Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mohamed Khider Biskra

Faculté des Sciences et des Sciences de l'ingénieur

Département d'Informatique

N°d'ordre :……
Série :……

Mémoire

Présenté en vue de l'obtention du diplôme de Magister en Informatique

Option**: Intelligence Artificielle**

:

Une approche formelle pour l'adaptabilité d'un agent

Par :

M^{elle} Belounnar Saliha

Soutenu le : 13 /03 / 2011

Devant le jury :

Mr. Bouhadada Tahar	**M.C. Université d'Annaba**	**Président**
Mr. Kazar Okba	**M.C. Université de Biskra**	**Rapporteur**
Mr. Khadir Tarek	**M.C. Université d'Annaba**	**Examinateur**
Mr. Farah Nadir	**M.C. Université d'Annaba**	**Examinateur**

Résumé

Un agent agit dans un environnement qui est dynamique, ainsi pour que l'agent assure une intervention rationnelle dans la résolution du problème, il est important qu'il adapte ses connaissances et son comportement. Ainsi l'adaptation lui assure une coordination des opérations avec ses accointant.

L'adaptation de l'agent peut être vue par la satisfaction de ses tendances fondamentales. Pour cela, il modifie son comportement ou sa structure de manière à corréler ses tendances et son environnement.

Dans notre travail, nous avons proposé une architecture méta niveau; afin de développer une structure décisionnelle ou un méta comportement adaptatif est modélisé formellement par les Réseaux de Pétri.

Mots clés :

Adaptabilité, spécification formelle, système multi agent, RdP.

ملخص

إن العون يعمل في محيط يتسم بأنه ديناميكي، و كي نضمن التدخل العقلاني للعون في حل المشاكل يجب أن يكيف معارفه و سلوكه. حيث أن التكيف يضمن له أيضا التنسيق مع الآخرين.

و يمكن أن ينظر إلى تكيف العون على أنه تلبية ميولا ته الأساسية. و لهذا فهو يغير سلوكه أو بنيته بحيث يربط ميولاته و محيطه.

من جهتنا فإننا نقترح هندسة فوقية المستوى بغية تطوير بنية تقريرية أو سلوك فوقي تكيفي ممثل رياضيا بشبكات بيتري.

كلمات مفاتيح:

تكيف، تمثيل رياضي، النضم متعددة الأعوان، شبكات بيتري.

Table des matières

Liste des figures

Liste des Tables

Introduction Générale

Les techniques d'intelligence artificielle distribuées ont été appliquées, avec succès, pour résoudre des problèmes de contrôle ou à la simulation de processus dynamique complexes tels que les écosystèmes, la robotique, la surveillance de patients en soins intensifs ou l'évolution économique. Elles sont souvent adaptées à la modélisation des systèmes comportant plusieurs entités qui interagissent et évoluent dans un environnement dynamique.

Pour aborder des problèmes complexes, un effort particulier a été porté ces dernières années sur les Systèmes Multi-Agents (SMA). Les systèmes multi agents s'appuient sur la métaphore de l'organisation collective. Ils proposent de modéliser les systèmes complexes à l'aide d'une société d'entités appelées agents. Chaque agent a ses propres compétences, mais il a besoin d'interagir avec les autres pour résoudre les problèmes de son domaine d'expertise et d'éviter les conflits.

Les SMA ont déjà fait leurs preuves comme outil puissant de modélisation de différents systèmes, la complexité de ces systèmes s'est trouvées considérablement accrue du fait de : leur distribution, la grande quantité d'informations qu'ils manipulent, leur aspect coopératif et adaptatif, leur ouverture ainsi que leur incertitude. La modélisation de ces systèmes complexes nécessite donc des agents adaptatifs. Les agents doivent en effet être capables de réagir aux changements et perturbations de leur environnement qui est complexe et dynamique. L'adaptation de comportement ou de la structure interne des agents permet ainsi d'acquérir dynamiquement des connaissances est obligatoire afin d'être apte à assurer sa survie dans des conditions nouvelles.

Deux niveaux d'adaptation peuvent être définis. Premièrement, l'adaptation au niveau du système peut être associée à l'auto-organisation. L'environnement agissant sur l'organisation d'un système, tout changement de l'environnement peut modifier l'organisation de ce système. Deuxièmement, l'adaptation existe au niveau de l'entité qui est l'agent. L'adaptabilité d'un agent peut être associée à sa capacité d'ajuster et réagir face aux variations des contraintes de l'environnement et des besoins des utilisateurs ; afin qu'il

soit capable d'atteindre ses objectifs malgré des changements de l'environnement. On parle alors d'agent adaptatif.

Un agent adaptatif est un système qui cherche la satisfaction de ses tendances fondamentales. Pour cela, il adapte et modifie son comportement ou sa structure de manière à corréler ses tendances et son environnement.

L'adaptation peut être :

- structurelle : il s'agit d'adapter la structure de l'agent à l'évolution de son environnement. La réorganisation de la liste des accointances d'un agent à l'arrivée de nouveaux agents dans le système est un exemple de cette adaptation,
- comportementale : il s'agit d'adapter le processus de décision de l'agent à l'évolution de son environnement. Adapter par exemple, le choix des actions à la situation du marché tel est le cas des firmes.

L'adaptation comportementale peut être statique ou dynamique.

L'adaptation dynamique est considérée comme étant la capacité de l'agent à s'adapter aux changements imprévus de son environnement par son évolution continuelle sans l'interruption de son exécution. Ce type d'adaptation nécessite des capacités d'apprentissage.

L'adaptation est statique quand on dote les agents de règles de comportements obtenues par la simulation de l'évolution de l'environnement et des réactions de l'agent à cet environnement.

Ces règles sont figées et nécessitent la prévision des changements possibles de l'environnement au moment de la conception de l'agent.

Notre thème s'inscrit dans ce cadre là, où on va essayer de proposer une approche formelle pour l'adaptabilité statique d'un agent. Et afin de situer et illustrer notre sujet, nous allons découvrir un exemple possible d'application.

Ce mémoire est organisé en quatre chapitres répartis comme suit :

Dans le premier chapitre, nous présentons un état de l'art sur les systèmes multi-agents et les agents adaptatifs. Notre propos dans ce chapitre n'est pas d'établir un état de l'art complet du domaine agent et multi agent, mais plutôt d'introduire ces domaines tout en donnant certaines définitions qui seront utilisées par la suite. Ensuite nous abordons la notion de l'adaptation, qui devient un moyen pour parvenir à surmonter les perturbations

éventuelles de l'environnement et pour réaliser des systèmes adaptatifs. Finalement nous discutons les différents travaux existant, qui sont conçu pour permettre aux agents de s'adapter statiquement et dynamiquement à leur environnement d'exécution.

Le deuxième chapitre est consacré à la présentation des approches de spécification formelles pour les systèmes multi agents. Nous détaillons, par la suite, les principaux concepts des RdPs ; qui sont considéré comme des outils prometteurs pour décrire et étudier les systèmes qui se caractérisent comme étant concurrent, asynchrone, distribué, parallèle, non déterministe, et stochastiques.

Dans le troisième chapitre, nous allons décrire notre contribution qui est la proposition d'une approche formelle pour l'adaptabilité d'un agent.

Dans le quatrième chapitre, nous poursuivons avec une étude de cas afin de valider notre proposition. L'illustration du sujet par une situation concrète va nous permettre de mieux comprendre l'architecture proposée. On a choisi comme cas à étudier le problème de fixation de prix dans les systèmes économiques. Nous précisons ensuite la plate-forme JADE et comment les différents composants de notre architecture peuvent être implémentés en utilisant cette plate-forme.

Enfin, nous terminons ce mémoire par une conclusion générale, qui récapitule les travaux réalisés et fait le point sur un ensemble de perspectives envisagées.

Chapitre I

Agent Et Adaptabilité

I.1 Introduction

Les systèmes multi-agents sont issus de l'intelligence artificielle distribuée (IAD), une branche de l'intelligence artificielle qui s'articule autour de trois axes :

- La résolution distribuée des problèmes qui s'intéresse à la manière de diviser un problème en un ensemble d'entités distribuées et coopérantes et à la manière de partager la connaissance du problème afin d'en obtenir la solution.
- L'intelligence artificielle parallèle qui développe des langages et des algorithmes parallèles pour l'intelligence artificielle (IA) visant ainsi l'amélioration des performances des systèmes d'IA.
- Les systèmes multi-agents qui privilégient une approche décentralisée de la modélisation et mettent l'accent sur les aspects collectifs des systèmes.

Notre propos dans ce chapitre n'est pas d'établir un état de l'art complet du domaine agent et multi agent (pour cela on pourra se référer par exemple à [Ferber 95], mais plutôt d'introduire ces domaines tout en donnant certaines définitions qui seront utilisées par la suite.

Les systèmes multi-agents, comme leur nom l'indique, sont des systèmes constitués de plusieurs agents. Il est donc nécessaire de commencer par définir ce que l'on appelle un agent. Une fois, la notion d'agent définie, nous introduirons le concept de système multi-agents. Ensuite nous abordons la notion de l'adaptation, qui devient un moyen pour parvenir à surmonter les perturbations éventuelles de l'environnement et pour réaliser des systèmes adaptatifs.

I.2 Les Agents et Système Multi Agent

I.2.1 Agent

I.2.1.1 Définitions

Il n'existe pas de définition adoptée par la communauté scientifique sur le terme "agent". En revanche tout le monde s'accorde à dire que la notion d'autonomie est au centre de la problématique des agents.

Définition de Erceau

<Les agents sont des entités physiques (capteurs, processeurs,...) ou abstraites (tâches à réaliser, déplacements,...), qui sont capables d'agir sur leur environnement et sur elles-mêmes, c'est-à-dire de modifier leur propre comportement. Elles disposent, pour ce faire, d'une représentation partielle de cet environnement et de moyens de perception et de communication> [Erceau 93].

Définition de Ferber

<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-agent, peut communiquer avec d'autres agents, et dont le comportement est la conséquence de ses observations, de ses connaissances et des interactions avec les autres agents>[Ferber 95].

Définition de Jennings, Sycara et Wooldridge

<Un agent est un système informatique, situé dans un environnement, et qui agit d'une façon autonome et flexible pour atteindre les objectifs pour lesquels il a été conçu>. Jennings, Sycara et Wooldridge [Wooldridge et al 98] (Figure I.1):

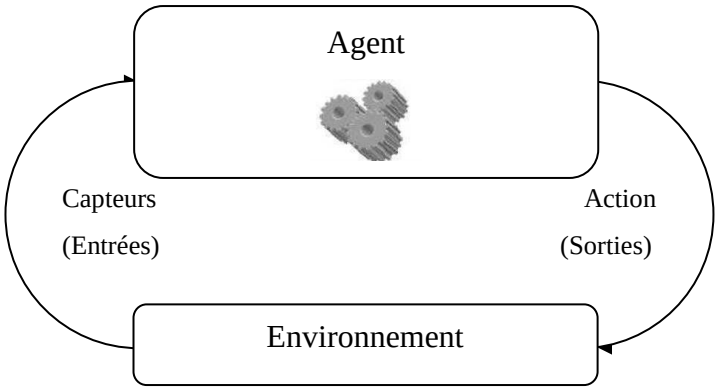


Figure. I.1 Agent et son environnement [Wooldridge et al 98]

I.2.1.2 Agents cognitifs vs réactifs

La granularité des agents impliqués dans une application varie selon deux écoles coexistantes aujourd'hui : l'école cognitive et l'école réactive. Suivant le type d'agent utilisé, on parlera de systèmes cognitifs ou de systèmes réactifs.

- **Agents réactifs**

Les raisonnements réflexes ou réactifs caractérisent une activité de l'agent dans laquelle les actions sont exécutées rapidement. Un agent de ce type évolue parmi un nombre important de ses semblables et c'est le résultat des interactions entre leurs activités qui donne une impression de comportement global 'intelligent' [Mandiau et al 02].

Les agents réactifs sont de plus bas niveau, ils ne disposent que d'un protocole et d'un langage de communication réduit, leurs capacités répondent uniquement à la loi stimulus/action [Labidi et al 93].

Ils ne possèdent pas de représentations de leur environnement. De ce fait toutes les informations relatives à leur comportement se trouvent dans l'environnement et leurs réactions dépendent uniquement de la perception qu'ils peuvent en avoir. Ils opèrent dans l'ici et le maintenant, c'est à dire qu'ils ne mémorisent pas les événements passés et ne peuvent pas non plus anticiper sur le futur **[**Soltani 07].

- **Agents cognitifs**

Les systèmes d'agents cognitifs sont fondés sur la coopération d'agents capables à eux seuls d'effectuer des opérations complexes. Un système cognitif comprend un petit nombre d'agents qui disposent d'une capacité de raisonnement sur une base de

connaissances, d'une aptitude à traiter des informations diverses liées au domaine d'application, et d'informations relatives à la gestion des interactions avec les autres agents et l'environnement [Labidi et al93].

Un agent cognitif se distingue par une représentation explicite de l'environnement, possède des croyances sur les autres, des intentions (c'est à dire des buts et des plans explicites lui permettant d'accomplir leurs buts) et des compétences. Les tâches demandées à des agents cognitifs relèvent, a priori, d'un raisonnement plus compliqué que celui des agents réactifs [Mandiau et al 02].

Le tableau suivant résume les différences entre les modèles cognitifs et les modèles réactifs.

Agent Cognitif	Agent Réactif
Représentation explicite de l'environnement	**Pas de représentation explicite**
Peut tenir compte de son passé	**Pas de mémoire de son historique**
Agents complexes	**Fonctionnement stimulus/réponse**
Système composé de petit nombre d'agents	**Système composé de grand nombre d'agents**

Tableau I.1. Différences entre agents cognitifs et agents réactifs [Mandiau et al 02]

I.2.1.3 Architecture et comportement

Un agent se caractérise essentiellement par la manière dont il est conçu et par ses actions. En d'autres termes par son *architecture* et son *comportement* [Ferber 95].

Le *comportement* apparaît comme une spécification externe de l'agent, et *l'architecture* définissant les relations internes permettant d'aboutir à cette spécification. Inversement, l'observateur perçoit un comportement à partir duquel il peut induire ou spécifier ce qu'une architecture est censé produire. Cette approche conduit à la réalisation de modèles comportement aux indépendant de l'architecture **[**Mandiau et al 02].

- **Architecture**

La figure (Figure. I.1) donne l'aperçu externe d'un agent, dans lequel l'agent est vu en tant que 'boite noire', dont seules les entrées et sorties sont représentées. L'origine (pour une entrée) ou le résultat (pour une sortie) peut, indépendamment de l'agent lui-même, être lié à des dispositifs physiques tels que des capteurs et actionneurs **[**Mandiau et al 02].

L'architecture correspond à un point de vue de concepteur, qui décrit l'organisation interne d'un agent. C'est à dire le principe d'organisation qui sous-tend l'agencement de ses différents composants. Il existe un grand nombre d'architecture envisageable. Certaines s'avèrent plus efficaces en temps de calcul, d'autres sont plus souples et permettent de coder un vaste éventail d'actions, d'autre encore présentent l'avantage de la simplicité [Ferber 95].

