer les capteurs d'odométrie
U	0	<i>ARMoL</i>	<i>ARMoD</i>	Activer les capteurs ultrasons
AL	0	<i>ARMoL</i>	<i>ARMoD</i>	Activer le capteur LMS
EL	0	<i>ARMoL</i>	<i>ARMoD</i>	Requête pour recevoir les données du capteur LMS

STOP	0	<i>ARMoL</i>	<i>ARMoD</i>	Arrêt d'urgence
STOP	0	<i>ARMaL</i>	<i>ARMaD</i>	Arrêt d'urgence

Tableau 1: Différents messages échangés entre les agents [66]

5. Développement des Agents

Chaque agent doit pouvoir répondre aux événements produit par son environnement et faire face à toutes les situations et ce, dans les plus brefs délais dans le but de respecter la dynamique du robot. Par conséquent, chaque agent est représenté par un processus père et une multitude de threads qui s'exécutent d'une manière autonome et parallèle. Ces threads représentent les différents modules de l'agent.

5.1 Partie embarquée

Cette partie a été développée avec le langage C/C++ car l'ordinateur embarqué ne dispose que du compilateur C/C++ (gcc). Deux processus sont lancés à partir d'un seul programme source. Chaque processus correspond à un agent. Chaque processus est constitué d'un certain nombre de threads qui s'exécutent en parallèle et d'une manière autonome (Cette partie existe déjà).

La figure 2 regroupe les légendes des différents éléments utilisés dans la schématisation des agents.

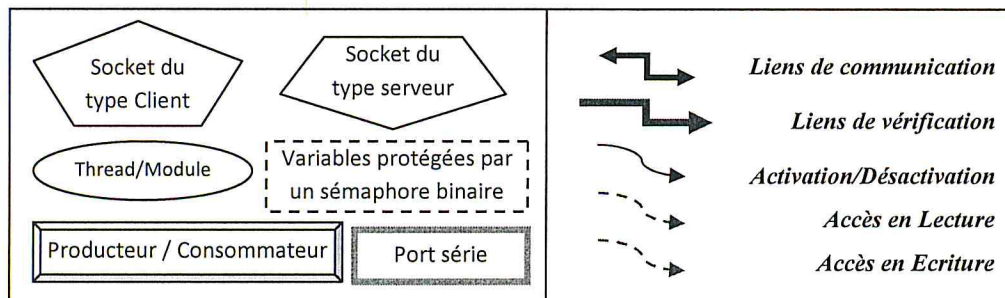


Figure 3: Légende des différents composants

5.2 . Partie Locale (Serveur web / Off-board)

Cette partie est constituée de quatre agents : l'*AS*, l'*ARMoL*, l'*ARMaL*, l'*Agent Vision* hébergés dans un serveur web (IIS) qui publie l'interface de contrôle. Ces agents sont implémentés en Visual C# 2008. L'application de contrôle est implémentée sous ASP.NET.

Afin de permettre l'exécution parallèle, chaque agent est associé à un processus indépendant des autres. Les modules de chaque agent sont implémentés sous formes de bibliothèques de liens dynamiques (Dynamic Link Library - DLL). Chaque module peut être sollicité afin d'exécuter un service qui peut être un calcul, un affichage de message, une structuration de données, une interprétation d'un message, ...etc.

L'interface web fournit les différentes fonctionnalités assurées par les quatre agents.

5.2.1 . Agent Robot Mobile Local (ARMoL)

Lors du lancement du processus correspondant à l'*ARMoL* (Figure 5) une instance de chaque module est créée. Ces modules sont au nombre de cinq :

- **Communication** : Ce module est composé de deux parties distinctes. La première partie permet la communication avec l'*ARMoD* (communication de bas niveau) et la deuxième partie (communication de haut niveau) permet la communication avec les agents locaux (*AS*, *ARMaL*). Les deux parties comportent les fonctions standards de communication, à savoir :
 - Connecter_Bas_Niveau (Adresse_IP, Port) ;
 - Connecter_Haut_Niveau (Adresse_IP, Port) ;
 - Déconnecter () ;
 - Recevoir () ;

- Interpréter (Message) ;
- Construire_Message (Données, Destinataire) ;
- Envoyer (Message) ;
- **Configuration** : Ce module permet la récupération à partir d'un fichier de tous les paramètres de connexion de notre système. Ce fichier contient l'adresse IP de l'ordinateur embarqué ainsi que tous les ports de connexion de tous les agents avec lesquels cet agent doit se connecter.
 - **Private** Ouvrir_Fichier (Nom_Fichier) ;
 - **Private** Fermer_Fichier (Nom_Fichier) ;
 - **Private** Lire_Fichier () ;
 - **Private** Interpréter () ;
 - **Public** Get_ROBOT_IP () ;
 - **Public** Get_Port_Réception () ;
 - **Public** Get_Port_Envoi () ;
- **Positionnement** : Deux fonctions principales caractérisent ce module :
 - La première permet de calculer la position d'une cible dans R_A connaissant sa position dans R_B et la position de R_B dans R_A .
 - La deuxième permet de calculer la position d'une cible dans R_B connaissant sa position dans R_A et la position de R_B dans R_A .

L'algorithme correspondant à ces deux méthodes est régi par le modèle de la base mobile.

 - **Public** Position_Cible_Base (New_X, New_Y, New_T, X_Cible, Y_Cible) ;
 - **Public** Position_Cible_Repère (New_X, New_Y, New_T, X_C_B, Y_C_B) ;
- **Surveillance** : Ce module veille au bon fonctionnement de l'ARMoD. Ceci se fait en vérifiant que tous les modules de l'agent fonctionnent le plus normalement possible. Dans le cas contraire, il crée une nouvelle instance du module qui a fait l'objet d'une erreur. Ce module permet d'inscrire dans un fichier *.log* tous les messages reçus et émis par cet agent, afin de pouvoir faire, ultérieurement, une détection d'anomalies concernant l'environnement de l'agent.
 - **Public** Vérificateur () ; // Cette méthode s'exécute dans un thread résident
 - **Public** Inscription_Log () ; // Cette méthode s'exécute dans un thread résident
- **Supervision** : Lors de la réception d'un message, le module de *communication* l'interprète et l'oriente automatiquement vers ce module. Ce dernier va solliciter le module correspondant à la requête afin de réaliser le service demandé. Après la récolte des résultats de la requête, ce module envoie les données au module de *communication*, qui va construire le message et l'envoyer au destinataire correspondant.

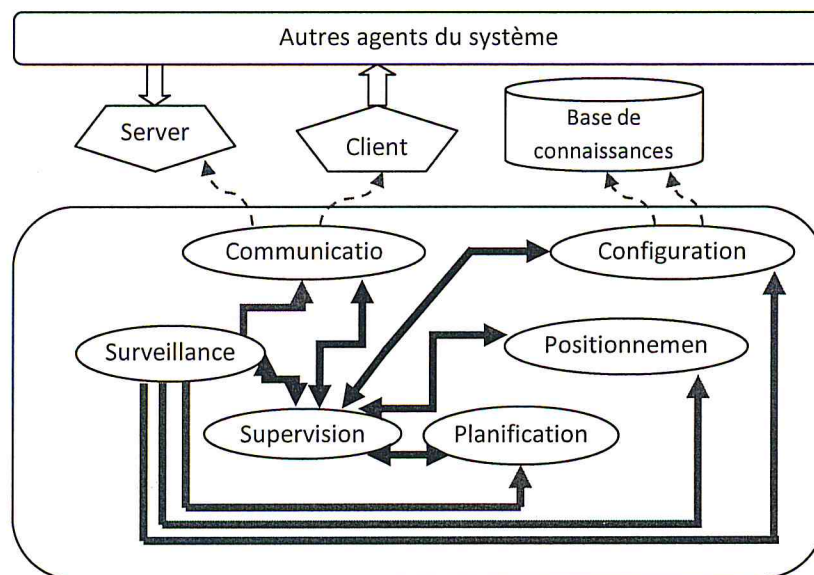


Figure 4: Architecture modulaire de l'ARMoL/l'ARMaL

5.2.2 . Agent Robot Manipulateur Local (ARMaL)

L'*ARMaL* (Figure 5) a été implémenté de la même manière que l'*ARMoL*. Il dispose donc des mêmes modules (*Supervision, Configuration, Surveillance, Communication, Positionnement*) qui ont des rôles identiques à ceux de l'*ARMoL* à quelques exceptions près :

- **Communication** : Une seule méthode le différencie du module de communication de l'*ARMoL*, à savoir, la méthode d'interprétation des messages. En effet, ces deux agents utilisent des types de messages différents. Il est donc logique d'avoir deux méthodes d'interprétation différentes.
- **Positionnement** : C'est dans ce module que réside la principale différence entre l'*ARMoL* et l'*ARMaL*. Ce module est composé de trois parties :
 - **Le MGD** : Le modèle géométrique direct d'un bras manipulateur a pour but le calcul de la position (x, y, z) et l'orientation (Φ, Θ, Ψ) de l'effecteur à partir des variables articulaires $(Q_1, Q_2, Q_3, Q_4, Q_5, Q_6)$.
Public MGD $(Q_1, Q_2, Q_3, Q_4, Q_5, Q_6)$; // Elle retourne $P(X, Y, Z, \Phi, \Theta, \Psi)$;
 - **Le MGI** : C'est le modèle qui permet d'exprimer la configuration du système en fonction de la situation de l'effecteur (ou les coordonnées généralisées $(X_E, Y_E, Z_E, \Phi_E, \Theta_E, \Psi_E)$ en fonction des coordonnées opérationnelles $(Q_i, i = 1..6)$). Une amélioration a été apportée au calcul du MGI afin de faciliter son utilisation. Cette amélioration consiste à générer toutes les orientations possibles correspondantes à une position $P(X, Y, Z)$. Ensuite, les configurations en coordonnées opérationnelles sont générées pour chaque combinaison d'orientations. La configuration $C(Q_i, i = 1..6)$, représentant l'erreur minimale entre la position P et la position $P'(X', Y', Z')$ tel que $P' = \text{MGD}(C)$, sera choisie.

Générer_Solution_MGI(X, Y, Z){

MAX = π ;

Tant que (Phi <= MAX){

Tant que (Tetha <= MAX){

Tant que (Psi <= MAX){

Solution = MGI(X, Y, Z, Phi, Tetha, Psi);

Si (une solution existe) **Alors** Sauvegarder la solution ;

Psi = Psi + PAS;}

Tetha = Tetha + PAS;

Psi = $-\pi$;}}

Phi = Phi + PAS;

Tetha = $-\pi$;

Psi = $-\pi$;}}

Pour chaque (Configuration $C \in$ Solutions){

$P' = \text{MGD}(C)$;

$$Err = \sqrt{(X_{p'} - X_p)^2 + (Y_{p'} - Y_p)^2 + (Z_{p'} - Z_p)^2}$$

Si (Err < ϵ) **Alors retourner** P' ;

}

retourner pas de solution ;

}

- **Espace de travail** : Cette méthode permet de déterminer si un point de l'espace fait partie de l'espace de travail du robot.
- **Supervision** : Ce module joue le même rôle que celui de l'*ARMoL*.

5.2.3 . Agent Superviseur

Cet agent joue le rôle de l'interface entre l'opérateur et le système robotisé. Comme tout autre agent, l'*AS* dispose des modules standards, à savoir, un module de *Communication*, un module de *Configuration*, un module de *Supervision*. De plus il dispose de quelques modules qui le différencient des autres agents (figure 6) :

- **Supervision** : Le rôle de ce thread est d'activer le(s) thread(s) correspondant(s) au type du message et aux données reçues par le thread de communication.
- **Configuration** : Ce thread permet la récupération des paramètres de connexion se trouvant dans un fichier. Ce fichier contient les paramètres de connexion des agents de l'architecture de contrôle.
- **Communication** : Il sert d'interface de communication de l'agent avec ses accointances. Il contient les fonctions usuelles de communication telles que *Connect*, *Stop*, *Envoi* et *Recevoir*.

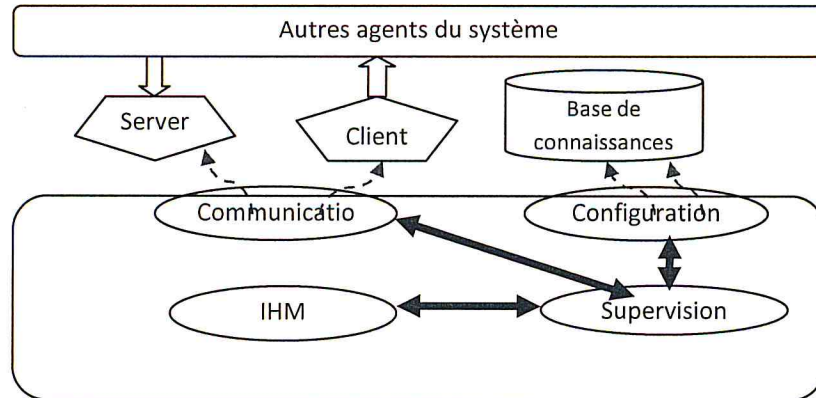


Figure 5: Architecture modulaire de l'AS


6. Conclusion

L'utilisation d'un système multi-agents pour le contrôle d'un robot manipulateur mobile a facilité la tâche d'implémentation de l'architecture proposée.

Le chapitre suivant présente la partie réalisation test expérimentale de notre travail.

Chapitre 5

Tests expérimentaux



1. Introduction

Afin de pouvoir valider l'architecture proposée dans les chapitres précédents, nous avons mis en place un environnement réel de tests. Ce dernier nous a servi pour la réalisation d'une multitude d'expériences dans le but d'analyser les performances de l'architecture proposée. Dans ce qui suit, nous présentons l'environnement de tests, quelques tests réalisés et les principaux résultats obtenus.

2. Environnement de tests

L'environnement de tests consiste en deux parties :

2.1 Serveur web

Le rôle du serveur web est d'héberger l'application web développée en ASP.Net. Pour cela, il doit avoir les services IIS (Internet Information Service) ainsi que les composants dépendant de IIS installés. Le détail d'installation de ces services sont données en à l'annexe B.

Une fois les pré-requis sont installés sur le serveur, le site web doit être hébergé dans le service IIS via la console dédiée. Cette opération est détaillé dans l'annexe C. À ce moment le serveur est prêt pour recevoir les requêtes d'un client autorisé.

2.2 Client de contrôle

Le client peut être n'importe quel équipement capable d'exécuter un navigateur web, comme PC, Laptop, PDA, Smartphone, etc. Il sert à se connecter au serveur web afin d'avoir accès à la page de l'application de contrôle qui permet de lancer des commandes sur le robot de n'importe où. Comme nous l'avons signalé, l'unique application requise du coté client est n'importe quel navigateur web.

3. Système robotique expérimental

Le *RobuTER/ULM* est le robot à manipuler qui a fait l'objet de notre travail. Il est mis à notre disposition au niveau de la division Productique et Robotique du CDTA.

3.1 Architecture du manipulateur mobile RobuTER/ULM

RobuTER/ULM est composé d'une base mobile rectangulaire sur laquelle est monté un manipulateur (Figure 1).

Le robot est contrôlé par un PC MMX industriel on-board et par quatre cartes microcontrôleurs MPC555 (Figure 1). Un bus CAN permet la communication entre le PC on-board et les cartes microcontrôleurs [67]. Les cartes MPC555 contrôlent l'ensemble des actionneurs du robot ainsi que quelques capteurs. La première carte est attachée à la base mobile. Deux autres cartes sont désignées pour le contrôle du manipulateur (respectivement les trois premières et les trois dernières articulations). La dernière carte MPC555 contrôle le capteur d'effort six axes.

3.2 Architecture de la base mobile RobuTER

RobuTER, est une base mobile à deux roues motrices actionnées par des moteurs électriques à courant continu. La direction du RobuTER est donnée par le différentiel des vitesses des deux roues motrices. Les deux roues folles à l'avant de la base mobile assurent la stabilité et l'équilibre de l'ensemble.

RobuTER dispose d'une ceinture de 24 capteurs ultrasons. Chaque capteur est formé par un émetteur et un récepteur ultrason.

Un télémètre laser LMS SICK200 est installé sur la base mobile. C'est un scanner de mesure dans le plan horizontale avec un angle de 0° à 180° .

3.3 Architecture du manipulateur ULM

Le manipulateur Ultraléger ULM de 6 ddl, est installé sur la base mobile RobuTER pour le transport ou la manipulation de petites pièces, le ramassage d'échantillon, etc. [67].

Le manipulateur est équipé d'un capteur de position des axes. Tous ces codeurs sont relatifs qui délivrent un nombre d'impulsion proportionnel à la rotation effectuée de l'axe.

Le manipulateur est équipé aussi d'un capteur d'effort six axes intégré sur la pince qui délivre, en continue, les forces (F_x , F_y , F_z) et les couples (T_x , T_y , T_z).

3.4 Caméra CCD et système de transmission/réception des images vidéo

Une caméra CCD est installée sur la pince du manipulateur ULM, aussi, un système de transmission/réception HF utilisant la voie hertzienne comme canal de transmission et les ondes électromagnétiques comme support d'informations est installé. Il est composé de deux modules :

- un module émetteur installé sur le PC on-board qui reçoit l'image vidéo issue de la caméra sous format analogique et l'envoie vers le récepteur.
- un module récepteur installé sur le PC off-board qui récupère l'image vidéo et l'introduit dans une carte d'acquisition vidéo Pinnacle 500PCI pour numérisation et exploitation.

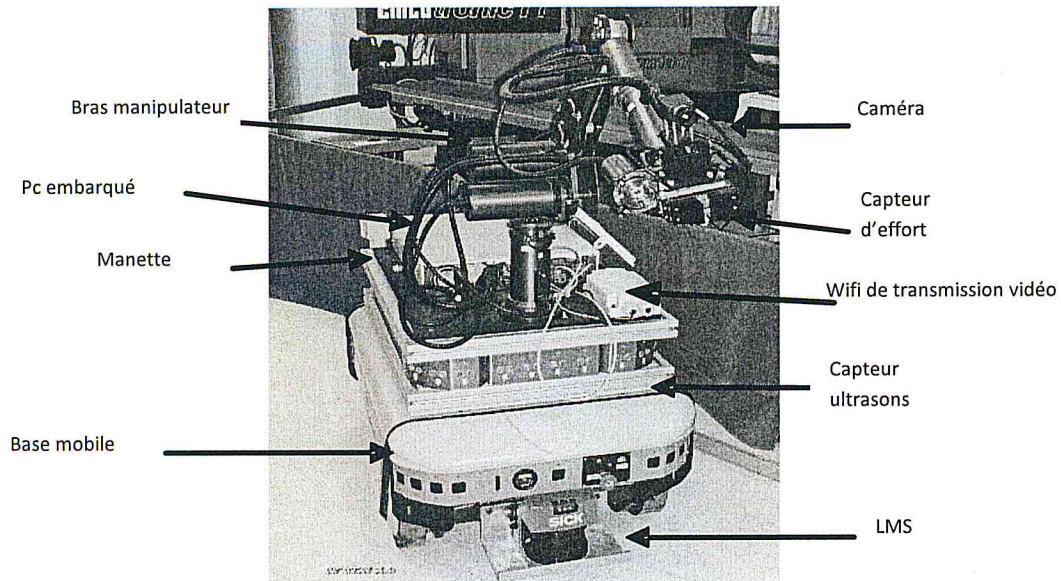


Figure 1: Manipulateur mobile RobuTER/ULM

4. Interface Homme/Robot

4.1 Page d'authentification

Suite à la connexion de l'opérateur au serveur, une page d'authentification apparaît (figure 2), l'opérateur doit saisir le nom et le mot de passe, pour accéder à la page de contrôle.



Figure 2: page d'authentification

4.2 Interface de télérobotique du RobuTER/ULM

Une fois le nom et mots de passe sont vérifiés, la page de contrôle apparaît (figure 2). Cette page est partagée en quatre parties comme suit :

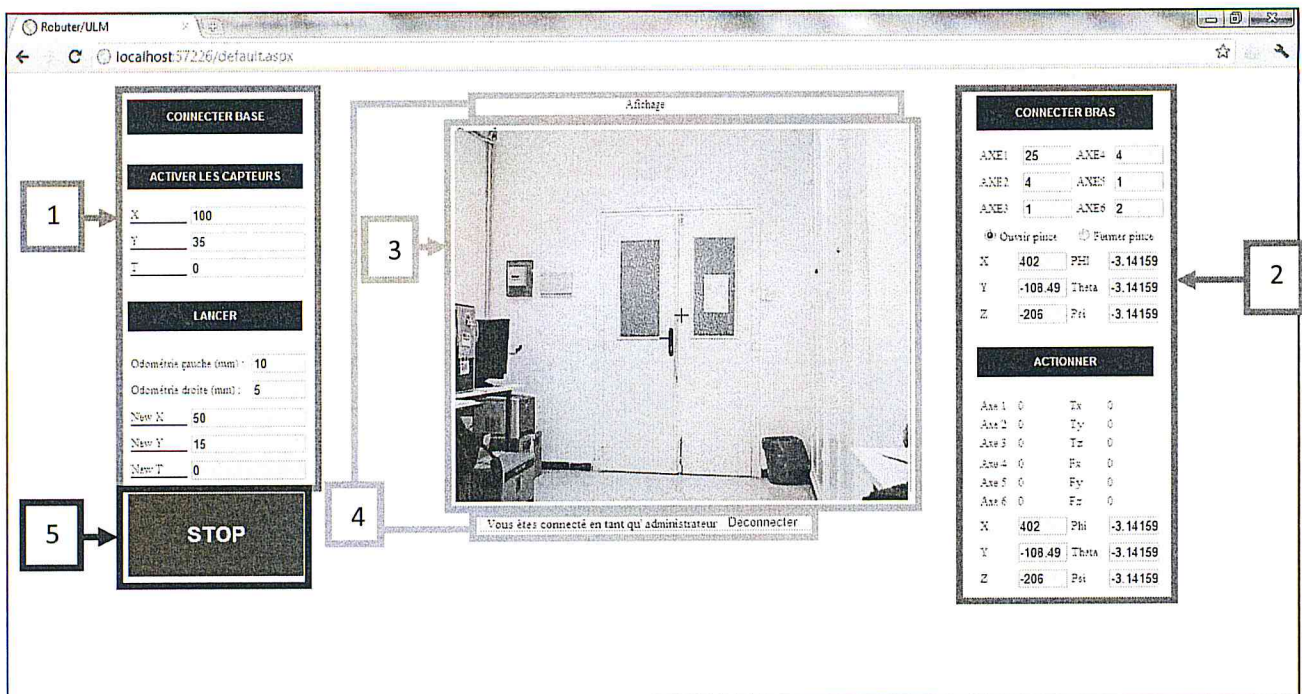


Figure 3: Interface opérateur pour le contrôle du RobuTER/ULM

4.2.1 Partie 1

Cette partie donnée par la figure 4 interagit avec l'ARMoL. Elle affiche toutes les données relatives aux capteurs de la base mobile, à savoir, la situation courante de la base mobile (New_X , New_Y , New_T), les valeurs issues des deux encodeurs droit et gauche (E_R , E_L) et les données odométriques (Odo_X , Odo_Y , Odo_T). Cette partie permet aussi de lancer la lecture des capteurs LMS et US, et de lancer le thread de calcul de l'odométrie. Finalement, elle permet d'ordonner à la base mobile de se mouvoir vers une situation donnée en spécifiant (X , Y , T).

CONNECTER BASE

ACTIVER LES CAPTEURS

100
 35
 0

LANCER

Odométrie gauche (mm) :
 Odométrie droite (mm) :
 New X
 New Y
 New T

STOP

Figure 4: Partie concernant la base mobile

4.2.2 Partie 2

Cette partie donnée par la figure 5 interagit avec l'ARMaL. Elle permet d'actionner le manipulateur, soit axe par axe (Axe_1 , ..., Axe_6), soit en spécifiant une situation donnée par (x , y , z , ψ , θ , ϕ).

CONNECTER BRAS

AXE1:	<input type="text" value="25"/>	AXE4:	<input type="text" value="4"/>
AXE2:	<input type="text" value="4"/>	AXE5:	<input type="text" value="1"/>
AXE3:	<input type="text" value="1"/>	AXE6:	<input type="text" value="2"/>

Ouvrir pince Fermer pince

X:	<input type="text" value="402"/>	PHI:	<input type="text" value="-3.14159"/>
Y:	<input type="text" value="-108.49"/>	Theta:	<input type="text" value="-3.14159"/>
Z:	<input type="text" value="-206"/>	Psi:	<input type="text" value="-3.14159"/>

ACTIONNER

Axe 1	<input type="text" value="0"/>	Tx	<input type="text" value="0"/>
Axe 2	<input type="text" value="0"/>	Ty	<input type="text" value="0"/>
Axe 3	<input type="text" value="0"/>	Tz	<input type="text" value="0"/>
Axe 4	<input type="text" value="0"/>	Fx	<input type="text" value="0"/>
Axe 5	<input type="text" value="0"/>	Fy	<input type="text" value="0"/>
Axe 6	<input type="text" value="0"/>	Fz	<input type="text" value="0"/>

X:	<input type="text" value="402"/>	Phi:	<input type="text" value="-3.14159"/>
Y:	<input type="text" value="-108.49"/>	Theta:	<input type="text" value="-3.14159"/>
Z:	<input type="text" value="-206"/>	Psi:	<input type="text" value="-3.14159"/>

Figure 5: Partie concernant le bras manipulateur

4.2.3 Partie 3

Cette partie concerne l'agent de vision. Elle offre la possibilité d'afficher, en temps-réel, les images vidéo délivrées par la caméra, de capturer une image et d'effectuer les traitements nécessaires. Cette partie permet aussi d'effectuer la reconstruction 3D, soit à partir de deux images gauche et droite, soit en utilisant une image avec les données issues du capteur LMS.

4.2.4 Partie 4

Zone d'affichage des messages. visualise le résultat à la suite de chaque opération. Elle indique l'opération réalisée ou les éventuelles erreurs survenues. L'opérateur peut se déconnecter à l'aide du lien « Déconnecter ».

4.2.5 Partie 5

Elle sert à arrêter complètement le robot (base mobile et manipulateur).

5. Tests expérimentaux

Dans cette section, nous allons exposer en détail le déroulement des manipulations du robot via notre application dès la connexion du client à l'exécution de la commande par le robot.

5.1. Temps de connexion

La connexion à l'application de contrôle du robot (connexion aux deux serveurs de la base mobile et au celui du bras manipulateur) a été testée dans trois cas différents selon le type de connexion. Le premier type consiste à établir une connexion directe via un câble réseau. Le deuxième exploite le réseau local du CDTA pour se connecter. Le dernier type utilise le réseau Internet. Les résultats obtenus sont résumés dans le tableau 1 et illustrés par l'histogramme de la figure 6 suivante :

Type de connexion	Temps de connexion (secondes)		
	Câble	Réseau local	Internet
Bras	8.97	8.99	9.01
Base	8.97	10.66	12.34

Tableau 1: Temps de connexion selon le type du réseau utilisé

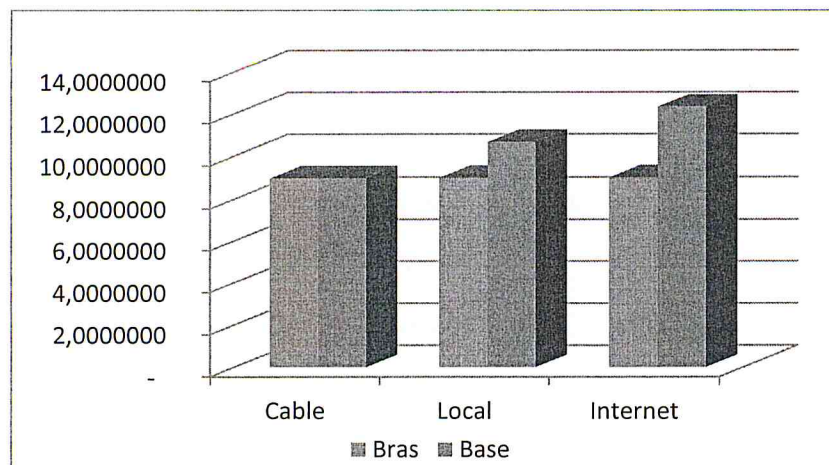


Figure 6: Temps de connexion selon le type du réseau utilisé

5.2. Exécution de tâches élémentaires

Quatre types de tâches élémentaires exécutées par le manipulateur mobile sont illustrés dans ce paragraphe :

Situation initiale du manipulateur mobile

- La situation initiale de la base mobile $Base_{Init}(X_{BInit}, Y_{BInit}, T_{BInit})=(0, 0, 0^\circ)$.
- La configuration initiale du manipulateur $Configuration_{Init}(Q_{1Init}, Q_{2Init}, Q_{3Init}, Q_{4Init}, Q_{5Init}, Q_{6Init})=(0, 0, 0, 0, 0, 0)$.
- La situation initiale de l'effecteur $Effector_{Init}(x_{EInit}, y_{EInit}, z_{EInit}, \psi_{EInit}, \theta_{EInit}, \varphi_{EInit})=(-432, -108.49, 164, -180^\circ, -180^\circ, -180^\circ)$.

Tâche 1 : Faire mouvoir le manipulateur à une situation donnée par $Effector_{Fin}(x_{EFin}, y_{EFin}, z_{EFin}, \psi_{EFin}, \theta_{EFin}, \varphi_{EFin})=(-330\text{mm}, -630\text{mm}, 1080\text{mm}, -135^\circ, -88^\circ, 5^\circ)$

La configuration finale du manipulateur correspondant à cette situation, calculée par le MGI, est $Configuration_{Fin}(Q_{1Fin}, Q_{2Fin}, Q_{3Fin}, Q_{4Fin}, Q_{5Fin}, Q_{6Fin})=(-60^\circ, 61^\circ, 30^\circ, 95^\circ, -15^\circ, 0^\circ)$.