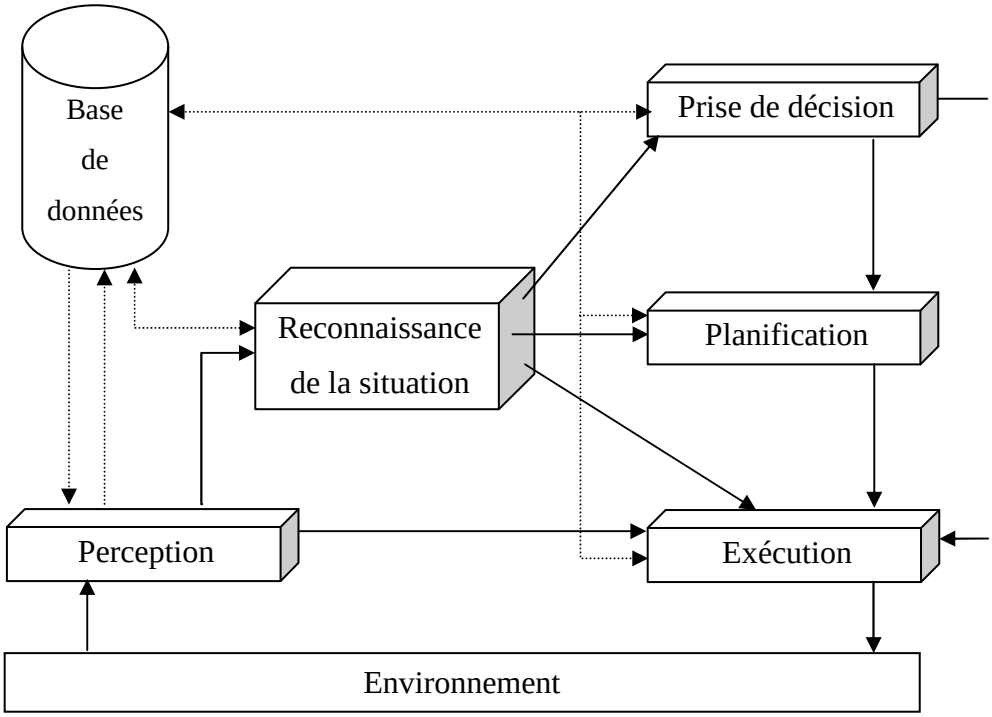


Figure. I.2 Architecture d'agent [Sabas 01]

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 terme d'action et donc, passe 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) [Sabas 01].

- **Comportement**

Le terme comportement est central dans la définition et la modélisation d'un agent. Le comportement est analysable sans connaître les détails d'implémentation. Il s'agit d'un phénomène qui peut être appréhendé par un observateur extérieur qui au regard des actions qu'entreprend l'agent, décrit la relation qui existe entre l'agent, son environnement et les autres agents.

Le comportement caractérise ainsi l'ensemble des propriétés que l'agent manifeste dans son environnement, on peut le comprendre en regardant sa manière d'évoluer et de répondre aux sollicitations de son environnement [Mandiau et al 02].

I.2.1.4 Caractéristiques des agents

Le terme agent est largement utilisé et on peut trouver une multitude de définitions. De ces définitions, qui apparaissent dans la littérature, comme par exemple [Pavon 06] et [Marjorie 02], nous pouvons identifier plusieurs propriétés qui caractérisent les agents :

• *Autonomie :* les agents contrôlent leurs actions et leurs états internes. Le système dans son ensemble est capable de réagir sans l'intervention d'un humain ou d'un autre agent. Il n'y a pas de définition unique du terme agent, par contre, il y a un consensus général pour considérer l'autonomie comme notion centrale de l'agent.

• *Réactivité :* ils perçoivent leur environnement et réagissent aux changements qui s'y produisent dans le temps requis ;

• *Initiative* : le comportement des agents est déterminé par les buts qu'ils poursuivent et par conséquent ils peuvent produire des actions qui ne sont pas seulement des réponses à leur environnement.

• *Habilité sociale* : pour satisfaire ses buts un agent peut demander la collaboration d'autres agents à qui il délègue ou avec lesquels il partage la réalisation de tâches. Pour cela il a besoin de se coordonner, négocier, en définitive, interagir.

• *Raisonnement* : un agent peut décider quel but poursuivre ou à quel événement réagir, comment agir pour accomplir un but, ou suspendre ou abandonner un but pour se dédier à un autre.

• *Apprentissage* : l'agent peut s'adapter progressivement à des changements dans des environnements dynamiques grâce à des techniques d'apprentissage.

• *Mobilité*: dans des applications déterminées il peut être intéressant de permettre aux agents de migrer d'un nœud à un autre dans un réseau tout en préservant leur état lors de sauts entre nœuds**.**

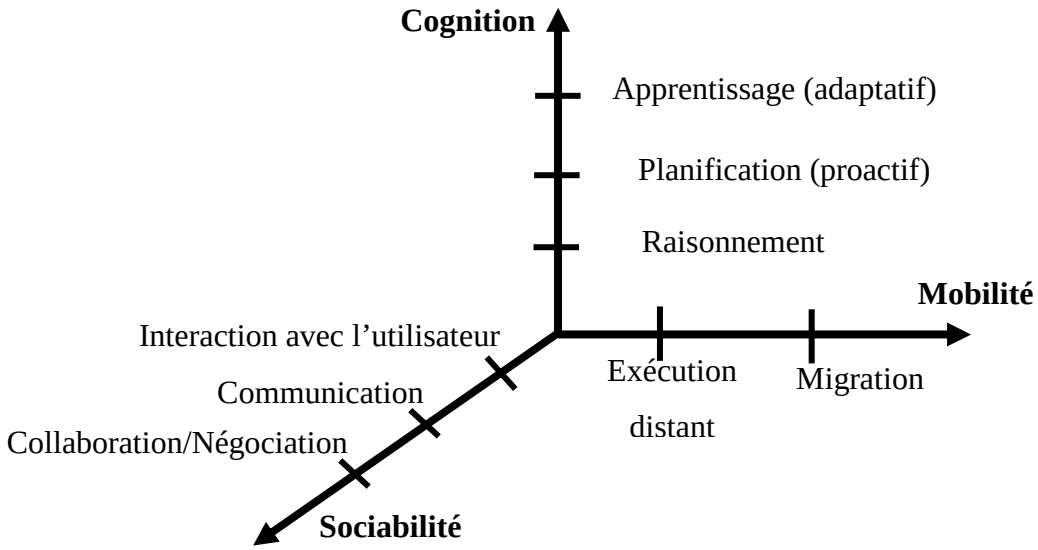


Figure. I.3 Caractéristiques des agents [Pavon 06]

I.2.2 Les systèmes multi agents

A la définition d'un agent, nous joignons naturellement celle d'un système multi-agents.

I.2.2 .1 Définition

On appelle Système Multi-Agent (SMA), un système composé des éléments suivants:

1. *Un environnement E*: c'est-à-dire un espace disposant généralement d'une métrique.

2. *Un ensemble d'objets O*: Ces objets sont situés, c'est-à-dire que, pour tout *objet*, il est possible, à un moment donné, d'associer une position dans *E*. Ces *objets* sont passifs, c'est-à-dire qu'ils peuvent être *perçus*, *créés*, *détruits* et *modifiés* par les agents.

3. *Un ensemble A d'agents*: qui sont des objets particuliers, lesquels représentent les entités actives du système.

4. *Un ensemble de relations R*: qui unissent des objets (et donc des agents) entre eux.

5. *Un ensemble d'opérations Op*: permettant aux agents de *A* de *percevoir*, *produire*, *consommer*, *transformer* et *manipuler* des objets de *O*.

6. *Des opérateurs* : chargés de représenter l'application de ces opérations et la réaction du monde à cette tentative de modification, que l'on appellera *les lois de l'univers*. [Ferber 95]

I.2.2.2 Domaines d'application des systèmes Multi-agents

Les systèmes multi-agents peuvent être appliqués à divers domaines [Wooldridje et Jennings 95] :

- Applications Industrielles : par exemple la fabrication industrielle (Manufacturing), processus de contrôle industriel, contrôle du trafic aérien.

- ❖ Applications commerciales : tel que le commerce électronique, Affaires.
- ❖ Applications médicales : diagnostic médical, contrôle des patients.
- ❖ Enseignement assisté par ordinateur.
- ❖ Recherche d'informations.
- ❖ Conception et développement des logiciels.
- ❖ Les jeux et les loisirs.

I.3 Adaptabilité

La notion intuitive d'adaptabilité précise qu'un système qui a cette propriété se comporte de façon adéquate face aux sollicitations de son environnement, et l'on entend par système un objet construit rationnellement, dans le but de servir les objectifs de ses utilisateurs [Abdessemed 06].

I.3.1 Définitions

La définition sémantique du terme est :

Adapter v. : rendre (un dispositif, des mesures, etc.) apte à assurer ses fonctions dans des conditions particulières ou nouvelles [Hach].

Adaptabilité n. f. Qualité d'un objet qui peut être modifié aisément en harmonie avec les changements auxquels son utilisation est soumise ou peut être soumise [Hach].

Notons que ces deux termes, 'adaptation' et 'adaptabilité' sont souvent confondus dans la littérature [Ledoux 01].

- L'adaptation est la modification à court ou long terme du comportement, de la morphologie et/ou de la physiologie d'un être vivant, afin d'optimiser sa survie et/ou celle de son espèce dans son environnement.
- L'adaptation est une modification (de son organisation, de son code génétique…) afin d'être plus en adéquation avec quelque chose (son milieu naturel, une situation politique nouvelle, une technologie qui bouleverse son organisation…)
- C'est la capacité des être vivant à se développer dans les conditions particulières (milieu humide, ses, salé, désertique, etc.)
- C'est une caractéristique qui aide un animal à survivre dans son habitat.

- L’adaptabilité est la capacité d’un système à ajuster ses mécanismes, ses processus et sa structure à des changements climatiques hypothétiques ou réel. L’adaptation peut être spontanée ou planifiée ; elle peut se produire en réponse à, ou en prévision d’une évolution des conditions.
- C’est le processus et le résultat de l’ajustement d’un organisme vivant ou d’une société aux conditions d’un environnement donné, ce qui lui permet de survivre, de se reproduire et de se développer.
- L’adaptabilité désigne l’action d’ajuster et réagir face aux variations des contraintes de l’environnement et des besoins des utilisateurs. On entend par « l’adaptabilité » la capacité ou le degré d’adaptation [Hamdane 08].

Ce sont des définitions tirées essentiellement d’encyclopédies et des dictionnaires, mais qu'entend-on nous par adaptation dans les systèmes multi agent ?

Deux niveaux d'adaptation peuvent être définis. Premièrement, l'adaptation au niveau du système peut être associée à l'auto-organisation. L'environnement agissant sur l'organisation d'un système, tout changement de l'environnement peut modifier l'organisation de ce système. Deuxièmement, l'adaptation existe au niveau de l'entité qui est l’agent. L'adaptabilité d'un agent peut être associée à sa capacité d'apprentissage afin qu'il soit capable d'atteindre ses objectifs malgré des changements de l'environnement. On parle alors d'agent adaptatif [Rodin 04].

I.3.2 L’adaptation et les systèmes multi agents

De nombreuses applications nécessitent la réalisation de logiciels caractérisés par une phase de spécification incomplète pour les raisons suivantes [Gleizes et Al 01] :

- le système doit évoluer dans un environnement dynamique et il est impossible de spécifier totalement toutes les situations auxquelles il aura à faire face
- le système est ouvert
- le système est complexe
- il n'existe pas de solution algorithmique connue pour résoudre le problème
- l'organisation interne du système n'est pas connue, a priori.

L’imprévu étant inhérent à la vie de ces systèmes, l’adaptation, devient un moyen pour parvenir à surmonter les perturbations éventuelles de l’environnement et pour réaliser des systèmes adaptatifs.

En effet, l'auto organisation, qui correspond à un changement décidé de manière autonome [Gleizes et Al 01], est un moyen essentiel pour que les systèmes multi-agents puissent évoluer dans un environnement dynamique, et donc s'adapter [Marcia 96].

Apprendre pour le système consiste donc à transformer sa fonction actuelle de manière autonome afin de s'adapter à l'environnement c'est-à-dire à changer l'organisation entre ses agents. Cet apprentissage doit lui permettre d'avoir une activité "correcte" dans l'environnement dans lequel il est immergé : ceci définit l'adéquation fonctionnelle d'un système [Gleizes et Al 01].

I.3.3 Agent adaptatif

Un agent s'adapte au sein d'un SMA, si face à une situation imprévue par le concepteur de l'application finale, l'agent ne se bloque pas mais réagit. Pour cela l'agent abandonne ponctuellement, parfois définitivement, l'objectif individuel qu'il poursuit. Le système peut alors modifier son comportement face aux évolutions de son environnement sans en tirer de conséquences durables. Ainsi, pour deux situations identiques rencontrées au cours de son histoire, le système adaptatif se comportera exactement de manière identique.

Une situation imprévue pour un SMA, supposé accomplir de manière performante et satisfaisante la fonction pour laquelle il a été conçu, est une des situations non coopératives déjà définies. En effet, si le fonctionnement du système en l'état est bon, aucune situation non coopérative ne peut se produire ; de telles situations surviennent si l'environnement perturbe le système [Camps et Gleizes 96].

Les agents adaptatifs sont capables d'apprendre par l'expérience et d'évoluer pour généraliser leur connaissance [Tran 07].

Les techniques d'apprentissage classique de l'intelligence artificielle s'appliquent particulièrement bien à l'apprentissage d'un agent (qu'il s'agisse de faire évoluer ses capacités d'action, de décision ou d'analyse) [Drieu 01]

I.3.4 Agent anticipatoire

L'anticipation est une activité mentale que tous les êtres vivants doués d'intelligence possèdent [Tran 07]. C'est un élément essentiel du raisonnement cognitif. Elle peut être définie comme la capacité à adapter son comportement en fonction de prédictions du futur.

Cette caractéristique est un facteur essentiel pour la survie des espèces. La prise de décision dans le présent est conditionnée par des résultats prévus ou souhaités dans le futur. Les systèmes dotés d'anticipation se composent de deux modules : la prédiction des états futurs de l'environnement et l'utilisation de la prédiction pour alterner le comportent présent pour éviter des situations non souhaitées. Par exemple, si un système contient un modèle de la météo et qu'il prédit qu'il va peut-être pleuvoir dans un futur proche, alors on doit prendre un parapluie afin de ne pas être mouillé. Un tel système correspond à un système anticipatoire [Tran 07].

I.3.5 Types d'adaptation

D'après [Guessoum 03], l'adaptation peut être :

- structurelle : il s'agit d'adapter la structure de l'agent à l'évolution de son environnement. La réorganisation de la liste des accointances d'un agent à l'arrivée de nouveaux agents dans le système est un exemple de cette adaptation,
- comportementale : il s'agit d'adapter le processus de décision de l'agent à l'évolution de son environnement. Adapter par exemple, le choix des actions à la situation du marché tel est le cas des firmes.

I.3.6 L'adaptation comportementale

Un agent adaptatif est un système qui cherche la satisfaction de ses tendances fondamentales. Pour cela, il adapte et modifie son comportement de manière à corréler ses tendances et son environnement.

L'adaptation comportementale peut être statique ou dynamique [Rejeb 05].

L'adaptation est statique quand on dote les agents de règles de comportements obtenues par la simulation de l'évolution de l'environnement et des réactions de l'agent à cet environnement.

Ces règles sont figées et nécessitent la prévision des changements possibles de l'environnement au moment de la conception de l'agent. L'adaptation est, par contre, dynamique quand les règles de comportement sont construites dynamiquement au fur et à mesure que l'environnement change.

L'adaptation dynamique [Phan 03] est considérée comme étant la capacité de l'agent à s'adapter aux changements imprévus de son environnement par son évolution continuelle sans l'interruption de son exécution. Ce type d'adaptation nécessite des capacités d'apprentissage.