Les variations des coordonnées articulaires du manipulateur sont données par Figure 6.

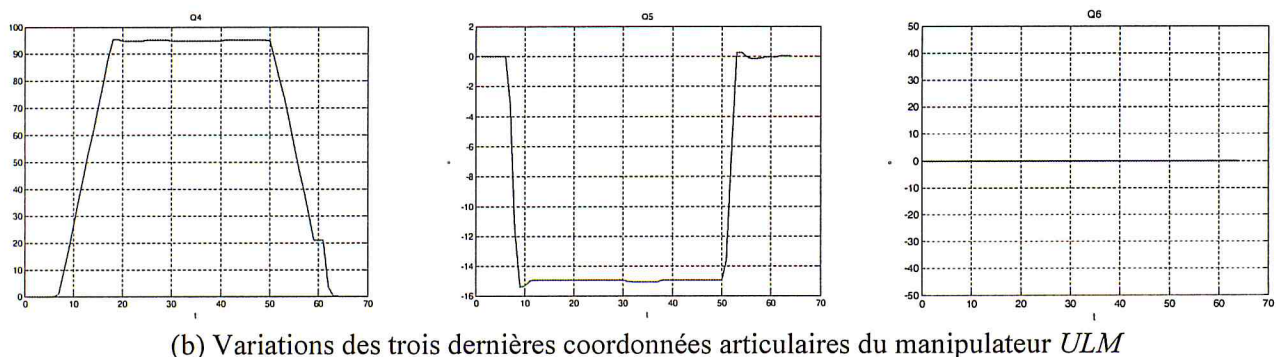
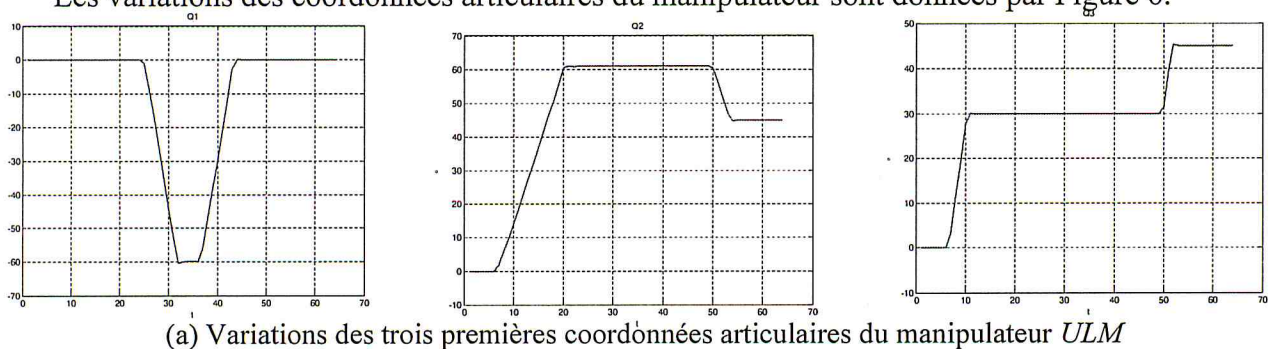


Figure 7: Les variations des coordonnées articulaires du manipulateur ULM

Tâche 2 : Faire mouvoir le manipulateur à une plusieurs situations données comme suit :

- $Configuration_{Init}(Q_{1Init}, Q_{2Init}, Q_{3Init}, Q_{4Init}, Q_{5Init}, Q_{6Init})=(0, 0, 0, 0, 0, 0)$ à l'instant $T=94s$, le manipulateur se trouve à la position $Configuration_{Init}$.
- $Configuration_{Open}(Q_{1Open}, Q_{2Open}, Q_{3Open}, Q_{4Open}, Q_{5Open}, Q_{6Open})=(0, 40, 28, 0, 0, 0)$ à l'instant $T=116s$, le manipulateur se trouve à la position $Configuration_{Init}$.
- $Configuration_{Fin}(Q_{1Fin}, Q_{2Fin}, Q_{3Fin}, Q_{4Fin}, Q_{5Fin}, Q_{6Fin})=(20, 32, 28, 0, 0, 0)$ à l'instant $T=156s$, le manipulateur se trouve à la position $Configuration_{Init}$.
- $Configuration_{Relax}(Q_{1Relax}, Q_{2Relax}, Q_{3Relax}, Q_{4Relax}, Q_{5Relax}, Q_{6Relax})=(0, 45, 45, 0, 0, 0)$.

La situation finale correspondant à $Configuration_{Fin}$ est donnée par $Effector_{Fin}(x_{EFin}, y_{EFin}, z_{EFin}, \psi_{EFin}, \theta_{EFin}, \varphi_{EFin})=(-815mm, -108.49mm, 650mm, -90^\circ, -90^\circ, -90^\circ)$

Les variations des coordonnées articulaires du manipulateur sont données par Figure 7.

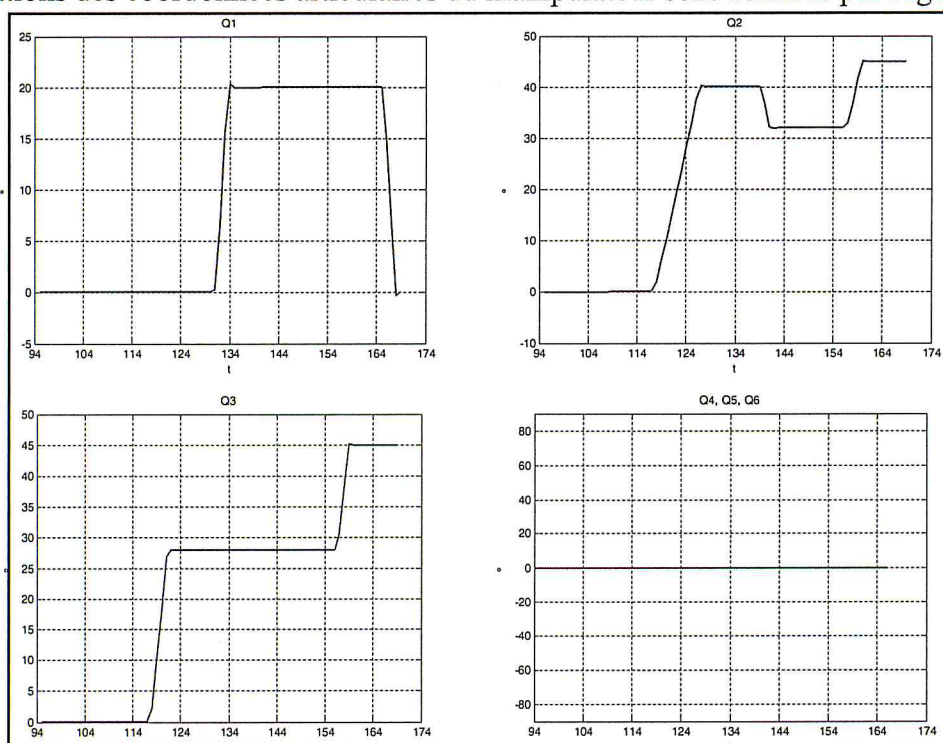


Figure 8: Variations articulaires du manipulateur ULM

Tâche 3 : Faire mouvoir la base mobile de sa situation initiale à une situation finale donnée par $B_{aseFin}(X_{BFin}, Y_{BFin}, T_{BFin})=(-1920, 2, 15^\circ)$ en présence d'un obstacle qui se trouve à la position $Obstacle(x_{Ob}, y_{Ob}, z_{Ob})=(-1000, 0, 50)$ d'une taille de $(800 \times 200 \times 100)$ mm (figure 9).

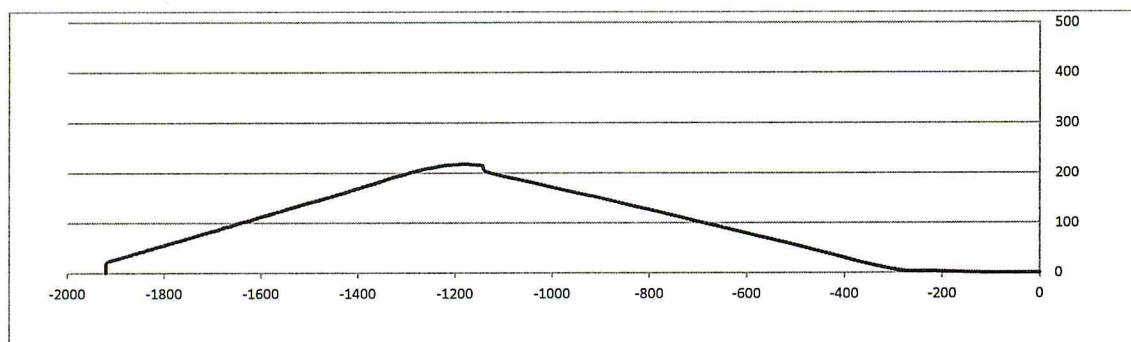


Figure 9: Trajectoire réelle suivie par la base mobile

Tâche 4 : Faire mouvoir la base mobile de sa situation initiale à une situation finale donnée par $Base_{Fin}(X_{BFin}, Y_{BFin}, T_{BFin})=(-3440, 13, 12^\circ)$.

Deux obstacles dans l'environnement. Le premier se trouve à la position $Obstacle_1(x_{Ob1}, y_{Ob1}, z_{Ob1})=(-1000, 400, 50)$ de $(800 \times 200 \times 100)$ mm. Le deuxième obstacle est à la position $Obstacle_2(x_{Ob2}, y_{Ob2}, z_{Ob2})=(-2000, -400, 50)$ d'une taille de $(600 \times 250 \times 100)$ mm (figure 8).

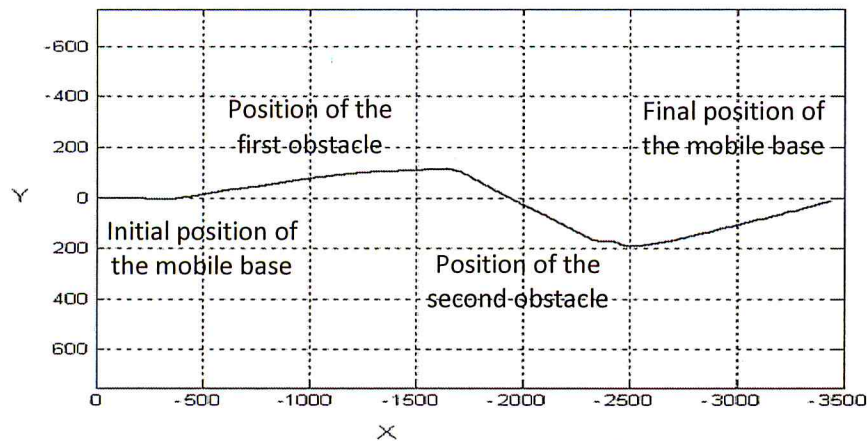


Figure 10: Trajectoire réelle suivie par la base mobile ainsi que l'évitement des obstacles présents

Le tableau 2 donne les temps moyens d'exécution (en secondes) des différentes opérations considérées dans ce chapitre.

	Opération 1	Opération 2	Opération 3	Opération 4
Temps exécution (secondes)	64	166	58	56

Tableau 2: Temps d'exécution des différentes opérations

6. Conclusion

Dans ce chapitre, nous avons présenté l'interface de l'application web de contrôle, via Internet, du manipulateur mobile RobuTER/ULM. Grâce à cette interface, l'opérateur dispose d'un tableau de bord pour le contrôle à distance du robot.

La connexion aux deux serveurs installés sur le robot (base mobile et bras manipulateur) a été testée dans le cas (i) d'une connexion directe via un câble réseau, (ii) connexion via le réseau local du CDTA, (iii) connexion via le réseau Internet.

Le chapitre a présenté aussi quelques tests de quelques tâches primitives tels que (i) faire mouvoir le manipulateur axe par axe, (ii) déplacer la base mobile de sa situation actuelle à une situation désirée, etc.

Cette interface peut être complétée par l'intégration d'autres fonctionnalités comme l'alerte à la détection d'un obstacle par les capteurs, etc.

Conclusion générale et perspectives

Notre travail consistait en le développement d'une application web client pour la télérobotique du manipulateur mobile *RobuTER/ULM* via *Internet*.

Après avoir présenté quelques types des manipulateurs mobiles existant, nous avons exploré les principaux travaux de la littérature portant sur la télérobotique. Depuis les années 70, différents projets de télérobotique ont été réalisés de part le monde.

De nos jours, avec l'évolution des technologies de la communication, les applications de la télérobotique sont de plus en plus utilisées pour réaliser différentes tâches. Particulièrement, on fait appel à la télérobotique pour réaliser des tâches complexes, risquée, voir impossible tel que l'exploration spatial, sous-marine, l'intervention dans les milieux hostiles, etc.

D'après l'étude de l'état de l'art, nous avons constaté que les différents travaux réalisés reposent sur une architecture type constituée de quatre composants :

- Composant de représentation graphique (vision) du robot dans l'environnement de travail ;
- Composant de détection de collisions ;
- Composant d'interface pour l'interaction avec l'opérateur ;
- Composant de communication.

Pour notre cas d'application, à savoir le *RobuTER/ULM*, l'architecture de contrôle est un système multi-agent, où chaque composant représente un ou plusieurs agents. Elle est constituée d'un ensemble d'agents hybrides qui pilotent l'ensemble de ressources hétérogènes du robot (base mobile, bras manipulateurs et système de vision) :

- Agent Superviseur.
- Agent de Vision.
- Agent Robot Manipulateur Local.
- Agent Robot Mobile Local.
- Agent Robot Manipulateur Distant.
- Agent Robot Mobile Distant.
- et un serveur web (installé au niveau du PC off-board).

Pour l'implémentation de l'architecture télérobotique du RobuTER/ULM différentes technologies ont été utilisées :

- Au niveau du robot (PC on-board), le langage C sous Linux est utilisé. Cette partie existe déjà. Elle comporte l'agent Robot Manipulateur Distant et l'agent Robot Mobile Distant.
- Au niveau du serveur, nous avons exploité la technologie Web de Microsoft.Net (ASP.Net et C#). Cette partie comporte le serveur web IIS, l'agent superviseur, l'agent de vision, l'agent Robot Manipulateur Local et l'agent Robot Mobile Local. C'est sur cette partie que porte l'essentiel de notre travail.


L'application développée offre à l'opérateur une interface web accessible via Internet pour le contrôle du RobuTER/ULM. Une fois connecté au serveur web via un navigateur, l'opérateur doit s'authentifier pour avoir accès à différentes commandes à exécutées par le robot. Ces commandes sont en deux catégories : contrôle du bras et contrôle de la base mobile. De plus, l'interface permet à l'utilisateur de visualiser l'environnement du robot via l'affichage de la vidéo envoyée par la caméra placée sur le robot. La caméra peut être pilotée via l'interface web offrant ainsi une souplesse de visualisation de l'environnement.