Maes [Maes 94] propose différentes méthodes d'adaptation dynamique :

- apprentissage par renforcement : l'agent effectue l'action la mieux notée, puis modifie la note en fonction des résultats. Il explore dans un pourcentage *n* de cas. Il suggère parmi ces méthodes, l'utilisation des systèmes de classeurs1. Dans ces systèmes, l'agent choisit parmi les classeurs qui correspondent à la situation la mieux notée. Il effectue les opérations correspondantes et améliore les notes des derniers classeurs utilisés quand le résultat est probant. Ces classeurs sont, par la suite, croisés génétiquement afin de conserver une exploration de l'environnement,
- construction de modèles : l'agent construit des modèles statistiques des actions qu'il effectue. En se référant à ces modèles, il choisit l'action qui le fera le plus progresser. En fonction des résultats et par corrélation avec ce qu'il a déjà appris, il modifie ses modèles pour en sélectionner le meilleur.

I.4 Les travaux sur l'adaptabilité

I.4.1 Les travaux de Sébastien Leriche et Jean-Paul Arcangeli

Afin de permettre aux agents de s'adapter statiquement et dynamiquement à leur environnement d'exécution, Sébastien Leriche et Jean-Paul Arcangeli [Sébastien et Arcangeli 04] proposent une architecture à micro-composants interchangeables. Ces microcomposants correspondent chacun à un mécanisme d'exécution non fonctionnel activé par délégation (boîte aux lettres, déplacement. . .).

La réflexivité de l'architecture permet de travailler sur deux niveaux de programmation bien distincts.

Ainsi le niveau de base contient le code fonctionnel de l'agent (comportement déterminé par le programmeur de l'application agent) et le niveau méta décrit les mécanismes d'exécution spécialisés (micro composants) de l'agent.

Prenons l'exemple d'un agent mobile créé sur un ordinateur relié à un réseau local filaire isolé de l'extérieur par des protections adéquates (pare-feu. . .) qui se déplace ensuite vers un environnement non fiable du point de vue de la sécurité des communications et avec des pertes potentielles du signal réseau (typiquement un portable dans un environnement public relié au réseau par liaison sans fil de type WiFi). Cet agent peut utiliser à l'origine des protocoles de communications ayant des propriétés non fonctionnelles de faible sûreté, faible sécurité et forte performance.

Après la mobilité, il serait judicieux grâce à des mécanismes d'adaptation dynamique d'utiliser des protocoles dont la sémantique est identique, mais ayant des propriétés non-fonctionnelles de forte sûreté et de forte sécurité au détriment de la performance.

Dans d'autres cas, il peut être nécessaire d'adapter le protocole de localisation des agents mobiles, les protocoles de communication (suivant le type de connexion), les informations et les services mis à disposition sur les sites visités (découverte et accès à des imprimantes, bases de données, informations géographiques. . .).

Les systèmes d'agents mobiles ont donc besoin de pouvoir s'adapter dynamiquement aux conditions d'exécution qu'ils rencontrent durant leur vie pour offrir un maximum de services utiles et efficaces aux applications qui les utilisent. Mobilité et adaptation sont ainsi deux propriétés complémentaires des agents ; toutes les deux participent à son autonomie.

Le modèle d'architecture proposé par Sébastien Leriche et Jean-Paul Arcangeli **dans** [Sébastien et Arcangeli 04] afin de permettre aux agents de s'adapter dynamiquement est :

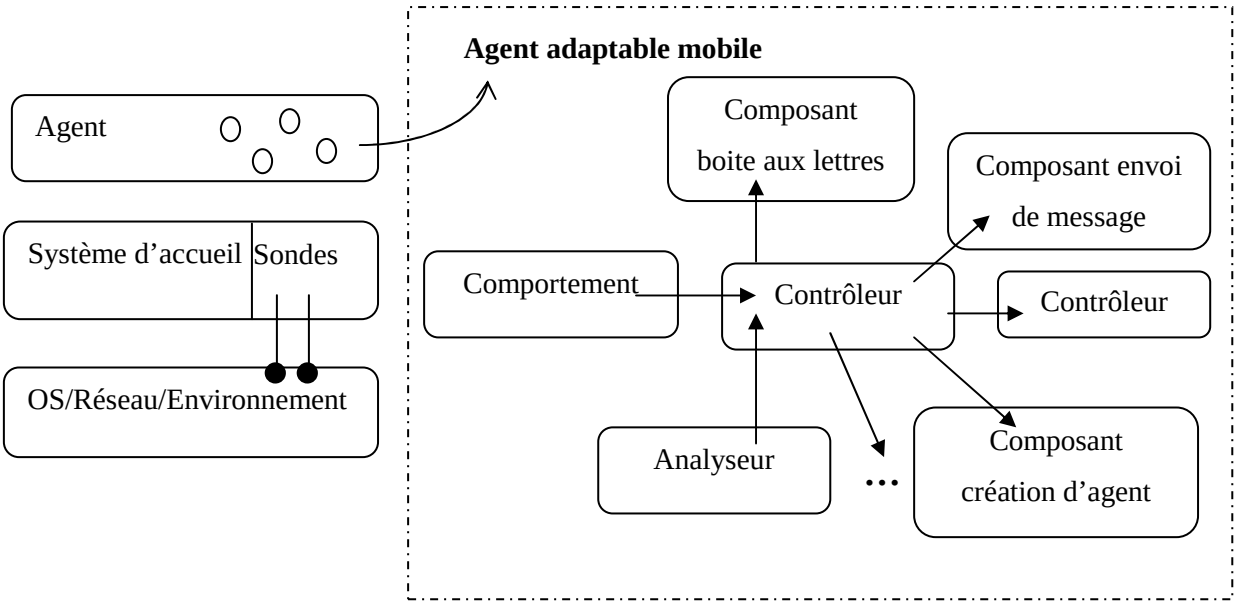


Figure I.4 Schémas de l'architecture des agents mobiles adaptables

Cette architecture à méta-objets est inspirée de l'architecture réflexive proposée dans [Marcoux et Al 98], avec ses méta-composants. La séparation claire de deux niveaux de programmation (concept de séparation des aspects) est rendue possible par cette réflexivité. Ainsi le niveau de base contient le code fonctionnel de l'agent (comportement déterminé par le programmeur de l'application agent) et le niveau méta décrit les mécanismes d'exécution spécialisés (micro-composants).

I.4.2 Les travaux de Zahia Guessoum et Al[Guessoum et Al 02]

Zahia Guessoum et Al [Guessoum et Al 02] présente une approche de construction d'agents par assemblage et raffinement de composants, mis en œuvre dans la plate-forme DIMA.

Le point de départ est la conception d'un composant minimal de pro-activité, étendant la notion d'objet vers la notion de composant pro-actif. Ce composant est ensuite spécialisé incrémentalement de manière à représenter différents types d'architectures d'agents (agents réactifs, agents adaptatifs, …).

- **Architecture d'agents DIMA**

DIMA [Sansonnet et Fegas 05] est un environnement de développement de systèmes multi-agents dont le développement a débuté en 1993, dans le cadre de la thèse de Z. Guessoum dans le thème Objets et Agents pour Systèmes d'Information et de Simulation (OASIS) du LIP6. DIMA a été utilisé pour développer plusieurs applications réelles (Ventilation artificielle, simulation de modèles économiques,...). Ces applications peuvent être des simulations, des résolutions de problèmes ou des systèmes de contrôles ayant éventuellement des contraintes temps réel. La première version de DIMA a été implémentée en Smalltalk-80 et a été ensuite portée en JAVA .

L'architecture d'agent DIMA [Guessoum et Al 02] propose de décomposer chaque agent en différents modules ou composants dont le but d'intégrer des paradigmes existants notamment des paradigmes d'intelligence artificielle. Ces modules représentent les différents comportements d'un agent tels que la perception (interaction agent-environnement), la communication (interaction agent-agent) et la délibération. Un agent peut ainsi avoir plusieurs composants qui peuvent être réactifs ou cognitifs. Un comportement réactif est décrit par un ensemble de méthodes (au sens objets) alors qu'un comportement cognitif peut avoir des méthodes qui activent des mécanismes de raisonnement (par exemple, un moteur d'inférence).

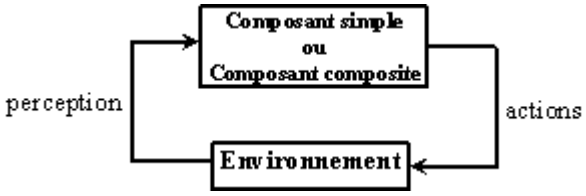


Figure I.5: L'architecture d'agents de DIMA

Pour gérer les interactions entre ces différents composants, DIMA [Sansonnet et Fegas 05] propose un module de supervision représentant le méta-comportement de l'agent. Ce méta-comportement réifie le mécanisme de contrôle de l'agent, il gère les interactions entre les différents modules et permet à l'agent d'observer ces comportements.

Un agent est ainsi une entité proactive et autonome. Cette architecture permet de dépasser la dichotomie classique et offre la possibilité de développer des applications multi-agents dont la granularité des agents est variable. Elle offre ainsi la possibilité de concevoir et réaliser différents types d'agents:

- des agents réactifs, décrits par des comportements réactifs. Ces agents ne disposent que d'un protocole de communication réduit. Ils répondent uniquement à une loi de type stimulus/réponse.

- des agents cognitifs, décrits par des comportements cognitifs. Ces agents disposent d'une capacité de raisonnement, d'une aptitude à traiter des informations diverses liées au domaine d'application, et d'informations relatives à la gestion des interactions avec d'autres agents et l'environnement.
- des agents hybrides, conçus pour allier des capacités réactives à des capacités cognitives, ce qui leur permet d'adapter leur comportement en temps réel à l'évolution de leur univers.

a - Les Composants proactifs

Un composant proactif présente une entité autonome et proactive. Le noyau de base de DIMA est un framework de composants proactifs. Un composant proactif à différentes compétences (une suite d'actions qui permettent de changer l'état interne de l'agent ou envoyer des messages aux autres agents) et son activité n'est pas restreinte à l'envoi et à la réception de messages. Ainsi, un composant proactif est décrit par un but, une compétence de base (ou comportement) et un méta-comportement qui définit la manière dont les compétences sont sélectionnées, séquencées et activées en fonction du but.

Différents paradigmes (automate, règle de production, etc.) sont réutilisés pour définir de nouvelles classes de composants proactifs. Par exemple, le comportement d'un composant proactif peut être modélisé par un ATN (Augmented Transition Network). Le composant proactif peut être considéré comme une brique de base pour construire des agents.

b - Agents interactifs

Pour définir des agents interactifs, les composants proactifs ont été enrichis d'un module de communication. Ce dernier gère les interactions avec les autres agents. Les comportements de ces agents interactifs intègrent notamment deux types d'actions : les actions liées aux concepts d'agents tels que l'envoi et la réception de messages et des actions liées au domaine.

Chaque agent interactif a deux composants : une boite aux lettres pour stocker ses messages et un module de communication pour gérer les messages envoyés et reçus.

c - Agents adaptatifs

Un méta-comportement est introduit dans l'architecture d'agents de DIMA. Ce méta-comportement donne à chaque agent la capacité de prendre des décisions appropriées au sujet de contrôle ou d'adapter son comportement avec le temps à des nouvelles circonstances. Il fournit à l'agent un mécanisme d'auto-contrôle pour adapter dynamiquement ses comportements selon son état interne et celui de son environnement.

Le méta-comportement est basé sur les données de l'agent lui-même, son environnement, et le système de décision utilisé. Il est basé sur deux types d'éléments : conditions et actions.

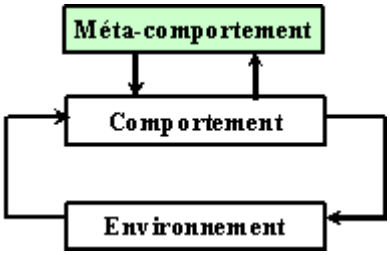


Figure I.6 : Architecture d'agents adaptatifs

I.5 Conclusion

Ce chapitre a présenté une vue globale sur les SMA, ainsi que quelque notion de base sur l'adaptation et les SMA adaptatifs. Le chapitre suivant est consacré aux différentes modèles de spécification formelles pour les SMA

.

Chapitre II

Modèles De Spécification Formelles Pour les SMA

II.1 Introduction

Dans plusieurs domaines de la science, l'étude d'un système complexe revient à l'étude d'un modèle de ce système. Il existe un grand nombre de modèles applicables à la compréhension et à la conception de systèmes multi-agents. On peut distinguer deux grandes familles: les modèles algébriques, qui tendent à décrire un agent en termes mathématiques, et les modèles opératoires qui utilisent des structures a priori plus informatiques, même si certains, tels les automates à états finis ou les réseaux de Petri, sont eux-mêmes formalisables en termes algébriques.

Aujourd'hui, les réseaux de Pétri (RdP) constituent l'un des modèles formels les plus avancés et les plus complet pour la description logique des structures d'un système parallèle. Ils sont des outils prometteurs pour la description et l'étude des systèmes de traitement d'informations concurrents, asynchrones, distribue, parallèles, non déterministes et stochastiques.

Comme outil graphique, les réseaux de Pétri permettent la visualisation du comportement dynamique et des activités concurrentes du système. Comme outil mathématique, ils permettent l'analyse de propriétés importantes, telles que l'absence d'une situation de blocage ou l'existence d'un régime permanent. Grâce à leur généralité et à leur souplesse, les réseaux de Pétri ont été proposés pour une large variété d'applications.

II.2 Modèles de spécification formelles pour les SMA

II.2.1 Les automates à états finis (AEF)

II.2.1.1 Définition

Un automate d'états finis déterministe est un quintuplet A=(Σ,Q,δ,q0,F) tel que :

- Σ est un alphabet fini.
- Q est un ensemble fini d'états de A.
- $q0 \in Q$ est l'état initial.
- $F \subseteq Q$ est l'ensemble des états finaux.
- $\delta : Q \times \Sigma \rightarrow Q$ est la fonction de transition de l'automate.

II.2.1.2 Représentation par diagrammes

La figure II.1 montre que l'automate se représente sous la forme d'un graphe orienté dont les nœuds sont les états de l'automate et les arcs ses transitions. Sur les arcs, on indique à la fois les évènements qui le font passer d'un état à un autre et les actions qui doivent être entreprises lors de cette transition ; Alors il s'agit d'une succession d'état ou le changement d'un état à un autre correspond à l'émission ou la réception d'un message.

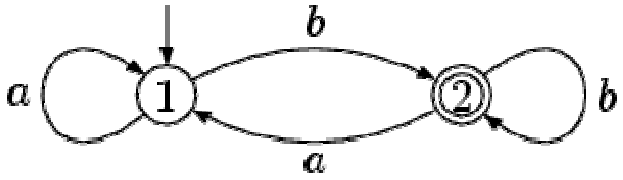


Figure II.1: Exemple d'un automate d'états finis

Les états (ici 1 et 2) sont reliés par des flèches selon la fonction de transition δ. Ici, on a δ (1,a)= 2.

II.2.2 Les réseaux de Petri

Les réseaux de Pétri sont définis comme étant un formalisme qui permet la description et l'analyse du comportement des systèmes concurrents, introduit par **Carl Adem Petri** en 1962. Les définitions concernant les **réseaux de Petri** portées sur deux aspects:

• **Un aspect structurel**: Quelles sont les actions, quels sont les sites, quelles sont les conditions pour qu'une action soit possible et quelles sont les conséquences d'une action?

• **Un aspect comportemental**: Comment représenter le fonctionnement d'un réseau de pétri? c-à-d ce qui se passe quand une action ou plusieurs actions sont exécutées.

II.2.2.1 L'aspect structurel

Un réseau de Petri (***R***) est un triple ***R*= (*P, T, W*)** où ***P*** est l'ensemble des places (les places représentent les sites) et ***T*** l'ensemble des transitions (les transitions représentent les actions) tel que $P \cap T = \phi$ et ***W*** est la fonction définissant le poids porté par les arcs tel que $W : ((P \times T) \cup (T \times P)) \rightarrow \mathbf{N} = \{0, 1, 2, \ldots\}$.

Le réseau ***R*** est **fini** si l'ensemble des places et des transitions est fini c-à-d $|P \cup T| \in \mathbf{N}$.

Un réseau ***R* = (*P, T, W*)** est **ordinaire** si pour toute $(x, y) \in ((P \times T) \cup (T \times P))$: $W(x, y) \leq 1$.Dans un réseau ordinaire la fonction ***W*** est remplacée par ***F*** où :

$F \subseteq ((P \times T) \cup (T \times P))$ tel que $(x, y) \in F \Leftrightarrow W(x, y) \neq 0$.

Pour chaque $x \in P \cup T$:

$^{*}x$ représente l'ensemble des entrées de ***x* :** $^{*}x = \{y \in P \cup T | W(y, x) \neq 0\}$

x^{*} représente l'ensemble des sorties de ***x* :** $x^{*} = \{y \in P \cup T | W(x, y) \neq 0\}$

Remarque* :** si $^{*}x = \phi$, ***x est dite **source**, si $x^{*} = \phi$, ***x*** est dite **puits**.

II.2.2.2 L'aspect comportemental

Le comportement d'un réseau de Petri est déterminé par sa structure et par son état. Pour exprimer l'état d'un réseau de Petri, les places peuvent contenir des jetons qui ne sont que de simples marqueurs.

II.2.2.2.1 Que signifie l'état dans un réseau de Petri

Dans la théorie des réseaux de Petri, l'état d'un réseau est souvent appelé *marquage* du réseau qui est définit par la distribution des jetons sur les places. Le marquage d'un réseau de Petri ***R*= (*P, T, W*)** est défini par la fonction de marquage $M : P \rightarrow \mathbf{N}$.

Un réseau de Petri **marqué** est dénoté par $\sum = (P, T, W, M_0)$ où M_0 est le marquage initial. Le comportement d'un **réseau de Petri** marqué est déterminé par ce qu'on appelle *règle de franchissement.*

II.2.2.2.2 Que représente une règle de franchissement?

Une règle de franchissement est une simple relation de transition qui définit le changement d'état dans un réseau marqué lors de l'exécution d'une action. Afin de définir une règle de franchissement, il est nécessaire de formaliser quand le réseau peut exécuter une action: on dit qu'une transition $\mathbf{t} \in T$ peut être **franchie** à partir d'un marquage *M* (qui représente l'état du system à un instant donné) si et seulement si chaque place d'entrée $\mathbf{p} \in {}^*\mathbf{t}$ de la transition **t** contient au moins un nombre de jetons qui est supérieur ou égal au poids de l'arc reliant cette place d'entrée **p** avec la transition **t** tel que: $M(\mathbf{p}) \geq W(\mathbf{p}, \mathbf{t})$ $\forall\ \mathbf{p} \in P$.

Une règle de franchissement est définie par $M'(\mathbf{p}) = M(\mathbf{p}) - W(\mathbf{p}, \mathbf{t}) + W(\mathbf{t}, \mathbf{p})$ pour tout $\mathbf{p} \in P$, ce qui veut dire que lorsque la transition **t** est franchit à partir d'un marquage M, il faut saisir $W(\mathbf{p}, \mathbf{t})$ jetons à partir de chaque place d'entrée à la transition **t** et déposer $W(t, p)$ jetons dans chaque place de sortie de la transition **t** ce qui permet de produire un nouveau marquage M'.

Le franchissement d'une transition **t** dénoté par $M\ [\mathbf{t}\ \rangle M'$ est dit l'**occurrence** de **t**. On dit que deux transitions $\mathbf{t_1}$, $\mathbf{t_2}$ (pas certainement distinctes) sont franchies en concurrence par un marquage M si et seulement si $\mathbf{M}(\mathbf{p}) \geq W(\mathbf{p}, \mathbf{t_1}) + W(\mathbf{p}, \mathbf{t_2})$ pour toute $\mathbf{p} \in P$.

Cette vision de l'exécution concurrente de deux transitions dans un RdP est contradictoire avec celle qui impose que deux occurrence de transition sont parallèles si et seulement si : elles sont causalement indépendantes et n'ont pas une relation de conflit entre eux. Deux occurrences sont en conflit si l'un des deux peut avoir lieu mais pas toutes les deux.

II.2.2.2.3 L'exécution d'un réseau de Petri

L'exécution d'un réseau de Petri est définie en termes d'un ensemble de séquences d'occurrence. Une séquence d'occurrence est une séquence de transitions franchissables dénotée par $\sigma = M_0\ \mathbf{t_1}\ M_1\ \mathbf{t_2} \ldots$ tel que $M_{i-1}\ [\mathbf{t}_i\ \rangle M_i$. Une séquence $\mathbf{t_1}\ \mathbf{t_2} \ldots$ est une séquence

de transitions (commencée par le marquage ***M***) si et seulement si il existe une séquence d'occurrence $M_0\ t_1\ M_1$... avec $M=M_0$. Si la séquence finie $t_1\ t_2\ \ldots t_n$ conduit à un nouveau marquage M' à partir du marquage ***M***, on écrit $M\ [t_1\ t_2 \ldots t_n\ \rangle M'$ ou simplement $M[t_1 t_2 \ldots t_n\rangle$ si on ne veut pas spécifier le marquage résultat.

L'ensemble de marquages accessibles d'un réseau marqué (**P, T, W, M_0**) est défini par $[M_0\rangle = \{M \mid \exists\, t_1\ t_2 \ldots t_n : M0\ [t_1 t_2 \ldots t_n\rangle\ M\}$.

II.2.2.3 Représentation d'un réseau de Petri

II.1.2.3.1 Représentation graphique

L'un des aspects les plus agréables des réseaux de Pétri est qu'il est extrêmement aisé de les visualiser; c-à-d, donner une interprétation graphique à sa structure qui peut être représentée à travers un **graphe** bipartie fait de deux types de sommets: les places et les transitions reliées alternativement par des arcs **orientés** qui portent des poids entier positifs, si un poids n'est pas porté alors il est égal à **1** (**RdP ordinaire**). Généralement, les places sont représentées par des cercles et les transitions par des rectangles, le marquage d'un **RdP** est représenté par la distribution de jetons dans l'ensemble de ses places telle que chaque place peut contenir un ou plusieurs jetons représentés par des points dans le cercle représentant la place.

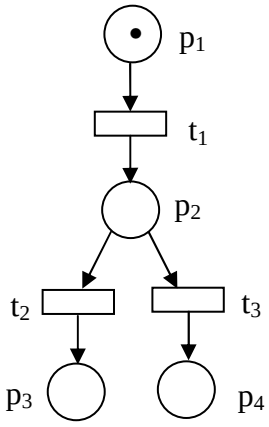


Figure II.2 Réseau de Petri marqué

II.2.2.3.2 Représentation matricielle

Une représentation matricielle d'un RdP est offerte afin de simplifier les Tâches d'analyse et de vérification effectuée sur un modèle RdP. Agir sur une représentation

graphique d'un modèle RdP est une Tâche délicate en comparant avec une représentation matricielle.

Il est possible de représenter la fonction ***W*** (fonction de poids) par des matrices.

Définition

Soit Un réseau de Petri ***R*= (*P, T, W*)** avec ***P*= {p_1, p_2, …, p_m} et *T*= {t_1, t_2, …, t_n}**, on appelle matrice des pré conditions *pré* la matrice m × n à coefficients dans **N** tel que ***pré* (i,j)= *W*(p_i, t_j)**, elle indique le nombre de marque que doit contenir la place p_i pour que la transition t_j devienne franchissable, de la même manière on définit la matrice des post conditions ***post*** la matrice **n × m** tel que ***post* (i,j)= *W*(t_j, p_i)** contient le nombre de marques déposées dans p_i lors du franchissement de la transition t_j. La matrice **C= *post* – *pré*** est appelée matrice d'incidence du réseau (**m** représente le nombre de places d'un **réseau de Petri** et **n** le nombre de transitions.)

Le marquage d'un réseau de Petri est représenté par un vecteur de dimension **m** à coefficients dans **N**. La règle de franchissement d'un réseau de Petri est définie par

$$M'(p) = M(p) + C(p, t).$$

Exemple

Pour le réseau de la figure II.2

***P*= {p_1, p_2, p_3, p_4}** ***T*= {t_1, t_2, t_3}**

$$Pré = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \qquad Post = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

La matrice d'incidence est : Le vecteur de marquage *M* est :

$$C = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 1 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \qquad M = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

II.2. 2.4 Les réseaux de Petri colorés

Les réseaux de Petri colorés **[Jensen 97]** sont une voie possible pour obtenir des modèles de taille largement inférieure aux modèles équivalents que l'on obtient avec les réseaux de Petri simples introduits précédemment. Pour un réseau de Petri simple, on ne distingue pas les différents jetons. Cependant, dans un réseau de Petri coloré les jetons ont de différents types (couleurs) qui permettent de les distinguer entre eux.

Parmi les avantages de l'utilisation des RDP Colorés, c'est qu'ils offrent une description hiérarchique aux systèmes ceci signifie qu'on peut construire un grand RDP Coloré en reliant un ensemble d'autres RDP Colorés entre eux. Cette caractéristique permet de modéliser des systèmes largement grand de manière maniable et modulaire. Il existe plusieurs définitions aux RDP colorés **[Jensen 98]** dont on a choisi celle défini dans **[Valette 02]**

Définition

Un RDP coloré $\mathbf{R_C}$ associé à un marquage initial est un 6-uplet

$\mathbf{R_C = < P, T, C_{coul}, C_{sec}, W, M_0 >}$ où :

P est un ensemble fini non vide de places.

- **T** est un ensemble fini non vide de transitions.

  Dont $\mathbf{P} \cap \mathbf{T} = \phi$.

- $\mathbf{C_{coul}}$ est un ensemble fini non vide de couleurs.
- $\mathbf{C_{sec}}$ est la fonction sous-ensemble de couleurs qui à chaque place et à chaque transition associe un sous ensemble de **C** ($\mathbf{P} \cup \mathbf{T} \rightarrow \mathbf{P}(\mathbf{C})$)
- **W** est la fonction d'incidence (équivalent de **C= Post – Pré**), chaque élément **W(p,t)** de **W** est lui-même une fonction :

  $W(p,t) : C_{sec}(p) \times C_{sec}(p) \rightarrow N$.
- $\mathbf{M_0}$ est le marquage initial, pour chaque place et pour chaque couleur possible dans cette place, il associe un nombre de jetons :

  $M_0(p) : C_{sec}(p) \rightarrow N$.

Remarques

1) ***N*** est l'ensemble des entiers naturels.

2) Une transition est franchissable s'il existe pour chaque couleur un nombre suffisant de jetons dans chaque place d'entré.

3) Le franchissement d'une transition va retirer un sous ensemble de jetons de chaque place d'entrée et ajouter un sous ensemble de jetons à chaque place de sortie.

II.2. 2.5 Les réseaux de Petri à Objet

Le concept d'objet consiste à structurer une application autour d'entités en capsulant à la fois des structures de données, sous la forme d'une liste d'attributs, et des méthodes de transformation de ces données.

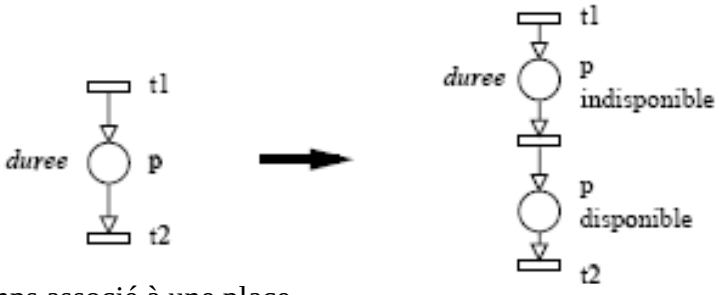
Une classe d'objets est définie par un ensemble d'attributs (propriétés) et un ensemble d'opérations (méthodes) permettant de manipuler les valeurs des attributs. Les classes ne sont que des définitions. Un objet particulier est une instance de classe d'objets.

Dans les réseaux à Objets, les jetons ne sont plus vus comme des constantes, mais comme des n-uplets d'instances de classes d'objets. En plus des attributs définis par la classe, un attribut implicite contient le nom de la place où l'objet est localisé. Les opérations sont associées aux transitions ; elles portent sur les attributs des objets situés dans les places d'entrée. Une opération associée à une transition t ne pourra être exécutée pour un objet que si celui-ci est localisé dans une place d'entrée de t.

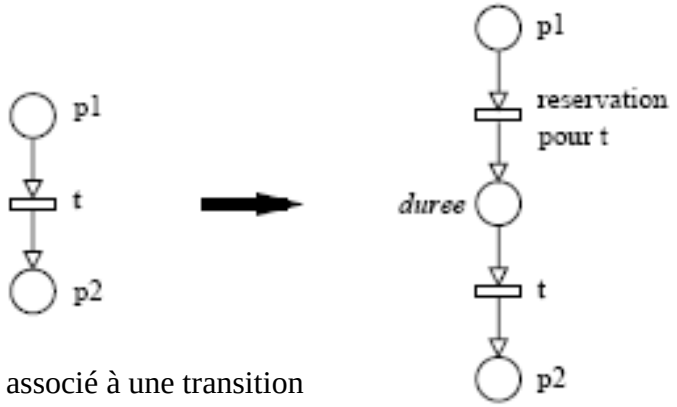
D'un certain point de vue cette approche est moins structurante que l'approche à objet classique puisque une opération est définie dans le cadre d'un ensemble de classes d'objets (celles des objets pouvant être dans les places d'entrées) et non d'une seule classe. D'un autre point de vue, elle est plus structurante car elle introduit une notion de contrôle. Pour qu'une opération soit applicable sur les attributs d'une instance d'objet, il est nécessaire que celle-ci soit dans un certain état c'est-à-dire qu'il soit dans une certaine place.

II.2. 2.6 Les réseaux de Petri temporisés

Un réseau de Petri ordinaire décrit une relation de causalité entre des événements. Un événement a est la cause de b, a précède toujours b, a et b sont ordonnés dans le temps. Le temps est pris en compte de manière qualitative. Des approches vont être présentées si après elles permettent de prendre en compte le temps de façon quantitative. Le temps est directement associé au réseau de Petri, il fait partie du contrôle au lieu d'être rejeté dans la partie donnée de façon non structurée.



a) temps associé à une place



b) temps associé à une transition

Figure II.3 Temporisation d'un réseau de Petri. **[Valette 02]**

- **Temps associé à une place**

Pour une place représentant une activité, il s'agit simplement de noter la durée de cette activité. On peut considérer, qu'en fait, la place concernée éclate en une séquence « place-transition-place ». La première place correspond à l'activité en cours, la transition correspond à l'événement *temps écoulé* et la dernière place correspond à une attente éventuelle (synchronisation avec d'autres activités) après la fin de l'activité (figure II.3 -a-). Pendant que l'activité est en cours le jeton ne peut être utilisé pour franchir une transition. On dit qu'il est *non disponible*. Une fois l'activité terminée, le jeton devient disponible et la transition t2 peut éventuellement être franchie.