Enfin, la télérobotique via le web ouvre plusieurs perspectives pour des applications intéressantes. L'application web que nous avons développée constitue une plateforme de contrôle du RobuTER/ULM. Nous envisageons comme suite de notre travail l'exploitation de cette plateforme pour le développement de différentes applications, à savoir :

- **Saisie d'objet :** Parmi les capacités du RobuTER/ULM, la manipulation d'objets joue un rôle primordial. Cette capacité est aussi un élément essentiel dans l'exploration et la modification de l'environnement du robot. Un manipulateur mobile doit, au moins, être capable de réaliser des tâches simples telles que la saisie et le transport d'objet.
La mission consiste à saisir un objet qui se trouve à une position X tout en évitant les obstacles éventuels. Au lieu que l'opérateur humain du robot positionne manuellement la base mobile et contrôle le manipulateur afin de saisir l'objet désiré, il spécifie seulement la position X de ce dernier. Les agents de l'architecture de contrôle génèrent, par la suite, le plan d'opérations, font mouvoir la base mobile et le manipulateur et, enfin, saisissent l'objet.
- **Appuie sur des boutons d'une machine :** La mission consiste à automatiser des tâches d'appuie sur des boutons d'une machine dans un milieu hostile ou dangereux a un opérateur humain. Au lieu que l'opérateur du robot spécifie la position du boutons sur lequel il faut appuyés, il clique simplement sur le bouton en question sur l'image issue de la caméra CCD. Les agents de l'architecture de contrôle fonctionnent, après, de façon autonome afin de générer un plan d'opérations, calculer les coordonnées 3D réelles du bouton, faire mouvoir le robot vers la machine et, enfin, appuyés sur le bouton.
- **Suivi d'une trajectoire :** La mission consiste à suivre une trajectoire opérationnelle prédéfinie tout en évitant les obstacles éventuels. En automatisent cette tâche, le robot devient capable d'écrire sur un tableau, d'effectuer une tâche de soudage continu, de pendre un mur, etc.

Annexe A

Agents et Systèmes multi-agents



1. Introduction

De nos jours, les systèmes informatiques sont devenus de plus en plus complexes. Ils sont souvent répartis sur plusieurs sites et sont constitués de logiciels en interaction. Afin d'accroître l'intégrité, de faire inter-opérer et d'augmenter la coopération entre ces logiciels existants, les notions d'agents et de systèmes multi-agents (SMA) ont apparues. La notion d'agent est utilisée en plusieurs domaines, entre autres, la sociologie, la biologie et l'informatique.

L'utilisation des SMA pour le contrôle distribué est largement répandue. Ces systèmes offrent un modèle de contrôle décentralisé basé sur des agents. Ici, plusieurs éléments, en l'occurrence les agents,

coopèrent les uns avec les autres pour atteindre leurs propres objectifs. L'ensemble de ces objectifs, synthétisés, constituent l'objectif final du SMA. Les opérations à accomplir pour atteindre l'objectif global sont réparties entre les agents (selon les compétences) et chacun accomplit son opération spéciale.

La nécessité d'une distribution de l'activité et de l'intelligence est justifiée par les raisons suivantes [68] :

- *Les problèmes sont souvent physiquement distribués.*
- *Les problèmes sont fonctionnellement très distribués et hétérogènes.*
- *Les systèmes de contrôle doivent pouvoir s'adapter de plus en plus à des modifications de structure et d'environnement* : devant la complexité de plus en plus croissante des applications, il est nécessaire de doter les systèmes de contrôle de capacités d'évolution telles que l'ajout de nouvelles fonctionnalités, la modification de l'utilisation, etc.
- *Les réseaux imposent une vision distribuée* : à l'heure des réseaux où toute l'information et la puissance de traitement sont réparties sur un certain nombre de nœuds, il est nécessaire de penser en termes de systèmes distribués.
- *La complexité des problèmes impose une vision locale* : lorsque les problèmes sont trop vastes et trop compliqués pour être analysés globalement, les solutions fondées sur des approches locales permettent souvent de les résoudre plus rapidement.
- *Le Génie Logiciel va dans le sens d'une conception en termes d'unités autonomes en interactions (des agents)* : l'histoire du développement des logiciels montre que la réalisation des programmes informatiques suit une démarche visant à la réalisation de systèmes conçus comme des ensembles d'entités de plus en plus distribués, mettant en jeu des composants autonomes.
- *La conception de systèmes complexes émerge vers une nouvelle discipline de recherche (celle des SMA)* : cette nouvelle discipline repose sur des concepts propres très puissants qui permettent de proposer des solutions réactives et robustes pour la conception de systèmes complexes.

Cette deuxième annexe porte sur les principes des agents, des SMA, ainsi que sur leurs principales caractéristiques individuelles et collectives.

2. Agent

L'IAD (Intelligence artificielle distribuée) et les SMA étudient des techniques de résolution collective des problèmes par des entités appelées agents et situées au sein d'un même système. Les SMA sont généralement employés pour résoudre des problèmes difficiles dans les domaines de l'IAD. Ces SMA sont notamment employés pour simuler le comportement d'une collectivité naturelle (cellules, insectes, etc.) ou pour créer des systèmes artificiels réalisant une fonction globale dans lesquels les connaissances, les traitements et le contrôle sont distribués :

- L'environnement dans lequel sont plongés les agents est dynamique.
- Les agents n'ont qu'une perception partielle de leur environnement.
- Les agents ont des capacités cognitives restreintes.
- Les agents ne peuvent pas tout traiter individuellement et doivent s'associer pour réaliser l'activité globale.

2.1. Définition d'un agent

Il n'existe pas encore un consensus sur la définition d'un agent [69]. Les définitions les plus significatives sont citées dans ce qui suit :

- Un agent est une entité autonome, réelle ou abstraite, qui est capable d'agir sur elle-même et sur son environnement qui, dans un univers multi-agents, peut communiquer avec d'autres agents, et

dont le comportement est la conséquence de ses observations, de ses connaissances et de ses interactions avec les autres agents [68].

- Un agent est un système informatique situé dans un environnement dans lequel il est capable d'effectuer une action flexible et autonome, compatible aux objectifs de la conception [70].
- Un agent est une entité logicielle ou physique à qui est attribuée une certaine opération qu'elle est capable d'accomplir de manière autonome et en coopération avec d'autres agents [69].
- Les agents sont des entités clairement identifiables de résolution de problèmes avec des bornes et des interfaces bien définies et destinés à atteindre un objectif spécifique. Ils sont autonomes et responsables de leur comportement et, capables d'adopter un comportement flexible pour résoudre des problèmes selon les objectifs de la conception. Les agents sont situés dans un environnement particulier et reçoivent des entrées liées aux états de cet environnement par des capteurs et, agissent sur cet environnement par des émetteurs. Les agents sont réactifs et *proactifs* (sont capables d'adopter un nouvel objectif) et sont capables, dans un univers multi-agents, de communiquer, coopérer, se coordonner et négocier les uns avec les autres [71].

La figure 1 donne, de façon générale, l'architecture interne d'un agent [72]. Lorsqu'un agent perçoit une situation dans l'environnement, il essaie de la reconnaître. Si la situation lui est familière, il peut enclencher un processus de planification afin de résoudre le problème. Il peut aussi reconnaître la situation en termes d'actions et passe donc à l'exécution de la tâche (Reconnaissance/Exécution). Lorsque l'agent perçoit des situations qu'il connaît très bien, il peut faire intervenir son comportement réactif en passant directement à l'action (Perception/Exécution). S'il ne peut pas résoudre un problème (situation non-familière), il engage un processus de coopération pour demander de l'aide aux autres agents (Reconnaissance/Prise de décisions) [71].

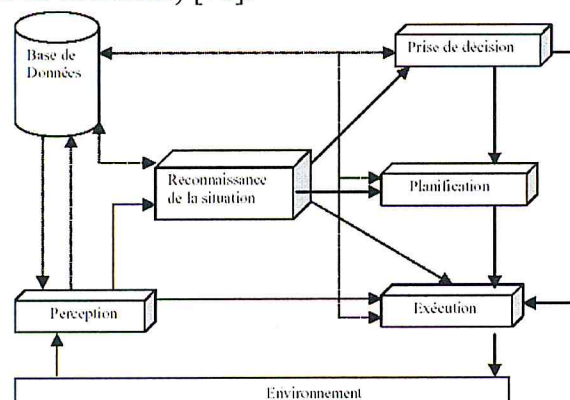


Figure 1: Architecture générale d'un agent

2.2. Caractéristiques d'un agent

Les caractéristiques principales d'un agent peuvent se dérouler des définitions données précédemment. Elles peuvent être résumées dans ce qui suit [73]:

- *Autonomie* : la propriété principale dont doit faire preuve tout agent et celle de l'autonomie, c.-à-d., qu'il doit pouvoir agir et prendre des décisions seul, en fonction des informations lui provenant des autres agents et de l'environnement.
- *Engagement* : un agent planifie ses opérations par coordination et négociation avec les autres agents. En construisant un plan pour atteindre son objectif, l'agent se donne les moyens d'y parvenir et, donc, s'engage à accomplir les opérations qui le satisfont.
- *Sociabilité* : l'agent doit être capable d'interagir avec les autres agents quand la situation l'exige afin de compléter ses opérations ou aider ces agents à accomplir les leurs. Cette perspective

sociale implique que l'accent soit mis sur le groupe plutôt sur l'agent individuel. Elle ne considère pas que le social soit en contradiction avec l'autonomie.

- *Intelligence* : c'est la capacité de l'agent à accepter les demandes de l'opérateur et d'amener à bien l'opération que lui est déléguée. À un degré d'intelligence plus élevé, l'agent devrait comprendre ce que l'opérateur veut, et planifier les moyens à mettre en œuvre pour atteindre cet objectif. À un degré supérieur, l'agent, non seulement planifie ses propres opérations, mais aussi tient compte des autres.

2.3. Classification des agents

Les entités composant un SMA peuvent être des agents réactifs, des agents cognitifs ou des agents hybrides, suivant leur rôle dans le système.

2.3.1. Agent réactif

Les agents réactifs ne disposent que d'un langage et d'un protocole de communication réduit [74]. Leurs capacités répondent uniquement à la loi stimuli/action (Figure 2). Un SMA réactif repose sur la coopération d'agents de faible *granularité* (complexité des fonctionnalités d'un agent) mais beaucoup plus nombreux.

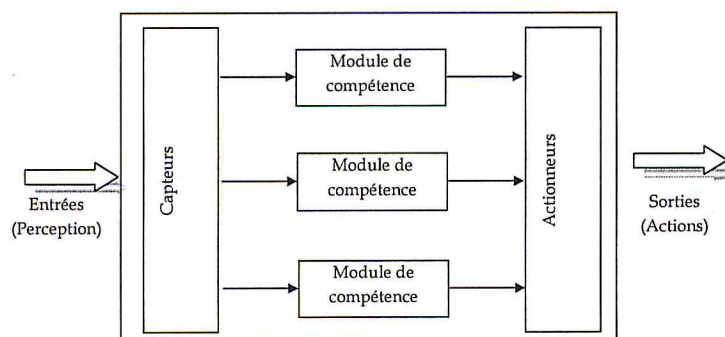


Figure 2: Architecture générale d'un agent réactif

Les inconvénients de ce type d'agents sont les suivants [74] :

- Si les agents réactifs ne possèdent pas un modèle de leur environnement, ils doivent posséder suffisamment d'informations locales leur permettant de choisir une action acceptable.
- Comme les agents réactifs basent leurs décisions sur des informations locales, il est difficile de voir comment ils pourraient tenir compte des informations non locales.
- Il est difficile de voir comment un agent purement réactif peut apprendre de son expérience et améliorer, ainsi, ses performances.

2.3.2. Agent cognitif (délibératif)

Un SMA cognitif comprend un petit nombre d'agents qui disposent d'une capacité de raisonnement sur une base de connaissances et, d'une aptitude à traiter des informations diverses. Ces dernières sont, soit liées au domaine d'application, soit celles relatives à la gestion des interactions avec l'environnement ou avec les autres agents. Chaque agent est assimilable, suivant le niveau de ses capacités, à un système expert plus ou moins sophistiqué. On parle, dans ce cas, d'agent de forte granularité.

Un agent délibératif ou cognitif (Figure 3) doit, donc, mettre à jour ses *croyances* ¹ avec les informations qui lui proviennent de son environnement, décider quelles options lui sont offertes (ses

¹ *Croyances* : ce sont ce que l'agent connaît de son environnement.

désirs²), filtrer ces options afin de déterminer de nouvelles intentions³ et poser ses actions au vu de ses intentions [75].

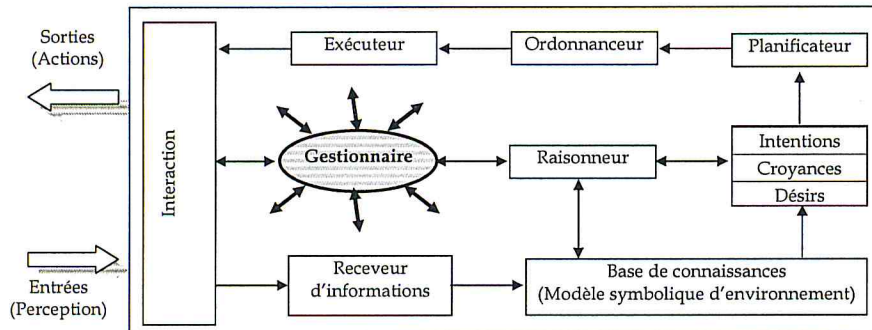


Figure 3: Architecture générale d'un agent cognitif de type BDI (Believe, Desir, Intention)

Pour la construction des SMA cognitifs, deux problèmes importants doivent être résolus [74] :

- Le problème de la traduction de l'univers de l'agent en une description symbolique, en tenant compte du temps.
- Le problème de représentation/raisonnement qui consiste à trouver une représentation symbolique des informations de l'univers complexe des entités et des processus et, à trouver comment les agents peuvent raisonner sur ces informations.

2.3.3. Agent hybride

Dans le cas des architectures hybrides, un agent est généralement composé de plusieurs couches, arrangées selon une hiérarchie [76]. La plupart des architectures ne considèrent que trois couches. Au plus bas niveau de l'architecture, on retrouve une couche purement réactive, qui prend ses décisions en se basant sur des données brutes en provenance des capteurs. La couche intermédiaire fait abstraction des données brutes et travaille, plutôt, avec une vision qui se situe au niveau des connaissances de l'environnement. Finalement, la couche supérieure se charge des aspects sociaux de l'environnement, c.-à-d., du raisonnement tenant compte des autres agents. Un agent hybride (Figure 4) est conçu pour allier des capacités réactives et des capacités cognitives ce qui le permet d'adapter son comportement, en temps réel, à l'évolution de l'environnement [77].

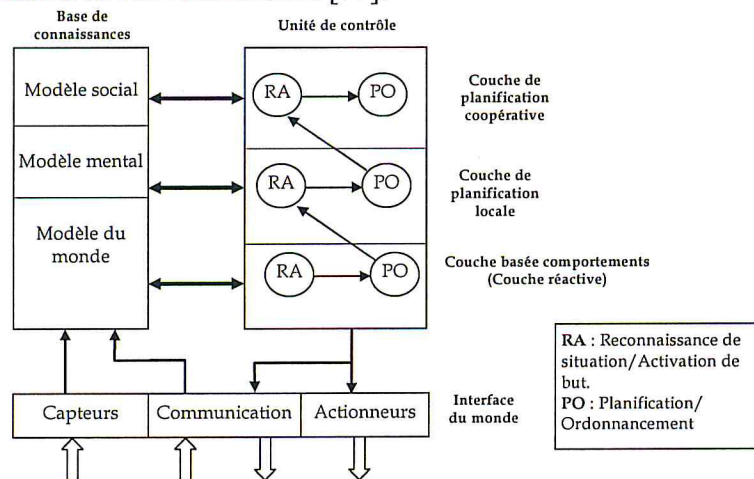


Figure 4: Architecture générale d'un agent hybride

² Désirs : les désirs sont les états possibles envers lesquels l'agent peut vouloir s'engager.

³ Intentions : elles consistent en les états envers lesquels l'agent s'est engagé, et envers lesquels il a engagé des ressources.

3. Systèmes multi-agents (SMA)

La diversité des connaissances et des modes de raisonnement qui collaborent à la résolution d'un problème rendent difficile la réalisation d'un système unique capable de tout résoudre. L'intelligence est, alors, distribuée entre plusieurs modules (ou agents) en interaction. L'efficacité du système dépend beaucoup de sa structure. Les points à considérer sont les suivants :

- *Facilité des communications* : la mémoire, la vitesse de transmission, etc.
- *Disponibilité des informations* : les informations sont réparties ou centralisées.
- *Flexibilité du système* : c'est la facilité d'un système à changer d'objectif, à changer d'organisation, à modifier sa structure, etc.
- *Existence ou non d'un superviseur*.

Les SMA se concentrent sur le comportement collectif d'un ensemble d'agents avec des objectifs, qui peuvent être, contradictoires. De ce fait, le problème est divisé en un ensemble de sous-problèmes. Chacun d'entre eux (objectifs différents) est envoyé à un/des agent(s) différent(s) [78].

3.1. Définition d'un SMA

Étant donné que la définition d'un système est un ensemble organisé d'éléments [78], un SMA peut être défini comme suit :

- Un SMA est un ensemble organisé d'agents. Cette définition signifie que dans un SMA, il existe une ou plusieurs organisations qui structurent les règles de cohabitation et de travail collectif entre agents telles que la définition des différents rôles, le partage de ressources, la dépendance entre opérations, les protocoles de coordination et de résolution de conflits, etc.
- Un SMA est un système distribué composé d'un ensemble d'agents, qui simulent, dans une certaine mesure, les capacités du raisonnement humain. Les SMA sont conçus comme un ensemble d'agent interagissant, le plus souvent, selon des modes de coopération, de concurrence ou de coexistence.
- Un SMA est un système composé d'un ensemble d'opérations à réaliser et un ensemble d'objets associés à l'environnement [68].

Un SMA est, donc, un univers d'agents dans lequel les agents communiquent pour résoudre un problème complexe ou une tâche difficile à résoudre par un seul agent. Les SMA sont des systèmes idéaux pour représenter des problèmes possédants de multiples méthodes de résolution, de multiples perspectives et/ou de multiples solveurs. Ces systèmes possèdent les avantages traditionnels de la résolution distribuée et concurrente de problèmes comme la modularité, la vitesse (avec le parallélisme) et la fiabilité (due à la redondance). Les types d'interaction incluent la *coopération*⁴, la *coordination*⁵ et la *négociation*⁶.

3.2. Intérêts et caractéristiques des SMA

3.2.1. Intérêts des SMA

Les SMA présentent beaucoup d'intérêt justifiant leur utilisation qui sont inspirées des autres disciplines du monde réel. Les principaux intérêts et avantages des SMA sont résumés dans les points suivants [78] :

- *Nature distribuée des problèmes* : les SMA s'adaptent bien à la réalité dans la mesure où de nombreux problèmes sont de nature distribuée.

⁴ *Coopération* : elle consiste à travailler ensemble à la résolution d'un objectif commun.

⁵ *Coordination* : c'est l'organisation de la résolution de sorte que les interactions nuisibles soient évitées et les interactions bénéfiques soient exploitées.

⁶ *Négociation* : la négociation consiste à parvenir à un accord acceptable pour toutes les parties concernées.

- *Tolérance aux fautes* : le fait de répartir les opérations entre un ensemble d'agents, augmente la tolérance aux pannes du système. Le SMA peut réussir sa tâche même si un agent tombe en panne et peut mieux fonctionner dans un environnement dégradé qu'un système mono-agent. De ce fait, l'arrêt de fonctionnement d'un agent n'influe pas sur la résolution du problème.
- *Efficacité* : les agents dans un SMA peuvent fonctionner en parallèle lorsque l'environnement matériel et logiciel le permet. Cette propriété augmente, par conséquent, la rapidité de résolution.
- *Modularité* : la complexité d'un système d'intelligence artificielle croît avec la taille de sa base de connaissances. Partitionner ce système en n agents, réduit sa complexité par un facteur, parfois, plus grand que n . La configuration résultante se trouve plus facile à développer, à tester et à maintenir.
- *Traitement distribué* : la réponse en temps-réel peut être accomplie par la distribution du contrôle et de traitement des informations au sein du groupe. L'utilisation de nombreux agents résolvant le même problème de façon différente produit, généralement, des solutions de meilleure qualité, en termes de complétude et de précision.
- *Comportement social* : plusieurs agents peuvent améliorer l'efficacité de nombreuses tâches, en raison du fait que certaines tâches ne peuvent, tout simplement, pas être effectuées par un seul agent. Les stratégies multi-agents ne permettent pas seulement d'augmenter l'utilité, mais elles permettent aussi de développer un aspect important d'intelligence. Les SMA permettent de résoudre des problèmes de taille et de complexité intractables telle qu'il n'est pas réaliste d'essayer de les résoudre avec un seul agent.
- *Apprentissage évolutif* : plusieurs stratégies peuvent évoluer simultanément car elles sont développées, testées et partagées par tous les agents du système.
- *Comportement coopératif* : les agents peuvent partager des opérations et des informations pour produire un comportement synergique. De plus, les agents peuvent se spécialiser pour accroître l'efficacité. Ils peuvent aussi augmenter et diversifier l'ensemble de tâches qu'ils peuvent accomplir.
- *Scalabilité* : il est très facile d'ajouter de nouveaux agents au SMA qui permet, par conséquent, d'ajouter de nouvelles capacités au système.
- *Complément des informations insuffisantes* : les SMA permettent d'intégrer des connaissances diverses et complexes. De plus, les connaissances insuffisantes peuvent être complétées suivant la résolution.
- *Réutilisation* : un agent d'un SMA peut être réutilisé pour implémenter une partie d'un autre SMA.
- *Agents homogènes et Agents hétérogènes* : plusieurs scénarios sont possibles dans les SMA et peuvent être organisés selon deux dimensions principales (i) l'hétérogénéité des agents et (ii) la quantité des messages échangés entre ces agents. Ces scénarios peuvent varier des *Agents homogènes* et *Agents hétérogènes* [79]. Le scénario le plus simple consiste en un *SMA homogène*. Ici, tous les agents ont la même structure interne, y compris les objectifs, les connaissances et les opérations possibles (compétences). Ils ont également la même procédure de sélection des opérations. La seule différence consiste en les entrées capteurs et les actions prises à tout moment. Dans un *SMA hétérogène*, les agents ont également différentes entrées capteurs et prennent différentes opérations en fonction de ces entrées, mais ils ont des objectifs différents, des ensembles d'actions différents et/ou des connaissances différentes [78].

3.2.2. Caractéristiques des SMA inspirées des autres disciplines

Les caractéristiques des SMA inspirées des autres disciplines telles que l'intelligence artificielle, la sociologie, la programmation orienté objet, etc. sont données par Figure 5 [80] :

- La communication et la mobilité sont inspirées des systèmes répartis.

- La rationalité, l'apprentissage, la proactivité, la coopération et la réactivité sont des caractéristiques inspirées de l'IA.
- Le caractère de l'agent dans son comportement avec les autres agents est inspiré de la psychologie.
- L'organisation des agents est inspirée de la sociologie.
- Les marchés (négociation, contrat, etc.) sont inspirés des théories économiques.
- L'autonomie est inspirée de la théorie de décision.
- La notion de classe, d'inférence et d'héritage dans le cadre de la *POA* (Programmation Orientée Agent) sont inspirées de la *POO* (Programmation orientée objet).
- Les agents ont le contrôle de leurs actions (autonomie) et, de plus, ils peuvent refuser de coopérer.
- Les agents sont réactifs comme les objets, mais ils sont, de plus, proactifs.
- Les agents sont d'habitude persistants et ils ont leur propre *thread* de contrôle.
- L'aspect de l'intelligence individuelle dans les systèmes de l'IA, est constitué d'un seul agent intelligent. Par contre dans les SMA, on retrouve l'aspect social, l'intelligence du comportement social, donc plusieurs agents intelligents existent dans le système.
- Dans les SMA, les structures de synchronisations et de coordination ne sont pas fixées. Ceci oblige, donc, à utiliser des mécanismes dynamiques.
- On ne peut pas supposer que tous les agents sont toujours désireux à coopérer (problème des agents individualistes).

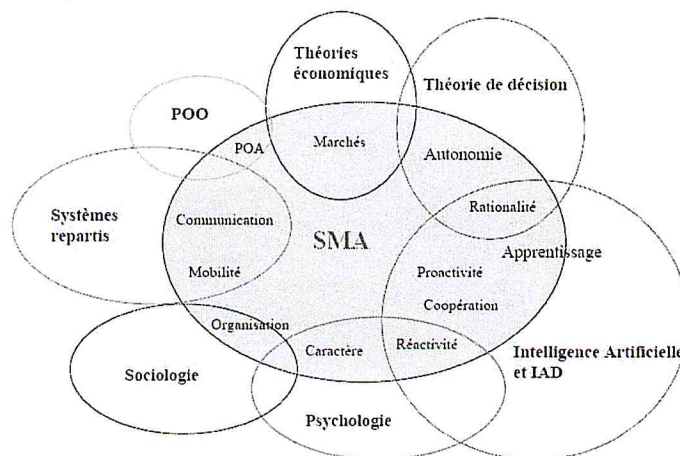


Figure 5: Caractéristiques principales des SMA inspirées des autres disciplines

3.3. Interactions inter-agents

Une des principales propriétés de l'agent dans un SMA est celle d'interagir avec les autres agents du système. Ces interactions sont généralement définies comme toute forme d'action exécutée au sein du SMA ayant pour effet de modifier le comportement d'un autre agent. Elles permettent aux agents de participer à la satisfaction d'un but global. Cette participation permet au système d'évoluer vers un de ses objectifs et d'avoir un comportement intelligent indépendamment du degré de complexité des agents qui le composent.

En général, les interactions sont mises en œuvre par un transfert d'informations entre agents ou entre l'environnement et les agents, soit par *perception*⁷, soit par *communication*⁸.

L'interaction peut être décomposée en trois phases non nécessairement séquentielles [81] :

⁷ *Perception* : par la perception, les agents ont connaissance d'un changement de comportement d'un tiers au travers du milieu.

⁸ *Communication* : par la communication, un agent fait un acte délibéré de transfert d'informations vers un ou plusieurs autres agents.

- La réception d'informations ou la perception d'un changement.
- Le raisonnement sur les autres agents à partir des informations acquises.
- Une émission de(s) message(s) ou plusieurs opérations (plan d'opérations) modifiant l'environnement.

Le degré de complexité des connaissances nécessaires pour traiter les interactions dépend des capacités cognitives (du raisonnement) de l'agent et du fait que l'agent a connaissance ou non de l'objectif du système global. En effet, un agent qui poursuit un objectif individuel au sein du système (c'est le cas des agents réactifs), ne focalise pas son énergie pour interagir avec les autres même s'il y est amené. Par contre, un agent qui participe à la satisfaction du but global du système tout en poursuivant un objectif individuel, va passer une partie de son temps à coopérer ou à se coordonner avec les autres agents. Pour cela, il doit posséder des connaissances sociales qui modélisent ses croyances sur les autres agents du système.

3.4. Communication entre agents

La communication est l'échange intentionnel des informations entre différentes entités du système dû à une perception d'un événement nécessitant cet échange. Les communications dans les SMA, comme chez les humains, sont à la base des interactions et de l'organisation.

Afin de standardiser ces communications, des langages de communication ou d'interaction entre agents ont été établis. L'intérêt de ces langages d'interaction est de réduire les communications en évitant une description exhaustive des messages *ad-hoc* et une gamme inutilement étendue de protocoles. Ces langages se focalisent, essentiellement, sur la manière de décrire exhaustivement des *actes de communication*⁹ d'un point de vue syntaxique et sémantique supportant un langage de représentation des connaissances [80].

3.4.1. Langages de communication entre agents

Plusieurs tentatives de normalisation de la communication inter-agents ont été effectuées au sein de la communauté multi-agents durant ces dernières années. Les principaux langages de communications sont cités dans ce qui suit :

3.4.1.1. Knowledge Query and Manipulation Language (*KQML*)

KQML, développé durant les années 90, est un langage assurant le haut niveau et le bas niveau de communication entre agents. *KQML* est indépendant de la syntaxe des messages, du mécanisme de transport et du langage de codage des messages [82].

Exemple de message *KQML* :

```
(
  tell
  :receiver A
  :sender B
  :ontology et-book
  :language PROLOG
  :content "price(ISBN 973-31-1096-5)"
)
```

3.4.1.2. Agent Communication Language (*FIPA-ACL*)

Récemment, la collaboration internationale des membres d'organisations universitaires et industrielles regroupées au sein de la *FIPA*¹⁰ a défini un certain nombre de spécifications principales

⁹ *Acte de communication* : il est assimilable à une action dans le sens où celui-ci produit des effets sur son destinataire (Hoet & Sabouret, 2009).

¹⁰ *FIPA* : Foundation for Intelligent Physical Agents.

d'agents. Notamment, un standard de langage de communication *Agent ACL* a été proposé et spécifié [83]. Les spécifications de *FIPA-ACL* se composent d'un ensemble de types de messages et de la description de leur pragmatique que sont, des effets sur les attitudes mentales des agents (expéditeur et récepteur).

FIPA-ACL est superficiellement semblable à *KQML*. Sa syntaxe est identique à celle de *KQML* excepté différents noms pour quelques primitives réservées. Le langage externe définit la signification prévue du message. Le langage interne ou le contenu, dénote l'expression à laquelle s'appliquent les croyances, les désirs et les intentions des interlocuteurs.