- **Temps associé a une transition**

Cette association n'a de sens que si la transition est interprétée comme une activité (interruptible) et non comme un événement instantané. Cette transition est éclatée en une séquence « transition-place-transition » (figure II.3-b-). La première transition correspond à l'événement instantané de début d'activité (on enlève tout les jetons), la place sert a

mémoriser l'activité courante et la dernière transition correspond à l'événement *instantané* de fin d'activité (met les jetons dans les places de sorties).

Le franchissement de la première transition suivant la place p_1 de la forme éclatée correspond à la réservation des jetons (les jetons *réservés* ne peuvent plus être utilisés pour franchir une autre transition que t). Après le franchissement de la deuxième transition les jetons sont libérés.

Dans ce cas comme dans le cas précédant, des jetons qui disparaissent ont été créés ou on peut dire qu'ils sont indisponibles ou réservés pendant un certain temps, puis qui réapparaissent dans le marquage. La figure II.3 montre que les deux approches sont en fait équivalentes.

II.2. 2.7 Les réseaux de Petri temporels

Certains mécanismes impliquent qu'un jeton donné ne soit visible que de certaines transitions à un instant donné. La solution est d'introduire une durée de sensibilisation $\theta_s(t)$ à chaque transition. Une transition t n'est franchissable que si elle est restée sensibilisée durant $\theta_s(t)$. la différence vis-à-vis la durée de franchissement est que pendant toute cette durée les jetons sont disponibles dans les places d'entrée de t et peuvent éventuellement être utilisés par une transition en conflit avec t. Les réseaux de Petri temporels sont plus généraux que les réseaux de Petri temporisés.

II.2. 2.8 Les réseaux de Petri stochastique

Pour pouvoir utiliser la puissance de l'analyse *Markovienne*, il faut que les sytèmes soient sans mémoire du passé, c'est-à-dire que si un événement produit un franchissement de transitions t et transforme le marquage M1 en M2, l'évolution future des transitions qui étaient sensibilisées par M1 avant le franchissement de t doit être identique à celle qu'elle subiraient si elles venaient juste d'être sensibilisées par M2. Seules les distributions géométriques et exponentielles vérifient ce fait. Les réseaux de Petri stochastiques sont définis par de telles distributions afin de pouvoir construire un processus Markovien équivalent et ainsi analyser les comportements du réseau. Les réseaux de Petri Stochastiques ajoutent de l'indéterminisme et des probabilités de franchissement de transitions.

II.3. Conclusion

Dans ce chapitre on a essayé de présenter les concepts de base des réseaux de Petri, qui sont un formalisme de modélisation des systèmes multi agent. Ils représentent un cadre formel de spécification des systèmes grâce à leur sémantique. Un grand avantage de l'utilisation des réseaux de Petri à leur force expressive due a l'aspect graphique offert par ces derniers. Différents extensions des réseaux de Petri ont été présentées dans ce chapitre. Suite à notre recherche, il semble que l'outil formelle n'a pas été utilisé pour spécifié l'adaptabilité de l'agent.

Le chapitre suivant est consacré à la présentation architecture d'un agent qui adapte constamment son comportement à son environnement ; donc, si l'environnement change, le comportement de l'agent va s'y adapter d e lui-même.

Chapitre III

Conception D'Un Agent Adaptatif

III.1 Introduction

Après avoir présenté brièvement les différentes définitions et concept du domaine nécessaire pour notre travail, nous allons présenter dans ce chapitre une architecture proposée pour les agents adaptatifs.

Rappelons que l'objectif est de définir une architecture d'un agent qui adapte constamment son comportement à son environnement ; donc, si l'environnement change, le comportement de l'agent va s'y adapter de lui même.

III.2 Architecture proposée

Dans cette section nous présentons l'architecture proposée pour les agents adaptatifs statiquement ; mais il faut citer d'abord les besoins et la nécessité d'adaptation statique pour les agents.

III.2.1 Nécessité d'adaptation statique pour les agents

Les besoins d'adaptation statique se retrouvent à toutes les échelles du logiciel. En fournissant une architecture souple et capable d'évolution, la maintenance est facilitée et par conséquence, le coût total est réduit.

Ainsi, les agents doivent pouvoir être configurés à « froid », c'est à dire avant leur exécution, et offrir des mécanismes d'évolution de leur architecture.

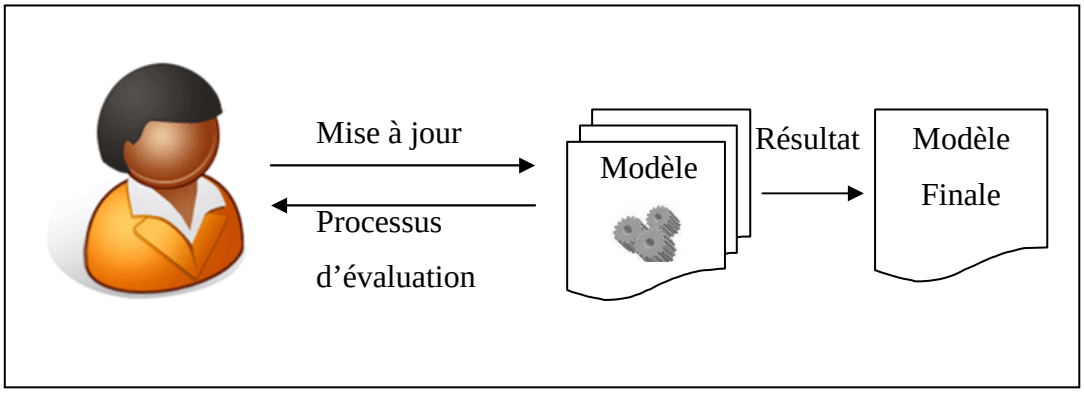


Figure III.1 : Principe d'adaptation statique

Ce modèle peut être utilisé hors-ligne pour faciliter la conception d'un agent. Il suffit donc de simuler l'évolution de l'environnement des agents et d'attendre que les bases de règles se stabilisent. Les agents peuvent ensuite être activés avec la base de règles ainsi obtenue.

III.2.2 Description de l'architecture d'agent adaptatif

Comme nous avons noté dans le chapitre précédent qu'il y a deux types d'adaptation structurelle et comportementale, cette dernière peut être statique ou dynamique ; nous nous intéressons à celle qui est statique où on dote l'agent de règles de comportements et des réactions de l'agent à cet environnement.

Dans ce type d'adaptation et afin de concevoir un agent adaptatif, il est nécessaire de développer une structure décisionnelle de base chargée de choisir une situation auxquelles se trouve confronté l'agent en fonction du contexte.

Plusieurs chercheurs ont souligné l'intérêt de la représentation explicite et séparé du contrôle ou de l'aspect des architectures méta niveau comme [Sébastien et Arcangeli 04] et [Guessoum 03].

En adoptant cette architecture pour notre agent et nous proposons d'introduire un système de décision adaptatif basé RdP; ce système représente un mécanisme d'auto-contrôle pour adapter les différents comportements de l'agent selon son état interne et celui de son environnement.

Ce système représente un méta-comportement pour l'agent et donne une capacité de prendre des décisions appropriées afin d'adapter son comportement, avec le temps, à des nouvelles situations.

Ce méta-comportement se base sur les données de l'agent lui-même, son environnement, et le système de décision utilisé. Il est basé sur deux types d'éléments : conditions et actions, ce qui permet de le modéliser formellement par les RdPs.

En revanche cette spécification formelle permet l'adaptation statique de l'agent de telle sorte qu'on peut modifier dynamiquement le méta-comportement de l'agent tout en vérifiant sa cohérence formellement et l'impact ou l'effet de nouveau méta-comportement introduit sur le comportement de base de l'agent ; ce qui permet de faciliter la conception d'un agent adaptatif.

Nous supposons que l'agent part d'un degré de connaissance minimum pour atteindre à travers ses différentes transactions une base de connaissance de plus en plus riche en informations (expériences), ce qui lui permet d'agir en fonction de ses performances passées.

III.2.2.1 Aspect structurel

L'architecture de notre agent adaptatif possède deux niveaux :

Niveau 1 : *le niveau comportements*; c'est là où on présente les différents comportements possibles ainsi que la base de connaissance de l'agent sur son environnement et sur les autres agents.

Niveau 2 : *le niveau méta-comportements* ; c'est le niveau décisionnel de l'agent, il permet le contrôle global de comportement de l'agent. Il englobe deux principaux modules qui sont le module de décision et le module formelle. A ce niveau l'agent manipule des méta-règles qui existent au niveau du module formel. Ses méta-règles sont construites graduellement.

Lorsqu'un agent perçoit une situation dans l'environnement, le module décisionnel choisi l'un de comportements et le déclenche selon les perceptions et les méta-règles existante au niveau du module formel.

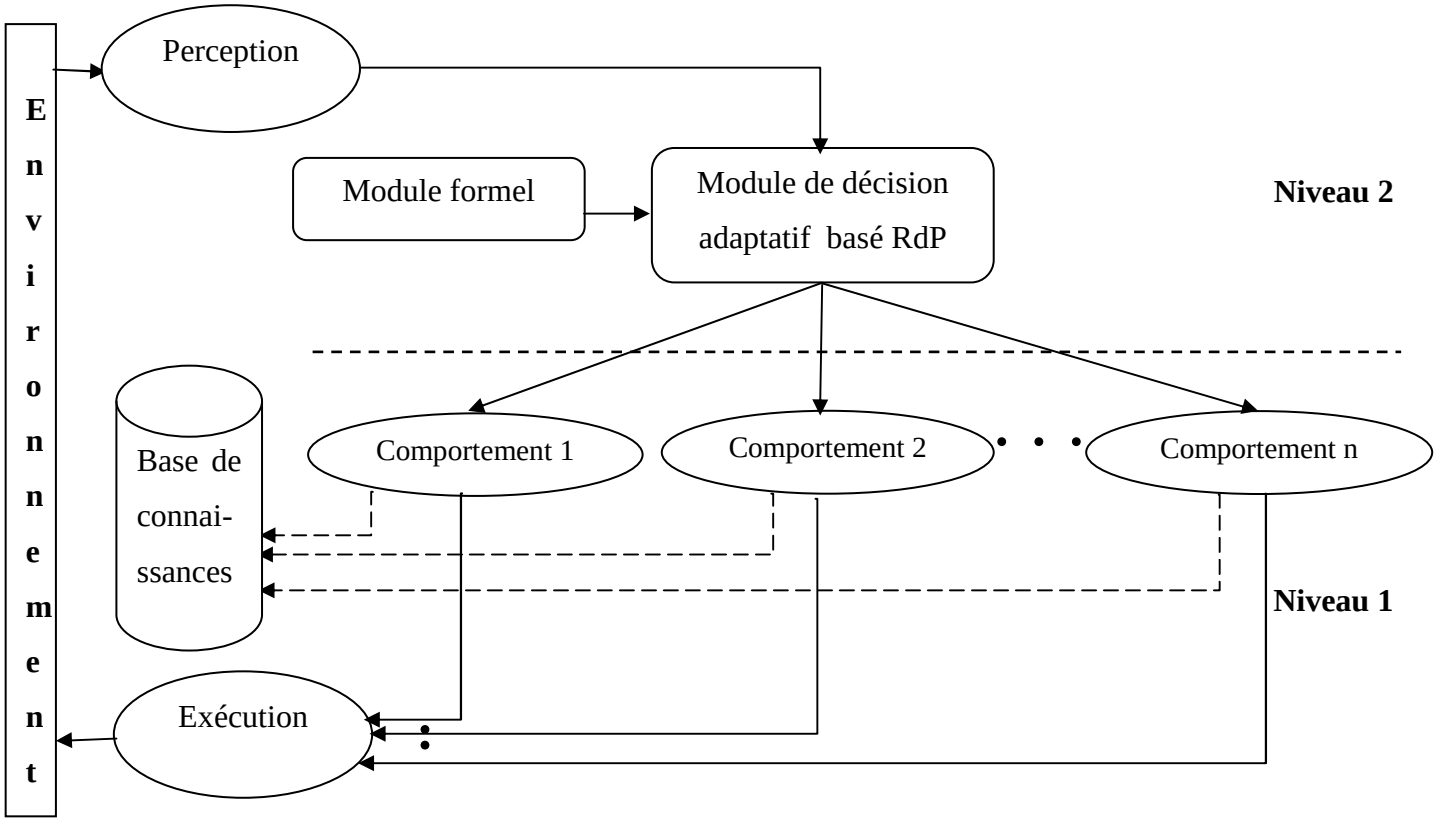
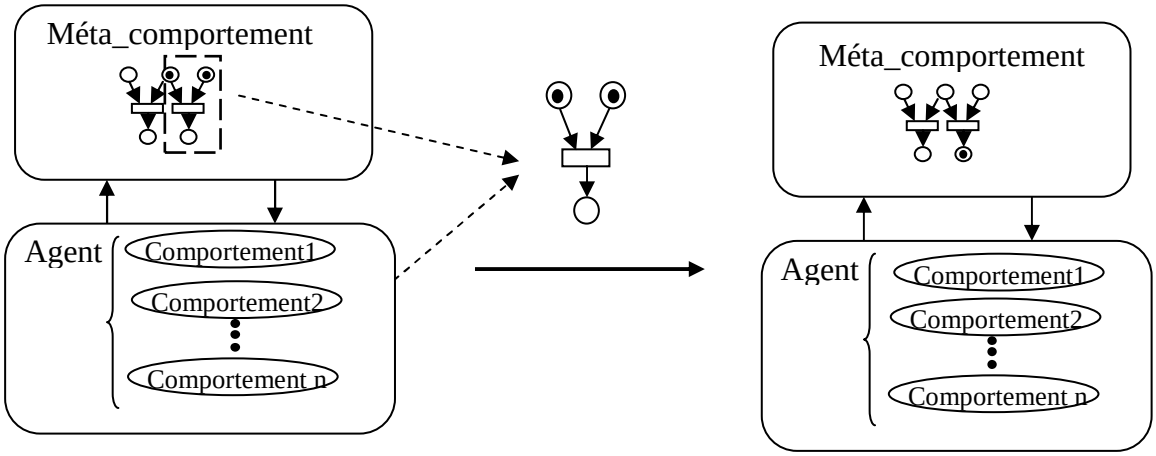


Figure. III.2. Architecture d'un agent adaptatif

III.2.2.2 Aspect fonctionnel

L'agent doit s'adapter au cours du temps ; aussi, il s'agît d'un processus qui se déroule au cours du temps ; à chaque instant, l'agent doit décider le comportement adéquat à effectuer en tenant compte de méta_règles dotés par le concepteur.

Ces méta_règles modélisé formellement par un Rdp ; situé au niveau méta-comportement ; permet le contrôle globale de comportement de l'agent ainsi son intérêt et que l'agent adapte constamment son comportement à son environnement ; donc, si l'environnement change, le comportement de l'agent va s'y adapter et par conséquence l'état de l'agent change (Figure III.3).



L'Agent dans un Etat 1 L'Agent dans un Etat 2

Figure III.3 Aspect fonctionnel de l'agent adaptatif

III.3 Le méta-comportement de l'agent

Le premier niveau de l'architecture proposée comprend un ensemble de modules des comportements ; le problème est alors de choisir le meilleur comportement approprié à la situation présenté. Il s'agit donc de définir le mécanisme et la stratégie nécessaire pour gérer le choix entre ces comportements. La solution que nous proposons consiste à introduire un deuxième niveau qui représente le méta-comportement de l'agent. Ce méta-comportement permet à l'agent d'adapter son comportement à la situation actuelle.

Le fait de doter l'agent d'un méta-comportement permet d'avoir :

1. Une capacité d'adaptation pour permettre au système de tenir compte des nouvelles contraintes et de nouvelles données des agents.

2. Une capacité de généralisation en se basant sur les exemples de problèmes que l'on peut résoudre. Cette propriété permet de palier au problème que pose l'acquisition de connaissances. Le but est de retrouver les règles qui permettent de résoudre le problème à partir d'un ensemble d'exemples.

Ce niveau de supervision repose sur les notions conditions et actions formant des règles. Ces conditions et actions définissent un RdP qui est une représentation synthétique du méta-comportement de l'agent.

Les conditions sont les places d'entrée, les actions représentent les places de sortie, la règle elle-même marqué par une transition et les arcs permet de relier entre les transissions et les places.

III.4. La Méta-règle

Étant donné un ensemble de règles permettant de résoudre une classe de problèmes, une méta-règle indique la manière d'appliquer ces règles (par exemple, appliquer d'abord les règles les moins coûteuses en temps de calcul, ou bien la plus pertinente ou condition actuelle).

Dans notre approche une méta-régle indique la manière de choisir le meilleur comportement parmi plusieurs comportements possibles selon les conditions présentés.

III.5 Description des différents modules de l'architecture

III.5.1 Le Module de perception

La perception constitue l'interface entre l'environnement et l'agent. Ce dernier perçoit l'information, qu'il identifie et reconstitue selon ses croyances internes.

Ce module gère les interactions entre l'agent et son environnement. Son rôle principal est d'initialiser et de mettre à jour un ensemble de données qui regroupe les valeurs de variables externes accessibles par l'agent.

III.5.2 Le Module d'action

Nous appelons par module d'action le module chargé d'exécuter l'acte généré par l'un des comportements déclenché.

III.5.3 Le Module formel

Ce module permet de modéliser le méta-comportement de l'agent en utilisant une spécification formelle qui est les réseaux de Pétri afin de l'analyser et le vérifié par des techniques algébriques de vérification de propriétés que les réseaux de Pétri proposent.

Il nous permet aussi d'ajouter d'autres méta-règles que le concepteur décide leurs importances au modèle étudié.

III.5.4 Le Module de décision adaptatif

Pour s'adapter à des modifications récentes de l'environnement, l'agent doit prendre des décisions à des points discrets de l'échelle des temps. Ces décisions permettent d'adopter le comportement le plus approprié à la situation actuelle. Ce module permet le choix de comportement le plus approprié à la situation actuelle en se basant sur le méta-comportement

III.6 Le Module formel

C'est le module responsable de la modélisation du méta-comportement de l'agent par un réseau de Pétri ; sa structure est la suivante :

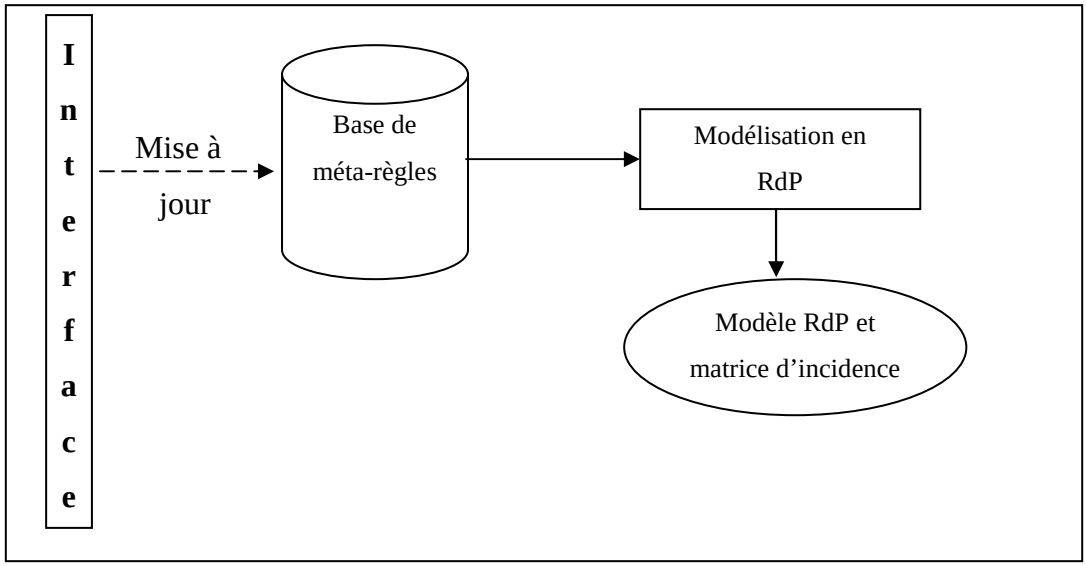


Figure III.4 : Structure du module formel

Pour modéliser le méta comportement avec un RdP il faut doter l'agent d'un ensemble de méta-règles qui facilitent le choix du comportement approprié à chaque situation envisagé par l'agent.

Chaque méta-règle est représentée sous la forme d'une dualiste (condition, action) :

- **Condition :** représente un ensemble de perception représentant un état possible de l'agent.
- **Action :** c'est l'une de comportement adéquat à la perception désigné.

Ces méta-règles seront modélisées par un RdP dont les conditions et les actions modélisés par des places et les règles par des transitions.

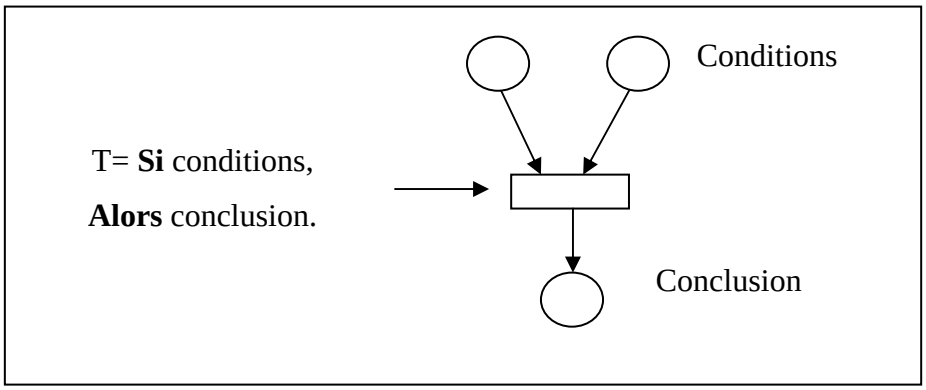


Figure III.5 : Méta-règles vers RdP

Dans le cas où une place du modèle RdP est une entrée de plusieurs transitions différentes (état de conflit), qui représente une condition utilisé par plusieurs règles, le module formel doit éliminer ce conflit tel que présenté dans l'exemple suivant :

Exemple

- **Si** condition 1 **et** condition 2 **Alors** conclusion1 (méta-règle1)
- **Si** condition 1 **et** condition 3 **Alors** conclusion2 (méta-règle2)

Ces deux méta-règles seront transformées de la manière suivant :

- **Si** condition 1 **alors** (condition 1)$_1$ **et** (condition 1)$_2$
- **Si** (condition 1)$_1$ **et** condition 2 **alors** conclusion 1
- **Si** (condition 1)$_2$ **et** condition 3 **alors** conclusion 2

On remarque que le nombre de conclusions supplémentaires ajoutés dépend du nombre de règle qu'utilise la condition de conflit.

D'autre part dans la représentation en RdP, il faut ajouter une transition supplémentaire pour chaque place en conflit, tel que le nombre de places de sortie de cette transition soit égale aux nombres de transitions qui mènent au conflit.

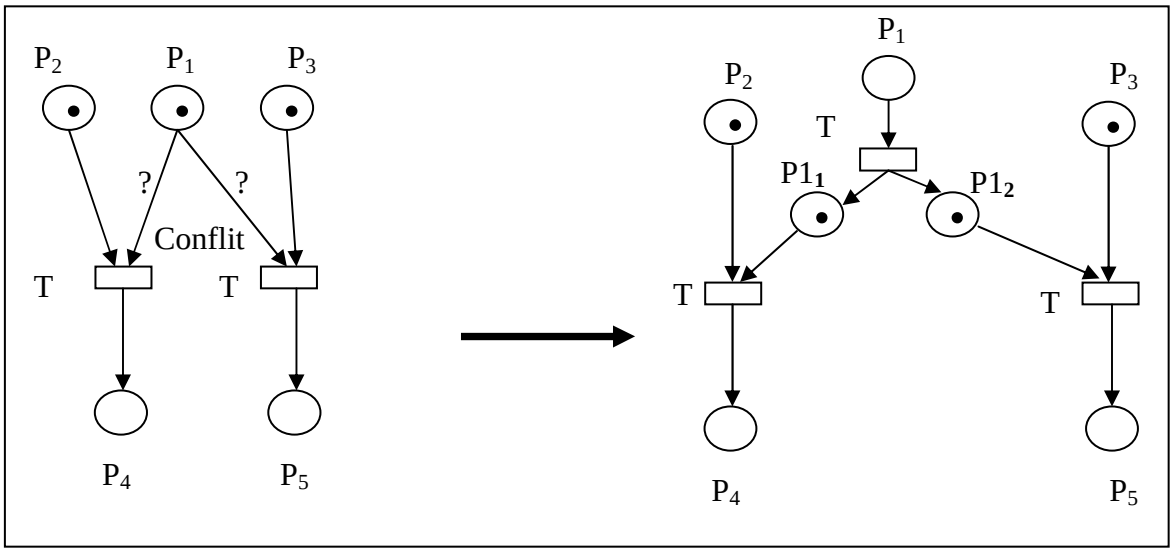


Figure III.6: Elimination du problème de conflit

Une fois le système a terminé la création du modèle RdP, il commence à créer la matrice d'incidence à partir de ce modèle. Donc nous obtenons comme résultats une matrice dont chaque ligne correspond à une place et chaque colonne correspond à une transition. Une case d'intersection entre une ligne p_i et une colonne t_j peut avoir l'une des 3 valeurs suivantes :

+1 si p_i est une place de sortie de la transition t_j.

-1 si p_i est une place d'entrée à la transition t_j.

0 autrement.

Exemple :

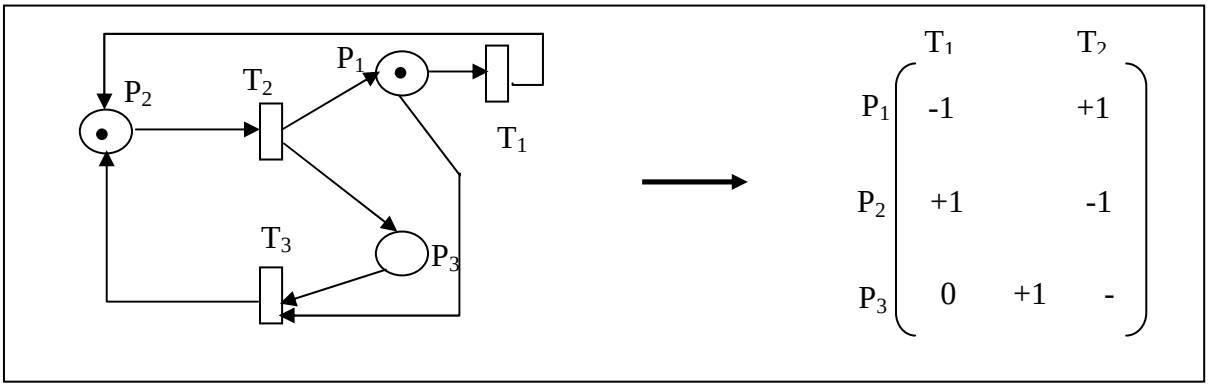


Figure III.7 : Création de la matrice d'incidence

III.7 Le Module de décision adaptatif

L'architecture interne de ce module est présentée dans la figure III.8:

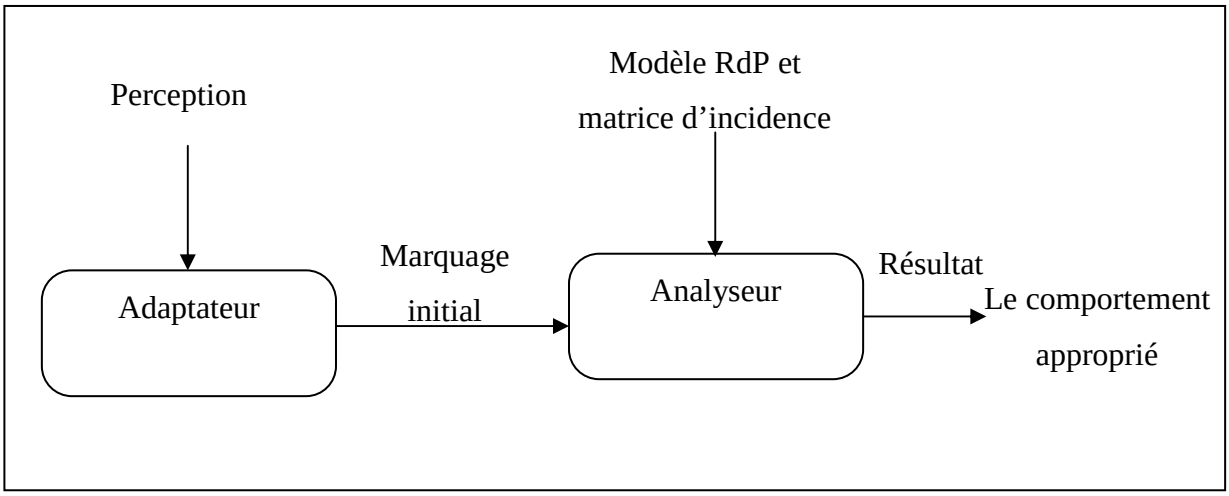


Figure III.8 : Module de décision adaptatif

III.7.1 Adaptateur

Son rôle est de transformer les perceptions en marquage initiale M_0, et associé à chaque perception qui représente une propriété de l'environnement une valeur :

1 si la propriété est vrai.

0 autrement.

III.7.2 Analyseur

La fonction principale que doit accomplir ce module est la construction de l'arbre de recouvrement comme outil d'analyse des RdPs. Son objectif est de découvrir touts les marquages que l'on peut atteindre à partir du marquage initial, ce dernier représente les perceptions de l'agent formant son état.

A ce niveau, ce module a comme entrée le modèle RdP ainsi que le marquage initial $M_{0,}$ donc il va exécuter le RdP afin de choisir le comportement le plus approprié.

- **Fonctionnement de l'analyseur**

Soit M le marquage obtenu après l'exécution d'une transition. L'équation d'état s'écrit alors :

$$\mathbf{M^{t+1}=M_0{}^t + U.V^t}$$

Tel que :

V= [v1, v2,…, vq] où vi est le nombre de fois que la transition t_i sera exécutée.

q : le nombre de transitions.

U : est la matrice d'incidence, et M_0 le marquage initial obtenu de module adaptateur.

M représente donc le marquage que l'on doit obtenir correspondant au comportement qui doit être exécuté (Figure III.9).