Exemple de message *FIPA-ACL* :

```
(QUERY-REF
:sender (agent-identifiant
:name WaiterAgent@example.com
:addresses (sequence http://example.com:7780/jade))
:receiver (set ( agent-identifiant
:name ChefAgent@example.com
:addresses (sequence http://example.com:7781/jade)))
:content "<rdf:RDF
xml:base=\http://example.com/WaiterAgent/msg1234\
xmlns:rdf=\http://www.w3.org/1999/02/22-rdf-syntax-ns#\
xmlns:owl=\http://www.w3.org/2002/07/owl#\>"
etc.)
```

3.4.1.3. Knowledge Interchange Format (*KIF*)

Ce langage prend en charge le niveau purement syntaxique (bas niveau) dans une communication entre deux agents. En effet, les messages échangés dans ce langage sont atomique, ou des expressions constituées, elles mêmes, d'atomes [83].

Exemple de message *KIF* :

```
(forall ?x (=> (P ?x) (Q ?x)))
(exists ?person (mother mary ?person))
(=> (apple ?x) (red ?x))
(<<= (father ?x ?y) (and (child ?x ?y) (male ?x)))
```

3.4.2. Protocoles de communication

Un protocole est une stratégie de niveau élevé poursuivi par les agents logiciels qui interagissent avec d'autres agents. Actuellement, il existe plusieurs protocoles de communications :

3.4.2.1. Protocole réseau contractuel (*Contract Net*)

Ce protocole est l'une des premières approches utilisées dans les SMA pour résoudre le problème d'allocation des opérations. Dans ce protocole, qui s'appuie sur une métaphore organisationnelle, les agents coordonnent leurs activités grâce à l'établissement de contrats afin d'atteindre des buts spécifiques. Dans le protocole *Contract Net*, les agents peuvent prendre deux rôles (i) *Gestionnaire* ou (ii) *Contractant*. L'agent qui doit exécuter une tâche, en l'occurrence le *Gestionnaire*, commence par la décomposer en plusieurs opérations et annonce chacune sur un réseau d'agents (*CFP*). Les agents qui reçoivent une annonce des opérations à accomplir évaluent l'annonce. Les agents qui ont les ressources appropriées, l'expertise ou l'information requise pour accomplir ces opérations, envoient au gestionnaire des soumissions qui indiquent leurs capacités à réaliser ces opérations (*Proposal*). Le gestionnaire rassemble toutes les propositions reçues et alloue chaque opération à l'agent qui a fait la meilleure proposition (*Contract*, *AcceptContract* et *RejectContract*). Ensuite, le gestionnaire et les

contractants échangent les informations nécessaires durant l'accomplissement des opérations. Par exemple, le contractant annoncera au gestionnaire quand l'exécution de l'opération sera terminée (Figure 6). Dans des cas exceptionnels, un gestionnaire peut annuler le contrat (*CancelProposal*) en annonçant au contractant qu'il faut abandonner l'exécution de l'opération [84].

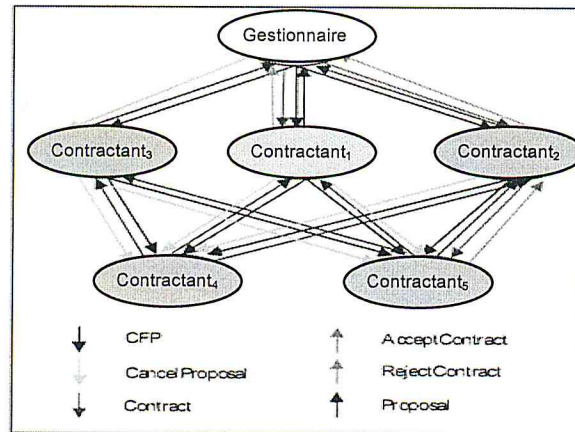


Figure 6: Protocole Contract Net

3.4.2.2. Tableau noir (Blackboard)

En IA, la technique du tableau noir (Blackboard) est très utilisée pour spécifier une mémoire partagée par divers systèmes (Figure 7). Dans un SMA utilisant un tableau noir, les agents peuvent écrire des messages, insérer des résultats partiels de leurs calculs et obtenir de l'information. Le tableau noir est, en générale, partitionné en plusieurs niveaux qui sont spécifiques à l'application. Les agents qui travaillent sur un niveau particulier peuvent accéder aux informations contenues dans le niveau correspondant du tableau noir et dans des niveaux adjacents. Ainsi, les données peuvent être synthétisées à n'importe quel niveau et transférées aux niveaux supérieurs alors que les buts de haut niveau peuvent être filtrés et passés aux niveaux inférieurs pour diriger les agents qui œuvrent à ces niveaux [84].

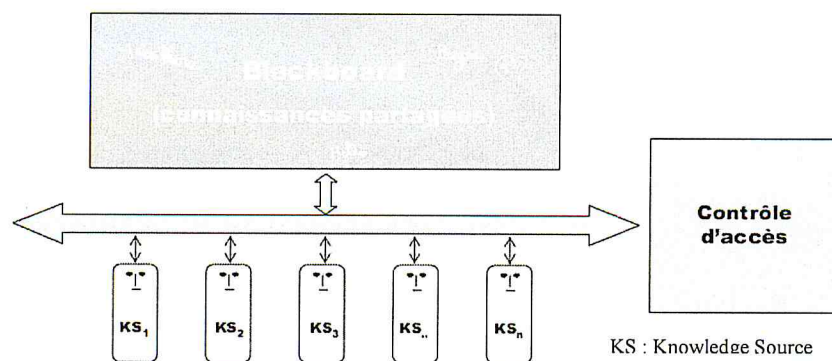


Figure 7: Tableau noir (Blackboard)

3.4.3. Modes et méthodes de communication

Deux modes et méthodes principales pour supporter la communication entre agents dans un SMA sont essentiellement distingués :

- **Mode synchrone** : les agents émetteur et récepteur fonctionnent au même rythme d'une horloge commune réglée au début de la communication. Les agents peuvent accéder à une base de données partagée (blackboard) dans laquelle les informations sont postées (partage de ressources).

- **Mode asynchrone** : dans le mode asynchrone, les données échangées sont émises et analysées selon une référence de temps différente et un rythme variable. Ici, les agents échangent des messages directement (envoi de message).

3.4.4. Coordination entre agents

Une autre problématique des SMA est la relation de l'agent avec les autres agents du même système. En effet, l'individu d'un SMA doit être capable de mettre en œuvre les actions qui répondent au mieux à ses objectifs, mais ceci ne doit, en aucun cas, nuire à la bonne cohésion du SMA. D'une autre manière, les agents d'un même SMA ne doivent jamais entrer en concurrence pour réaliser un même but. La problématique de l'individu et de sa relation au monde couvre aussi la notion d'engagement de l'agent vis-à-vis d'un agent tiers [85].

3.5. Formes de contrôle dans un SMA

Une architecture de contrôle séquentielle utilise, en général, un volume important de connaissances ce qui entraîne une certaine complexité de l'architecture quant à son développement et à sa maintenance. En effet, une décomposition de tâches complexes en plusieurs opérations permet de réduire cette complexité. De ce fait, un système distribué est donc souhaitable. Ainsi, la décomposition des connaissances en plusieurs modules permet, d'une part, de diminuer la complexité des différents niveaux de représentation et, d'autre part, de faciliter le développement, le test et la mise à jour du système. De plus, une architecture séquentielle, contrairement à une architecture distribuée, ne permet pas la distribution du contrôle. Cependant, le contrôle distribué permet des prises de décision interactives, à l'aide de mécanismes de coopération et de négociation, où les différents points de vue d'experts pour un même problème sont échangés. La distribution du contrôle permet, donc, de tenir compte des points de vue divergeant des différents experts [85].

Deux formes opposées de contrôle dans un SMA sont distingués :

- *Contrôle centralisé* : il est établi par un agent du SMA qui supervise l'ensemble des activités des autres agents (*Agent Superviseur*). Ce type de contrôle facilite la cohérence du système, mais, il restreint l'autonomie des agents. Ainsi, la modification, l'ajout ou la suppression d'un agent ne sont pris en compte que par l'agent *Superviseur*, et n'ont pas ou très peu d'influence sur les autres agents. L'inconvénient majeur d'un contrôle centralisé réside dans le fait que si l'agent *Superviseur* est bloqué ou tombé en panne, l'ensemble du système l'est également.
- *Contrôle décentralisé* : les décisions sont prises localement par chaque agent. Ce type de contrôle permet une plus grande fiabilité et extensibilité du système, mais, il engendre des problèmes de cohérence puisque les agents sont autonomes et ont tous le même poids dans le processus de prise de décisions.
- *Contrôle mixte* : En pratique, il peut être intéressant de développer des structures de contrôle mixte permettant des prises de décisions locales dont la cohérence est assurée par une forme quelconque de contrôle centralisé.

4. Conclusion

Cette annexe a porté sur les principes des SMA et des interactions dans la résolution collective des problèmes. Les concepts de base permettant de mettre en place l'échange d'information, d'induire des comportements spécifiques et d'interagir avec les autres agents ont été abordés à travers les langages de communication.

Les SMA héritent bien évidemment des avantages des systèmes à base d'agent tels que la robustesse et la flexibilité. Mais, leur principal intérêt est qu'ils présentent une manière particulière d'aborder les problèmes compliqués à résoudre de façon classique en temps réel.

Annexe B

Publication de l'architecture de télérobotique



1. Introduction

Dans cette dernière annexe, nous allons donner une description des étapes à suivre pour la publication d'une application web dans IIS.

2. Installation des services internet (IIS)

Par défaut, les services Internet ne sont pas installés, disponibles uniquement pour la version Professionnelle. C'est à l'utilisateur de l'installer, à prés installation de Windows XP Pro, procéder au lancement de l'installation de ces services.

Sélectionnez "Démarrer / Panneau de configuration / Ajouter ou supprimer des programmes".
Puis, cliquez sur le bouton "Ajouter ou supprimer des composants Windows" (figure 1).

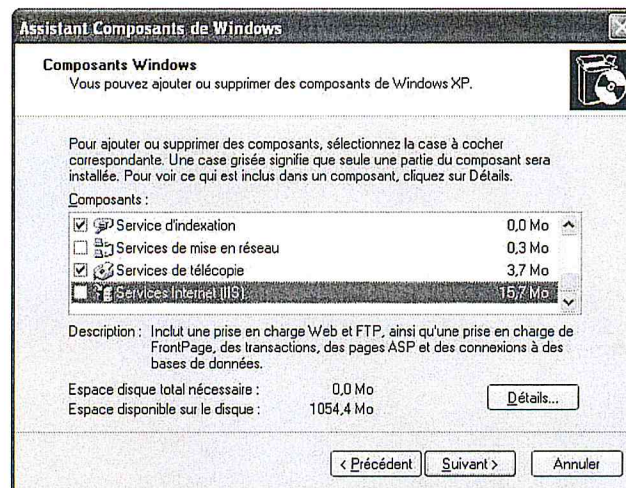


Figure 1: Ajouter des Composants Windows

Cliquez sur 'Détails'. Choisissez un maximum de composants (figure 2).

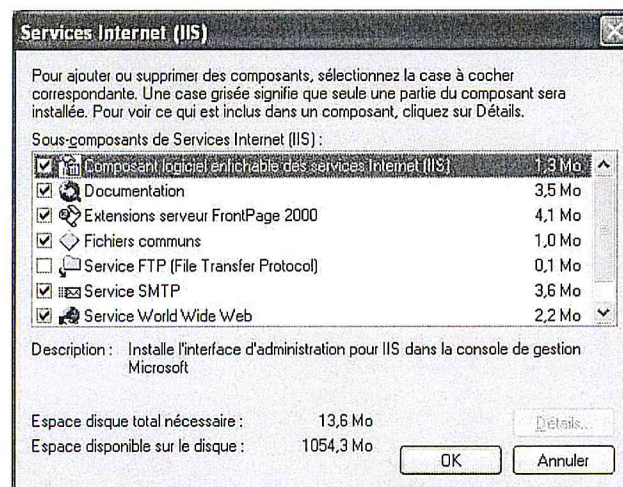


Figure 2: Choix des Composants

Insérez le CD-ROM d'origine de Windows XP (figure 3).

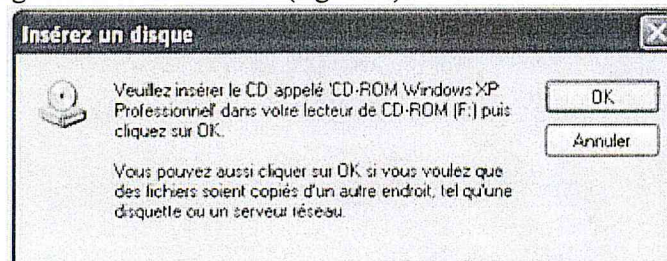


Figure 3: Insérez le CD-ROM Windows XP

Puis cliquez sur 'OK' pour commencer l'installation (figure 4).

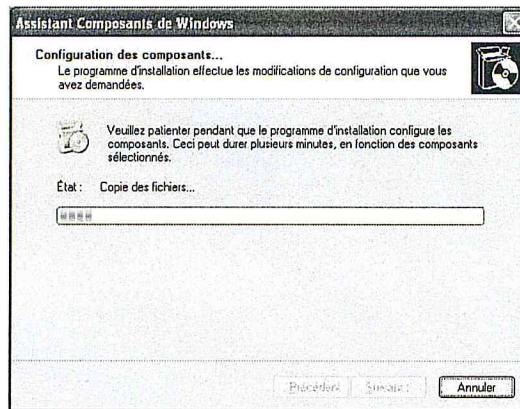


Figure 4: commencer l'installation

L'installation est maintenant terminée.

Pour tester le bon fonctionnement des services Internet en ouvre le navigateur à l'adresse <http://localhost/localstart.asp> (figure 5).

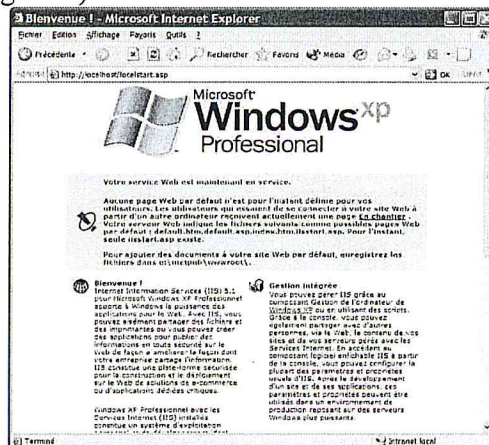


Figure 5: Page de Test

Après avoir installé les outils nécessaires, nous allons voir comment héberger un site web dans IIS. Pour ce faire, aller dans le dossier contenant le site et cliquer sur propriété du dossier (figure 6) :

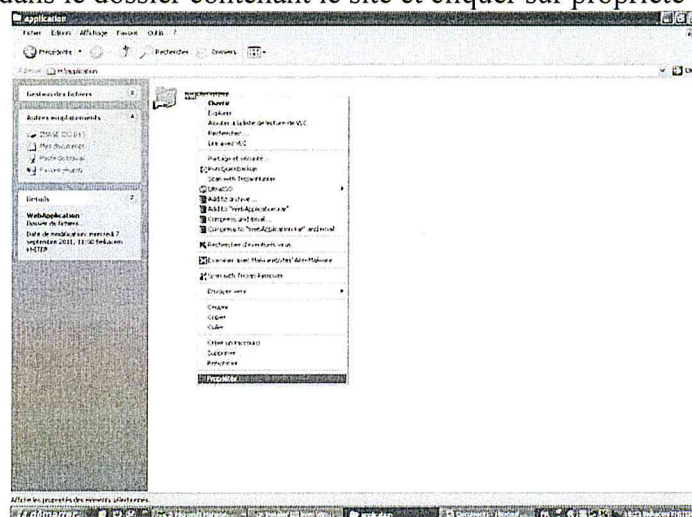


Figure 6: Répertoire au se trouve l'application web

Dans la fenêtre qui apparaît, aller dans l'onglet « Partage Web » tel que montrée par figure 7. Activer le bouton radio « Partager ce dossier » et spécifier l'emplacement de partage « Site web par défaut » (figure 7).