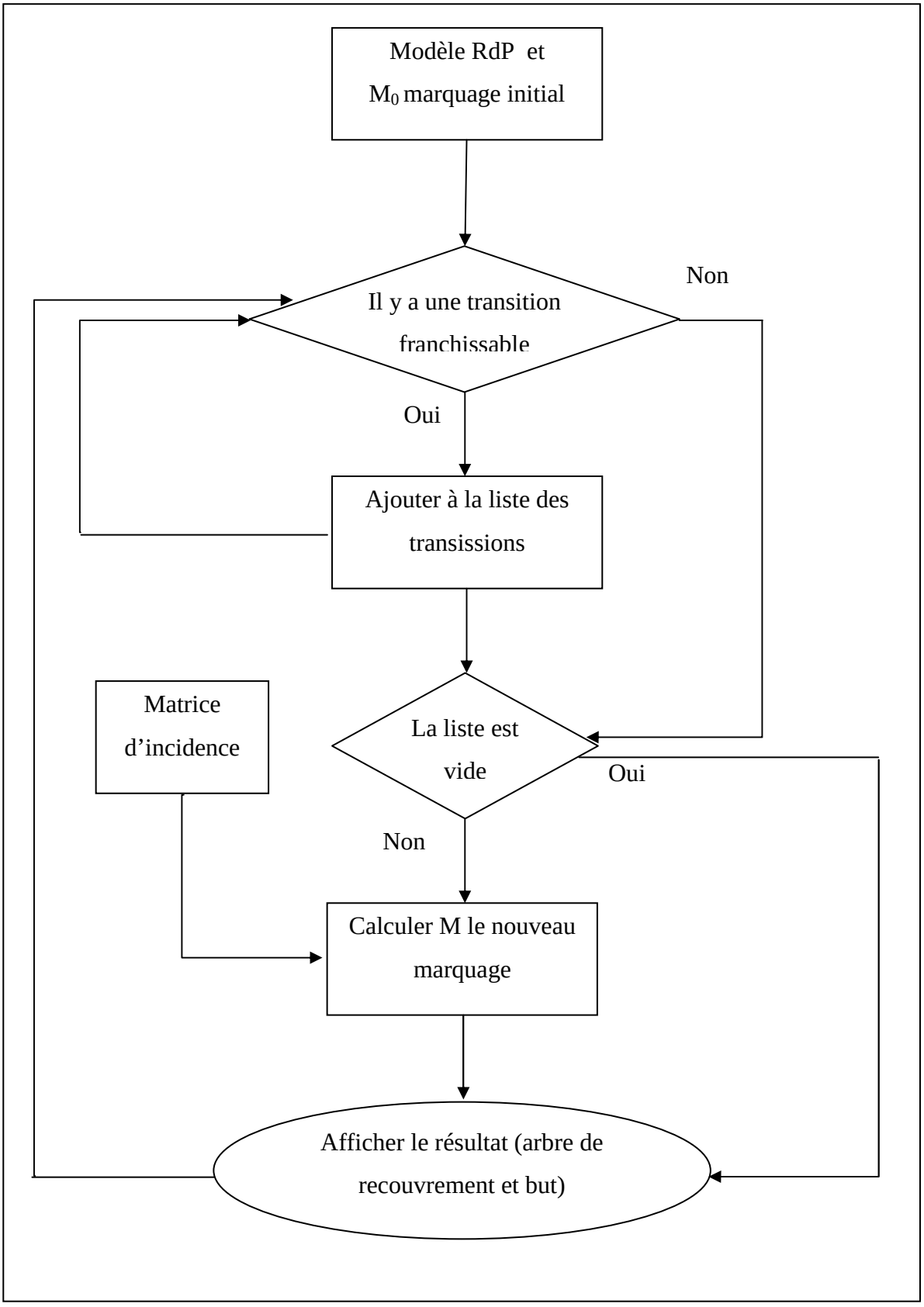


Figure III.9 Le fonctionnement de l'analyseur

III.8 Conclusion

Un agent adaptatif est un système apte a modifié son comportement de manière à corréler ses tendances et son environnement, donc c'est une adaptation comportementale.

Dans ce type d'adaptation et afin de concevoir un agent adaptatif, il est nécessaire de développer une structure décisionnelle de base chargée de choisir une situation auxquelles se trouve confronté l'agent en fonction du contexte.

Pour ce là on a proposé dans ce chapitre une architecture a deux niveaux pour un agent adaptatif basé sur les RdPs, où le premier est le niveau comportements; c'est là où on présente les différents comportements possibles ainsi que la base de connaissance de l'agent sur son environnement et sur les autres agents. Le deuxième c'est le niveau méta-comportements ; c'est le niveau décisionnel de l'agent, il permet le contrôle global de comportement de l'agent.

Pour mieux clarifié cette architecture on va présenter dans le chapitre suivant une étude de cas ainsi qu'une implémentation sous un environnement de développement.

Chapitre Ⅳ

Etude De Cas Et Implémentation

IV.1 Introduction

Dans le chapitre précédent de ce mémoire, nous avons proposé une approche formelle pour l'adaptabilité d'un agent. Afin d'illustrer les différentes idées et concepts inclus dans l'architecture proposée, nous allons utiliser cette approche comme base pour une étude de cas dans un système économique. Le but est de dérouler les principaux aspects de notre approche sur un exemple concret afin de montrer la faisabilité et la mise en évidence de nos idées.

Pour cela, nous procédons comme suit : nous commençons par limiter le cadre de notre application qui est la fixation de prix dans une firme économique, Par la suite, nous décrivons brièvement la plate-forme que nous avons adoptée pour l'implémentation de notre architecture, pour montrer par la suite comment nous l'exploitons dans le cadre de notre travail. Les résultats obtenus à partir de l'implémentation de notre étude de cas sont présentés à la fin.

IV.2 L'adaptabilité et le marché financier

La nature incertaine et imprévisible du marché rend l'adaptation un concept important dans ce domaine.

Les marchés sont caractérisés par un ensemble de firmes en interaction. En effet, les firmes ne sont pas des entités isolées mais interagissent avec d'autres firmes. Cette interaction peut être : (i) directe comme dans le cas où les firmes ont des actions commerciales communes, (ii) indirecte en considérant l'effet des autres firmes sur le marché ou en observant les stratégies des firmes qui ont réussi. Pour se rapprocher plus de la réalité d'un marché compétitif, nous optons pour une interaction indirecte. Celle-ci

s'effectue à travers des échanges d'informations par l'intermédiaire du marché et à travers l'observation de l'impact des actions des différentes firmes sur ce marché.

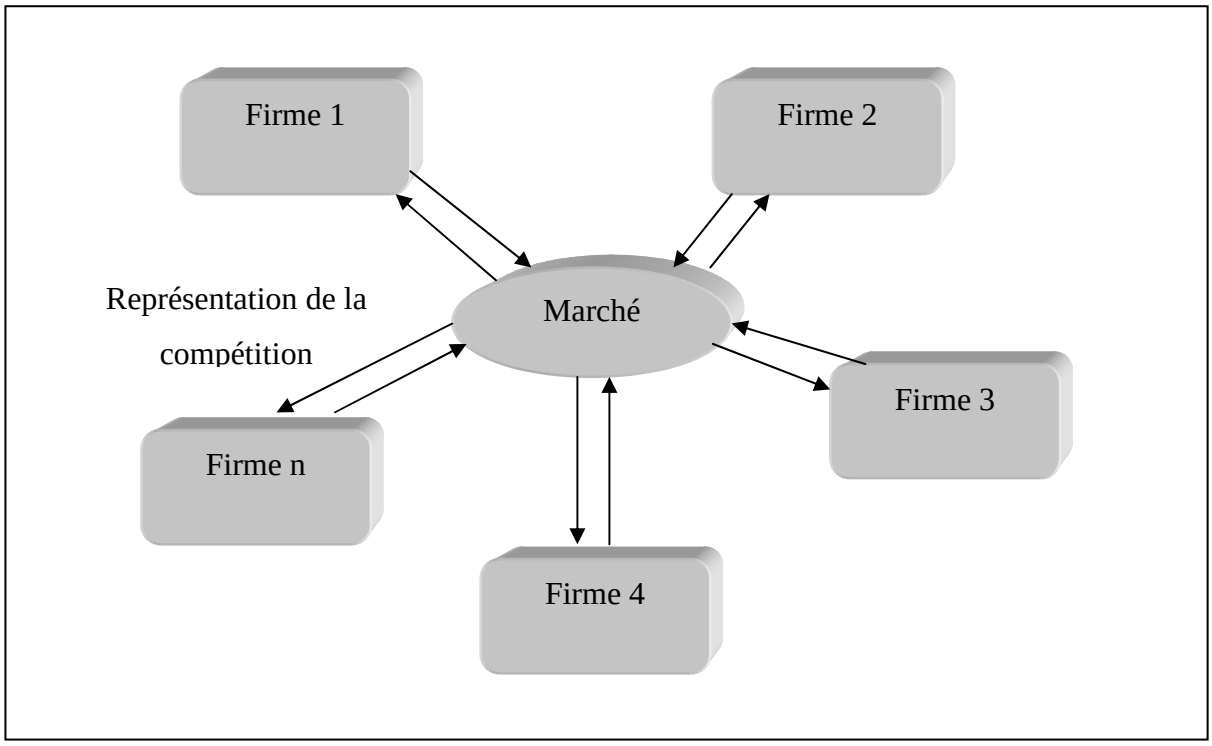


Figure IV.1 : Interaction Firmes-marché

IV.3 Principe de fixation de prix

Le prix permet d'évaluer le niveau de vente et de déterminer la rentabilité de l'entreprise. Sa fixation apparaît comme une décision importante qui dans la majorité des cas est la résultante de l'addition des différents coûts. Mais il n'est pas question d'ignorer d'autre facteur comme le consommateur et la concurrence.

IV.3.1 Les coûts

Les coûts constituent la somme des charges engagées pour tel produit ou tel fonction. On distingue :

- les coûts fixes : ils sont constants quelle que soit la production ou les ventes (charges de structure : amortissements, locations, assurances…) et les coûts

variables : ils varient en fonction de la production (matières premières, consommation d'eau, électricité, commission des vendeurs…)

- les coûts directs : ils sont directement imputables au coûts d'un produit (matières premières, salaires des ouvriers …) et les coûts indirects : ils ne sont directement imputables à aucun produit et doivent faire l'objet d'une répartition entre les différents produits (Lubrifiants, salaires du personnel administratif, frais de chauffage…)
- les coûts complets : le total des charges engagées pour la production et la distribution du produit et les coûts partiels : coûts calculés à un stade d'analyse intermédiaire par rapport au stade final de la distribution du produit (coût d'achat, coût de distribution).

IV.3.2 Le prix de vente

L'ensemble des coûts permet de définir un coût de revient c'est à dire le prix payé par l'entreprise pour la fabrication et la distribution du produit. Le coût de revient permet également de fixé le prix de vente :

PRIX DE VENTE = COUT DE REVIENT COMPLET + MARGE

La marge ajoutée au coût de revient permet de dégager un bénéfice, elle est fixé en fonction des objectifs et des contraintes. Cette marge est exprimée en pourcentage du coût de revient (taux de marge) ou du prix de vente (taux de marque).

IV.3.3 Les objectifs pour fixer un prix de vente

Voici quatre objectifs pour déterminer un prix de vente:

1) **La survie :** fixer un prix bas ; qui couvrira quelques coûts fixes et variables. La baisse des prix est du à la guerre au sein du secteur, et elle devrait pouvoir permettre à l'entreprise de survivre, puisque la concurrence est impitoyable.

   ❖ Quand l'utiliser:

   - Lorsque l'entreprise est dans une situation de surcapacité.
   - Lorsque l'entreprise est dans un évènement concurrentiel défavorable.
   - Lorsque les besoins des consommateurs ont changé.

- Lorsque le produit est désuet.

❖ Pourquoi l'utiliser:

- Pour garder en vie les activités commerciales d'une entreprise.
- Pour assurer la rotation des stocks.

❖ Conclusion:

- Objectif à court terme seulement car l'entreprise ne fait aucun profit ou très peu.

2) La maximisation de la croissance des ventes (ou prix de pénétration de marché): fixer le prix le plus bas de la concurrence, ce qui permettra de maximiser les ventes. On veut toucher une part importante du marché pour le conquérir rapidement

❖ Quand l'utiliser:

- Lorsque la demande n'achète le produit qu'au prix le plus bas; le client est sensible au prix.
- Lorsque la concurrence est très importante.
- Lorsque les consommateurs sont en grand nombre.

❖ Pourquoi l'utiliser:

- Pour décourager la concurrence actuelle ou éventuelle.
- Pour attirer la plus grande part de marché.
- Pour obtenir des économies d'échelles, des coûts réduits, et par conséquent, des profits importants

❖ Conclusion:

- L'entreprise peut utiliser cette pratique de prix à tout moment.

3) L'écrémage du marché: établir un prix élevé pour attirer un segment de marché (limité à haut pouvoir d'achat). Le plafond sera représentatif à la fois de la qualité et de la notoriété. A long terme il sera important d'avoir un avantage concurrentiel perçu et décisif (innovations)

- ❖ Quand l'utiliser:
  - Lorsque le produit est nouveau.
  - Lorsqu'il existe des consommateurs peu sensibles au prix (ils sont prêts à acheter votre produit à un prix élevé).
  - Lorsque les coûts de production sont élevés.
- ❖ Pourquoi l'utiliser:
  - Parce que l'entreprise est la seule à offrir ce produit, cette nouvelle technologie, ce nouveau service, etc.
  - Parcequ'un prix élevé projette l'image d'un produit supérieur.
- ❖ Conclusion:
  - Objectif à court et moyen termes car des concurrents vont être attirés par les profits obtenus. Cependant, lorsque les concurrents grugeront votre part de marché, vous pourrez diminuer votre prix afin d'attirer un autre segment de marché.

4) La différenciation par la marque: établir un prix élevé pour offrir un produit de luxe.

- ❖ Quand l'utiliser:
  - Lorsqu'il existe une clientèle peu sensible au prix.
  - Lorsque votre produit est fiable et de qualité, que votre service à la clientèle est efficace et que votre produit est de luxe.
- ❖ Pourquoi l'utiliser:
  - Pour être le chef de file sur la qualité du produit
- ❖ Conclusion:
  - L'entreprise peut utiliser cette pratique de prix à tout moment.

IV.3.4 La fixation de prix en fonction de la concurrence

La décision de prix devra reposer sur le trinôme : coût/demande/concurrence et s'adapter à celui-ci tout au long de la vie du produit. La fixation du prix peut-être présentée par le schéma ci-après.

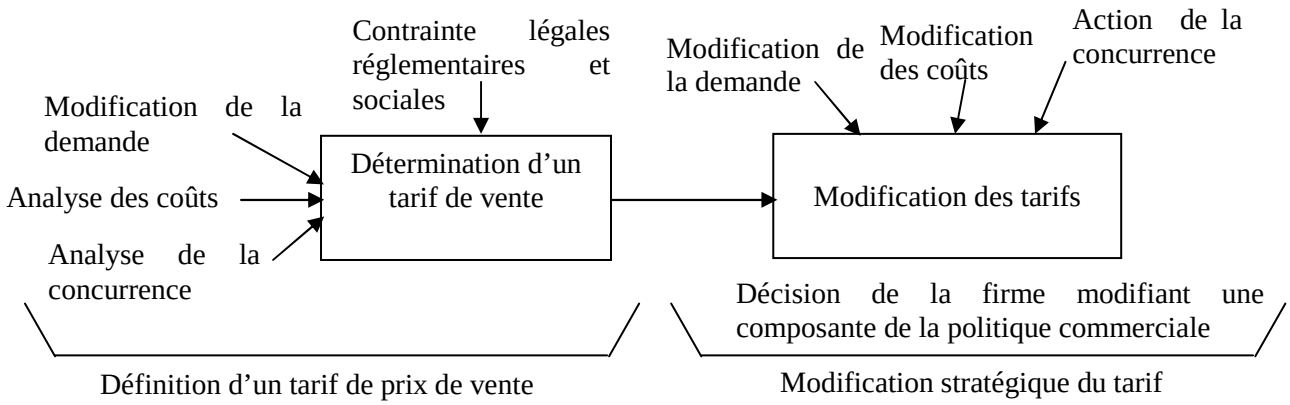


Figure IV.2 : La fixation de prix

On peut distinguer 3 attitudes dans la fixation des prix en fonction de la concurrence :

1/ Les entreprises fixent leur prix au niveau du prix moyen du marché. Elles s'alignent sur le prix pratiqué par leurs concurrents qui dominent afin d'éviter une confrontation sur les prix (guerre des prix). Cette méthode est utilisée lorsque les coûts sont mal connus, la rentabilité procurée par le prix moyen du marché est insuffisante, ou l'entreprise est en position faible (concurrence vive ou oligopole).