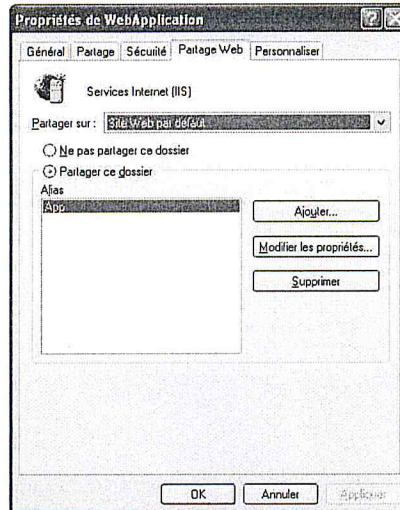


Figure 7: Partage web

Sélectionner l'onglet « sécurité » de la même fenêtre (figure 8),

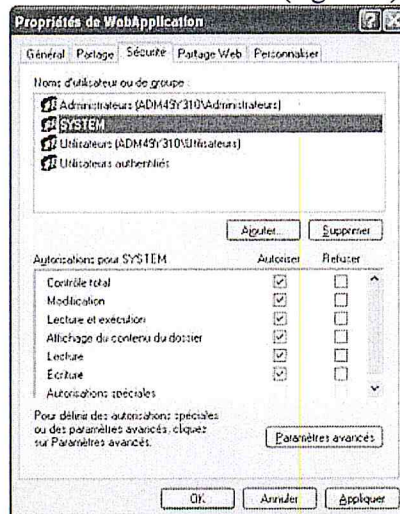


Figure 8: l'onglet sécurité

Ajouter l'utilisateur ASPNET a l'aide du bouton « ajouter » et lui attribuer les droits nécessaire pour l'accès au dossier (figure 9).

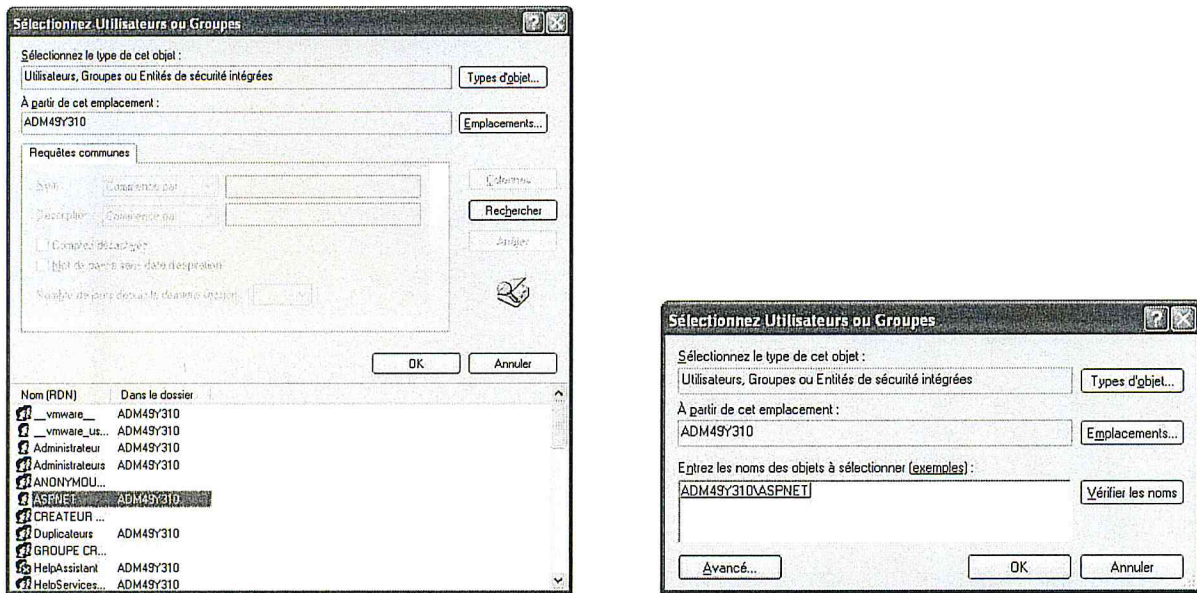


Figure 9: Ajouter l'utilisateur ASPNET

Une fois le dossier contenant le site est partagé, différents paramètres doivent être spécifiés dans iis. Pour cela lancer la console « service internet iis » : Panneau de configuration/Outils d'administration/« service internet iis » (figure 10).

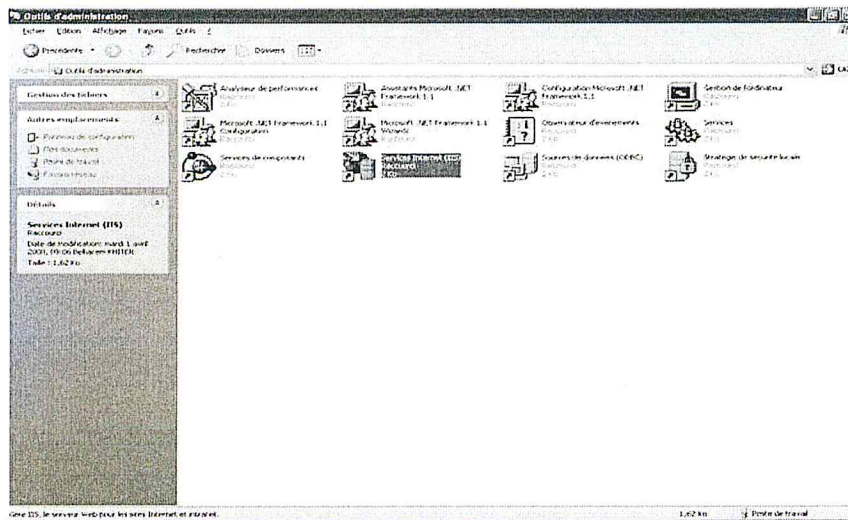


Figure 10: service internet IIS

Dans sites web, cliquer sur propriétés du site web par défaut (figure 11).

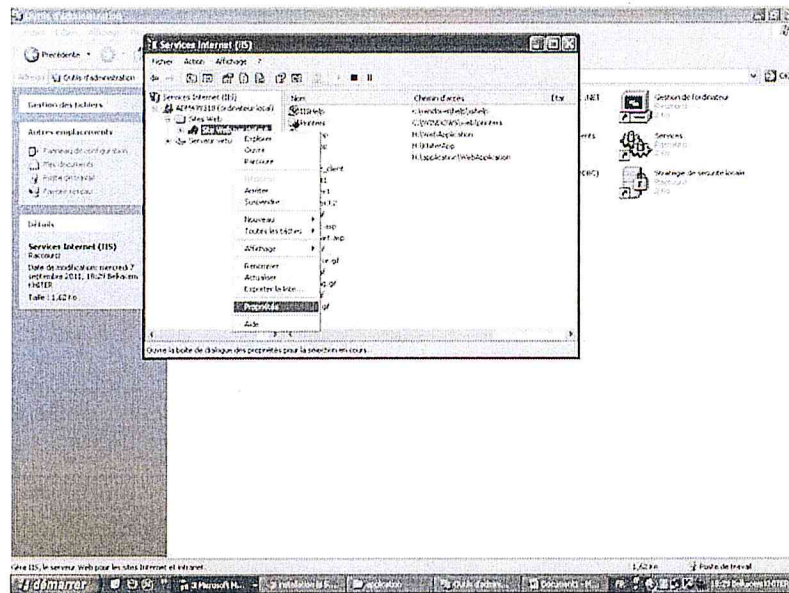


Figure 11: propriétés du site web par défaut

Dans l'onglet « site web », attribuer un numéro de port unique au site (ici par exemple 8080) afin de lui identifier parmi les autres sites hébergés dans le même serveur.

D'autres paramètres peuvent être réglés comme, l'adresse IP d'écoute et le délai de connexion (figure 12).

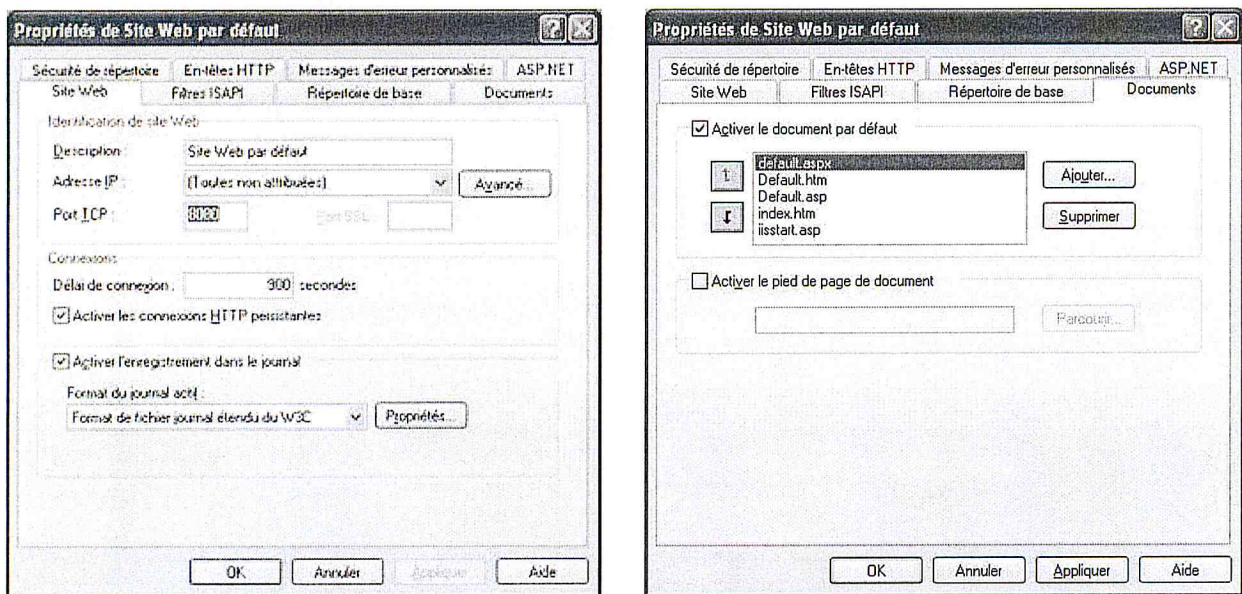


Figure 12: paramètre du site web par défaut

Dans l'onglet documents, spécifier la page web par défaut de votre site (figure 12).

Dans l'onglet « ASP.NET », indiquer la dernière version d'ASP.NET pour prendre en compte toutes les fonctionnalités (figure 13).

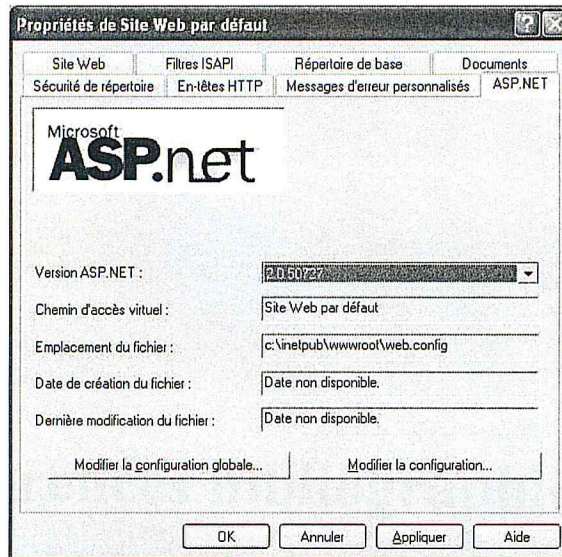
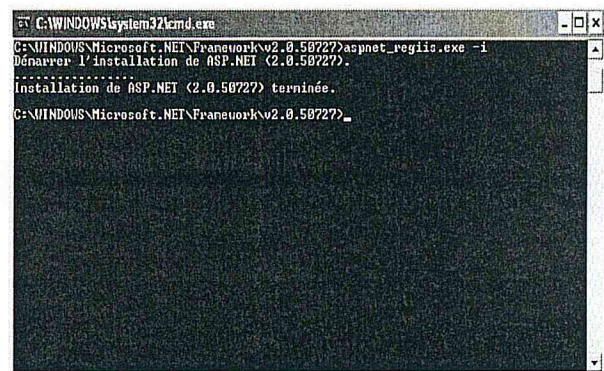
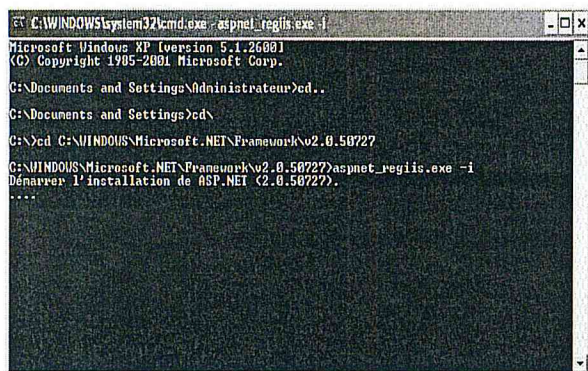


Figure 13: version d'ASP.NET pour prendre en compte

Une autre alternative d'installation de IIS via la ligne de commande est offerte par la commande :
 Aspnet_regiis.exe -i



- [07] Staritz P. J., Skaff S., Urmson C. & Whittaker W., "Skyworker: a robot for assembly, inspection and maintenance of large scale orbital facilities", *The International Conference on Robotics and Automation (ICRA 2001)*, Seoul, Korea, pp. 4180–4185, 21-26 May 2001.
- [08] Laengle T., Hoeniger T., Rembold U. & Woern H., "Cooperation in Human-Robot-Teams". *International Conference on Informatics and Control (ICI&C'97)*, St. Petersburg, Russia, 9-13 June 1997.
- [09] Dombre E., Duchemin G., Poignet P. & Pierrot F., "Dermarob: A Safe Robot for Reconstructive Surgery". *IEEE Transactions on Robotics and Automation*, vol. 19, no. 5, October 2003.
- [10] <http://www.cnrs.fr/Cnrspresse/Archives/n356a1.htm#>
- [11] <http://www.gfmer.ch/>
- [12] <http://www.agrobotics.dk/>
- [13] Graf B., Hans M. & Schraft R. D., "Care-O-bot II—Development of a Next Generation Robotic Home Assistant", *Autonomous Robots*, 16(1), pp. 193–205, 2004.
- [14] Drouillon F., "Chimères et gargouilles informatiques", *Thèse de Doctorat, Université Paris 8 - Vincennes Saint Denis*, 22 octobre 2003.
- [15] <http://www.science-et-vie.net/definition-robot-354-pg4.html>
- [16] Waarsing B. J. W., Nuttin M. & Van Brussel H., "Behaviour-based mobile manipulation: the opening of a door". *The First International Workshop on Advances in Service Robotics (ASER'03)*, Bardolino, Italy, pp. 168-175, 13-15 March 2003.
- [17] www.futura-sciences.com/fr/news/t/robotique/d/el-e-le-robot-qui-apporte-les-objets_15053/
- [18] Rodriguez A. N., "ASSET : une architecture générale pour la télérobotique", *Thèse de Doctorat, Université Paul Sabatier de Toulouse*, France, 2003.
- [19] Ferrell W. R., "Remote manipulation with transmission delay", *IEEE Transactions on Human Factors in Electronics*, v. 6, p. 24-32, september 1965.
- [20] Bejczy A., Kim W., & Venema S., "The phantom robot: Predictive displays for teleoperation with time delay", *IEEE International Conference on Robotics and Automation ICRA '90*, p. 546-551, 1990.
- [21] Sheridan T., "Space teleoperation through time delay: Review and prognosis", *IEEE International Conference on Robotics and Automation ICRA '93*, p. 592-606, 1993.
- [22] BEJCYZ A., "Teleoperation : The language of the human hand", *IEEE International Workshop on Robot and Human Communication, ROMAN'92*, p. 32-43, september, 1-3, Tokyo, Japan, 1992.
- [23] LEDRMAN S., and KLAZKY, R., "The intelligent hand: An experimental approach to human object recognition and implications for robotics and ai", *AI Magazine*, 1994.
- [24] VERTUT J., et COIFFET P., "Les Robots: Téléopération", v. 3A, et 3B, Hermès, 1984.
- [25] COIFFET P., et KHEDDAR A., "Téléopération et télérobotique", *Traité IC2, série Systèmes automatisés*, Hermès, 2002.
- [26] KHEDDAR A., et COIFFET P., "Téléopération et réalité virtuelle", *Traité IC2, série Systèmes automatisés*, Hermès, 2002.
- [27] OTMANE S., "Télétravail Robotisé et Réalité Augmentée : Application à la Téléopération via Internet", *Thèse de Doctorat, Université d'Evry-Val d'Essonne*, Paris, FRANCE, 2000.
- [28] BACKES P. G., TSO K. S., THARP G. K., "Mars Pathfinder Mission Internet-Based Operations Using WITS", *IEEE International Conference on Robotics and Automation ICRA*, 1998.
- [29] GOLDBERG K., MASCHA M., GETNER S., ROTHENBERG N., "Desktop Teleoperation Via the World Wide Web", *IEEE International Conference on Robotics and Automation ICRA*, 1995.
- [30] TAYLOR K., TREVELYAN J., "Australia's Telerobot On The Web", *26th International Symposium On Industrial Robots, Singapore*, 1995.
- [31] Goldberg K., Santarromana J., "The Telegarden", <http://telegarden.aec.at/>, 1996.

- [32] STEIN M., "Painting on the world wide web : the PumaPaint project", *In Proceeding of the IEEE IROS'98 Workshop on Robots on the Web*, octobre 1998.
- [33] Otmane S., Mallem M., Kheddar A., Chavand F., "ARITI : an Augmented Reality Interface for Teleoperation on the Internet", *Advanced Simulation Technologies Conference.*, p. 254-261, April 16-20, Wyndham City Center Hotel, Washington, D.C., USA. 2000.
- [34] SAUCY P., MONDADA F., "KhepOnTheWeb : Open access to a mobile robot on the Internet", *IEEE robotics and automation magazine*, 2000, p. 41-47.
- [35] LE PARC P., OGOR, P., VAREILLE J., MARCE L., "Web based remote control of the mechanical systems", *IEEE International Conference on Software Telecommunications and Computer Networks*, Split, Croatie, 2002.
- [36] FRAISSE P., AGNIEL C., ANDREU D., DE LOS RIOS J. S., "Teleoperations over IP Network : Virtual PUMA Robot", *International Conference on Industrial Technology ICIT'03*, Maribor - Slovénie, décembre 2003.
- [37] GNAEDINGER E., MICHAUT F., LEPAGE F., "Implémentation d'une architecture de Qds spécifique permettant l'adaptation d'une application de contrôle à distance d'un robot mobile", *Quatrième Conférence Internationale sur l'Automatisation Industrielle*, Montréal, Canada, juin 2003.
- [65] B. ÁLVAREZ, A. IBORA, J.A. PASTOR, C. FERNÁNDEZ, A. ALONSO, J.A. DE LA PUENTE, "Software Architecture for Development of Mechatronic Systems: Service Robots", *Dedicated Systems Magazine*, No. 4, pp17-22, 2001.
- [38] Q. STAFFORD-FRASER, "The Trojan Rom Cofee Pot:a (non- technical) Biography", *Document accessible via l'URL: <http://www.cl.cam.ac.uk/coffee/qsf/coffee.html>*, mai 1995.
- [39] K. GOLDBERG, M. MASCHA, S. GENTNER, N. ROTHENBERG, C. SUTTER, J. WIEGLEY, "Zesktop tele-operation via the World Wide Web", *IEEE International Conference on Robotics and Automation*, Japon, mai1995.
- [40] K. TAYLOR, J. TREVELYAN, "Australia's telerobot on the web", *International 26th Symposium on Industrial Robotics*, Singapour, octobre 1995.
- [41] B. Dalton, K. Taylor, "A Framework for Internet Robotics", *IEEE International Conference On Intelligent Robots and Systems (IROS): Workshop on Web Robots*, Canada, octobre 1998.
- [42] M. J. COX, J.E.F. BARUCH, "Robotic Telescopes: An Interactive Exhibit on the World-Wide Web", *2nd International Conference of the World-Wide Web*, USA, octobre 1994.
- [43] U. Nehmzow, A. Buhlmeier, H. Durer, M. Nolte, "Remote Control of mobile robot via Internet", *Department of Computer Science, University of Manchester, Technical Report Series*, UMCS-96-2-3, 1996.
- [44] P.G. Backes, K.S. Tao, G.K. Tharp, "Mars Pathfinder Mission Internet-based Operation using WITS", *IEEE International Conference on Robotics and Automation*, Belgique, mai 1998.
- [45] O. Michel, P. Saucy, F. Mondada, "Khep On The Web: An Experimental Demonstrator in Telerobotics and Virtual Reality", *Proceedings VSMM'97*, Suisse, septembre 1997.
- [46] R. SIMMONS, "Xavier: An Autonomous Mobile Robot on the Web", *IROS'98: Workshop on Web Robots*, Canada, octobre 1998.
- [47] W. BURGARD, A.B. CREMERS, D. FOX, G. LAKEMEYER, D. HÂHNEL, D. SCHULZ, W. STEINER, S. THRUN, "The interactive museum tour-guide robot", *15th National Conference on Artificial Intelligence*, USA, juillet 1998.
- [48] T. Fong, C. Thorpe, C. Baur, "Advanced Interfaces for Vehicle Teleoperation: Collaborative Control, Sensor Fusion Displays, and Web-based Tools, Vehicle Teleoperation Interfaces Workshop", *IEEE International Conference on Robotics and Automation*, USA, avril 2000.
- [49] S. Grange, T. Fong, C. Baur, "Effective Vehicle Teleoperation on the World Wide Web", *IEEE International Conference on Robotics and Automation (ICRA 2000)*, USA, avril 2000.
- [50] A. RASTOGI, P. MILGRAM, "Augmented Telerobotic Control: A visual interface for unstructured environments", *KBS/Robotics Conference*, Canada, octobre, 1995.

- [51] G.V. KONDRASKE, R.A. VOLZ, D.H. JOHNSON, D. TESAR, J.C. TRINKLE, "Network-based Infrastructure for Distributed Remote Operations and Robotics Research", *IEEE Transactions on Robotics and Automation*, Vol. 9, No. 5, pp702-704, octobre 1993.
- [52] J. BALARAM, H. STONE, "Automated Assembly in the JPL Telerobot bed", *Intelligent Robotic Systems for Space Exploration*, Kluwer Academic Publishers, 1992.
- [53] R. Volpe, J. Balaram, "Technology for Robotic Surface Inspection in Space", *AIAA Conference on Intelligent Robots in Field, Factory, Service, and Space (CIRFFSS)*, USA, mars 1994.
- [54] D. DVORAK, R. RASMUSSEN, G. Reeves, A. Sacks, "Software Architecture Themes in JPL's Mission Data System", *IEEE Aerospace Conference*, USA, mars 2000.
- [55] N. MUSCETTOLA, "Remote Agent: To Boldly Go Where No AI System Has Gone Before", *Artificial Intelligence*, No. 103, pp5-48, 1998.
- [56] R. VOLPE, I. NESNAS, T. ESTLIN, D. MUTZ, R. PETRAS, H. DAS, "The CLARAty Architecture for Robotic Autonomy", *2001 IEEE Aerospace Conference*, USA, Mars 2001.
- [57] L. Piguet, T. Fong, B. Hine, P. Hontalas, E. Nygren, "VEVI: a Virtual Reality Tool for Robotic Planetary Exploration", *Virtual Reality World Conference*, Allemagne, février 1995.
- [58] B. HINE, C. STOKER, M. SIMS, ET. AL, "The Application of Telepresence and Virtual Reality to Subsea Exploration", *2nd Workshop on Mobile Robots for Subsea Environments (ROV'94)*, USA, mai 1994.
- [59] J. BARES, D. WETTERGREEN, "Lessons from the Development and Deployment of Dante II", *1997 Field and Service Robotics Conference*, Australia, December 1997.
- [60] L. Nguyen, M. Bualat, L. Edwards, L. Flueckiger, C. Neveu, K. Schwehr, M.D. Wagner, E. Zbinden, "Virtual Reality Interfaces for Visualization and Control of Remote Vehicles", *Vehicle Teleoperation Interfaces Workshop, IEEE International Conference on Robotics and Automation*, USA, avril 2000.
- [61] D. KABER, R. ZHOU, D. SONG, «Design and Prototyping of an Economical Teleoperations "bed" for Human factors Research: cost, resource requirements and capability assessment", *25th International Conference on Computers & Industrial Engineering*, USA, mars 1999.
- [62] E. SAHIN, P. GAUDIANO, "KITE : The Khepera Integrated Environment", *First International Khepera Workshop*, Allemagne, décembre 1999.
- [63] M. Finke, J. Strassner, J. Speier, L. Peters, M. Pauly, M. Göbel, H. Surmann, "An Interactive Environment for Autonomous Robots", *Topical Workshop on Virtual Reality and Advanced Human-Robot Systems, 15th World Congress of the International Measurement Confederation*, Japon, juin 1999.
- [64] L. FLÜCKIGER, "Interface pour le Pilotage et l'analyse des Robots basée sur un Générateur de Cinématiques", *Thèse de doctorat*, Ecole Polytechnique Fédérale de Lausanne, 1998.
- [66] Hentout A., Bouzouia B. & Toukal Z., "Multi Agent Generic Model Architecture for Piloting Mobile Manipulator Robots". *The International Computer System and Information Technology Conference (ICSIT'05)*, Algiers, Algeria, 19-21 July 2005.
- [67] Guerineau N., Moignard C. & Pomiers P., "RobuTER et Bras Ultraléger : Manuel d'utilisation et de maintenance", *Version 1.0, Robosoft*, France, 1999.
- [68] Ferber J., "Les systèmes multi-agents : vers une Intelligence collective". *Inter-Edition*, Paris, France, 1995.
- [69] Demazeau Y. & Müller J., "Decentralized IA 2". *The Second European Workshop on Modelling Autonomous Agents and Multi-agent World (MAAMAW-90)*, Amsterdam, The Netherlands, 2001.
- [70] Jennings N. R. & Wooldridge M., "Agent-Oriented Software Engineering", *In Handbook of Technology*, Edited by J. Bradshaw, AAAI/MIT Press, 2000.
- [71] Sabas A., "Systèmes Multi-agents : Une analyse comparative des méthodologies de développement, Vers la convergence des méthodologies de développement et la standardisation des plateformes SMA". *Mémoire de Maîtrise en mathématiques et informatique appliquées*, Université du Québec à Trois-Rivières, Canada, Décembre 2001.

- [72] Goût J., "Intégration de la Planification Temporelle dans l'Architecture Décisionnelle d'un Robot Autonome". *Thèse de doctorat*, Université de Paul Sabatier de Toulouse, France, 1999.
- [73] Chaib-draa B., Jarras I. & Moulin B., "Système multi-agents : principes généraux et applications". *Agent et systèmes multi-agents*, J. P. Briot et Y. Demazeau, Hermès, 2001.
- [74] Toukal Z., "Contrôle/Commande distribué d'une organisation d'agents robots". *Thèse de doctorat*, Université Paris XII, Spécialité Robotique, France, Février 2000.
- [75] Müller J.-P. "Modélisation organisationnelle en systèmes multi-agents". *Septième École d'été de l'ARCo*, Université de Neuchâtel, Bonas, Switzerland, 10-21 juillet 2000.
- [76] Osherenko A. "Plan Representation and Plan Execution in Multi-agent Systems for Robot Control". *The KI-2001 Workshop Joint German/Austrian Conference on Artificial Intelligence in Planning, Scheduling, Configuration and Design/15. Workshop Planen, Scheduling und Konfigurieren*, Entwerfen, Wien, Austria, 18 September 2001.
- [77] Fischer K., Chaib-draa B., Muller J. P., Pischel M. & Gerber Ch., "A Simulation Approach Based on Negotiation and Cooperation Between Agents: A Case Study". *IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews*, 29(4), November 1999.
- [78] Nebot R., "Agent-based Architecture for Multirobot Cooperative Tasks: Design and Applications", *Ph. D. Thesis*, University Jaume I, Spain, 2007.
- [79] Stone P., Veloso M., "A Survey of Multi-agent and Multi-robot Systems", *Robot Teams: From Diversity to Polymorphism*, A K Peters, Ltd., pp. 37–92, 2002.
- [80] Florea A. M., "Introduction aux agents et systèmes multi-agents". Politechnica University of Bucharest, <http://turing.cs.pub.ro/auf2/html/chapters/>, 2002.
- [81] Chaib-draa B., "Interaction between agents in routine, familiar and unfamiliar situations". *International Journal of Intelligent and cooperative Information systems*, 1996.
- [82] Labrou Y., Finin T., "A Proposal for a new KQML Specification". *Rapport Technique TR CS-97-03*, Baltimore, USA, 1997.
- [83] <http://www.fipa.org/>
- [84] Davis R., Smith R. G., "Negotiation as a Metaphor for Distributed Problem Solving", *Artificial Intelligence*, 20(1), pp. 63–100, 1983.
- [85] Yattara M., "Systèmes multi agent et systèmes distributes", Avril 2011, http://kalanko.org/affichage_article.php?id_article=3.