2/ Les entreprises leaders, qui sont en position de force sur le marché décident d'un prix inférieur ou supérieur à celui du marché. Elles choisissent un prix bas pour accroître leur part de marché ou maintiennent un prix élevé pour accroître leur profit.

3/ Les entreprises peuvent enfin choisir un prix plus élevé que la moyenne car elles pensent que le consommateur acceptera de payer ce prix pour un produit ou une entreprise auxquels il est fidèle (produit de luxe).

Le problème de la fixation des prix se pose dans les entreprises dans des circonstances multiples. On peut en dégager 7:

1. Lancement d'un produit nouveau,
2. Lancement sur un nouveau canal de distribution,
3. Lancement sur un nouveau marché,
4. Quand le produit existant franchi une étape dans son cycle de vie,
5. Quand la concurrence modifie ses prix ou lorsqu'on veut attaquer la concurrence
6. Quand les conditions économiques générales changent,
7. Quand les conditions de fixation du prix de revient du produit se modifient

IV.4 Description générale du modèle

Afin de montrer la validité, la fiabilité et l'extensibilité de notre approche, nous prenons comme exemple d'application un modèle simple de théorie des jeux appliquée à la finance.

En fonction du nombre de concurrents sur le marché et de l'importance de la demande, l'entreprise doit adapter son prix de manière à ne pas pénaliser sa part de marché tout en maximisant son profit.

L'information joue un rôle vital sur les places financières, et il est de coutume de penser que le plus le meilleur, i.e. mieux vaut s'informer le plus possible avant de prendre une position, à l'achat comme à la vente.

Les firmes les plus informés identifieront toute divergence du prix de l'action par rapport à sa valeur fondamentale et agiront en conséquence, exploitant au passage les firmes les moins informés, tout en maintenant le prix de l'action au plus près de sa valeur fondamentale. Les marchés sont dits efficients lorsque le prix reflète ainsi toute l'information disponible. Ceci conduit donc que le prix reflète en permanence toute l'information disponible.

Le scénario de notre exemple se déroule entre deux agents : la firme et le marché.

La firme veut obtenir des informations concernant la concurrence ainsi que l'état global du marché. En effet l'agent marché génère aléatoirement ces informations afin de simuler l'incertitude du marché réal,

L'agent firme doit choisir en premier lieu la stratégie adéquate à l'information obtenue ; en utilisant les méta-régles situé au niveau de méta-comportement. Par la suite le prix doit être défini et le profit calculé.

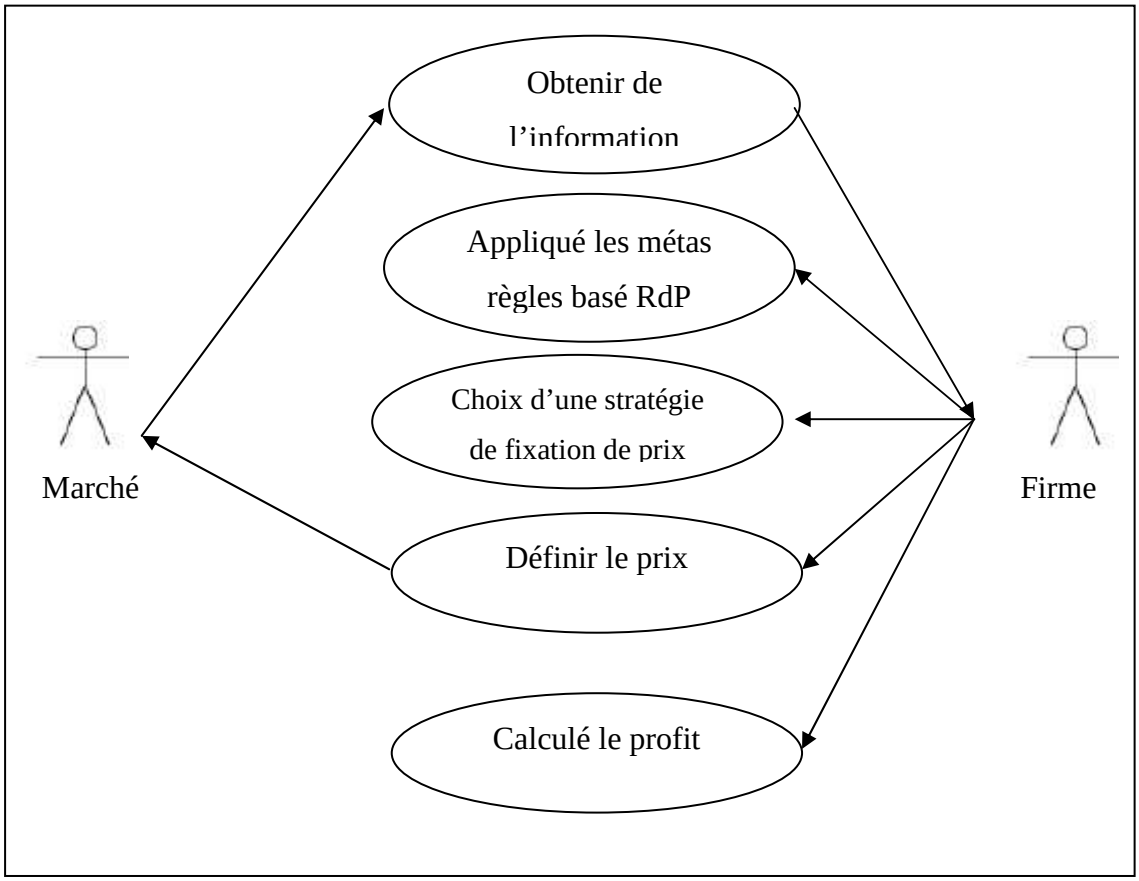


Figure IV.3 : Description générale du modèle

IV.5 Firme adaptative basé RdP

Les firmes basées sur RdP sont obtenues par l'intégration des RdPs dont l'agent représentant la firme (figure IV.4). La firme reçoit la perception du contexte a partir du marché. Il exécute le RdP pour déterminer l'action adéquate. Cette action est appliquée par la firme. Elle engendre une évaluation qui sera retournée et sauvegardée pour mettre à jour les paramètres internes de firme.

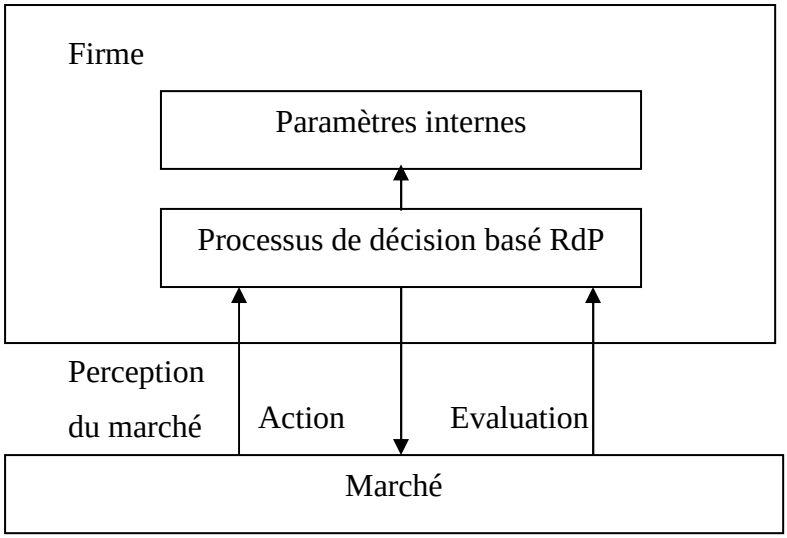


Figure IV.4 : La firme adaptative basé RdP

IV.6 Comportement de la firme adaptative

Le comportement de la firme est décrit par la figure IV.5. A chaque période, la firme commence par déterminer sa position relative sur le marché et les conditions de ce dernier en fonction de ses propres paramètres et de sa perception des performances des firmes concurrentes sur le marché (nombre de concourent, type de concurrence, prix haut du concurrence, prix bas de concurrence). L'analyse de la concurrence par la firme doit tenir compte à la fois des paramètres de concurrence obtenu par le marché et ses paramètres internes ; afin de choisir la stratégie adéquate pour la fixation de prix à proposé au client.

Cette analyse est difficile à effectuer étant donné les changements et les perturbations de leur environnement. Cette incertitude et perturbation est simulé par l'agent marché ; ce dernier responsable de la dotation de la firme des différente changement et informations des d'autre firme conçurent.

La firme adaptative met à jour ses paramètres internes tels que le nombre de concourent, type de concurrence, prix haut du concurrence, prix bas de concurrence ; et doit choisir une meilleure stratégie bien adaptée à la situation actuelle du marché pour fixer le prix en fonction de ces différentes paramètres et de méta règles présentées au niveau de son méta comportement.

Cette tâche est critique étant donné que le choix d'une stratégie qui va à l'encontre de l'évolution du marché peut être fatal pour la firme. La difficulté de cette décision émane de la complexité du marché dans lequel la firme évolue.

La firme doit donc, essayer de combiner sa perception de l'environnement avec les connaissances acquises qui sont sous forme RdP, pour raisonner sur l'état actuel de ce dernier et prendre la décision la plus adéquate.

Une fois la stratégie sélectionnée, elle est exécutée. Elle conduit ainsi, à un nouveau prix de compétition ; ce dernier va infecter bien entendu le marché ainsi un nouvel état se présente.

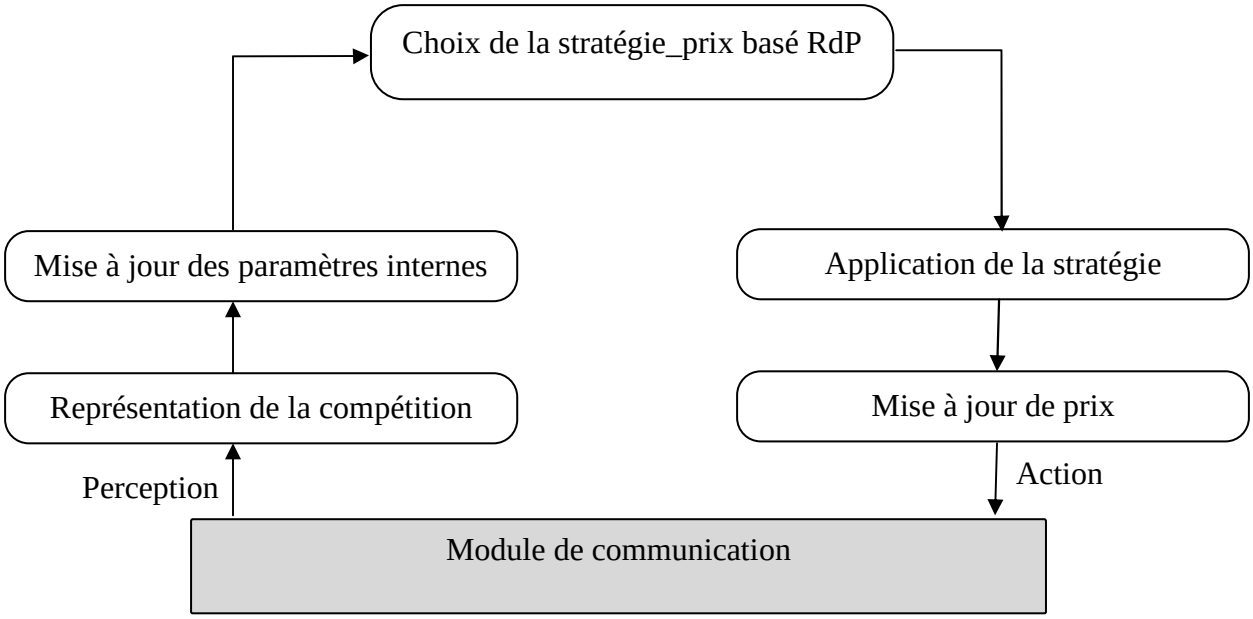


Figure IV.5: Le fonctionnement de la firme adaptative

IV.7 Stratégies de fixation de prix

Pour que la firme choisie la stratégie à appliquée pour la fixation de prix, elle doit disposer des informations suivantes :

Niveau de concurrence : c'est un paramètre qui représente le nombre de firme concurrente ; on a définit cinq niveau de concurrence donc c'est un échelon de 1 à 5.

Type de concurrence : c'est un paramètre qui représente la qualité de firme concurrente ; on a définit trois niveau descendant de concurrence (A, B, C).

Prix bas de concurrence : c'est le prix le plus bas annoncé par les firmes concurrentes.

Prix haut de concurrence : c'est le prix le plus haut annoncé par les firmes concurrentes.

Les stratégies appliquées par la firme adaptative afin de fixé le prix sont :

1. *L'écrémage :* choix le prix le plus haut.
2. *La survie :* choix le prix le plus bas.
3. *L'alignement :* fixe le prix au niveau du prix moyen du marché.

IV.8 Outils de programmation

IV.8.1 Choix du langage de programmation JAVA

Java présente un langage de programmation développé par *Sun Microsystem,* ou les premières versions sont lancées dés 1995, il a réussi à intéresser et intriguer beaucoup de développeurs à travers le monde.

Le choix du langage de programmation Java nous est en quelque sorte dicté par les contraintes d'exécution de l'application. En effet, Java permet l'écriture d'applications un peu particulières que l'on appelle appliquettes ou plus communément applets, qui s'exécutent dans un navigateur supportant Java après avoir été chargées à partir d'un serveur Web.

Les applets sont écrites dans un vrai langage de programmation et possèdent un potentiel bien plus important que toute combinaison de HTML (HyperText Markup Language), XML (eXtensible Markup Language) ou toute autre forme de scriptage. Cela donne la possibilité d'exécuter de véritables applications sur tout ordinateur disposant d'un navigateur supportant Java, et ce, sans installation préalable de logiciel sur cette même machine.

Java permet aussi de développer des applications qui sont comparables à tout autre programme écrit dans un autre langage, au détail près qu'elles nécessitent comme pour les applets, une « machine virtuelle » pour leur exécution. La machine virtuelle Java interprète des classes Java exécutables qui sont générées par la compilation du code source. L'intérêt d'une telle démarche est de permettre l'exécution d'un même programme (sans recompilation) sur n'importe quel système informatique qui possède une machine virtuelle Java, et donc, de façon indépendante du système d'exploitation.

IV.8.2 Choix de la plate forme JADE

JADE est une plate-forme multi-agents développée en entier en JAVA, créé par le laboratoire TILAB. Jade a pour but de simplifier le développement des systèmes multi-agents tout en fournissant un ensemble complet de services et d'agents conformes aux spécifications FIPA.

La plate-forme JADE inclut tous les composants obligatoires qui contrôlent un SMA. Ces composants sont l'ACC, l'AMS et le DF (voir figure IV.6).

1. **Le Système de gestion d'Agent** (**AMS**- Agent Management System)
   Agent qui exerce le contrôle de supervision sur l'accès et l'usage de la plateforme ; il est responsable d'authentifier les agents résidents et de contrôler la les enregistrements.
2. **Le Canal De communication** (**ACC**- Agent Communication Canal)
   Agent qui fournit le chemin pour les interactions de base entre les agents dans et en d'hors de la plateforme ; c'est la méthode de communication implicite qui offre un service fiable et précis pour le routage des messages ; il (l'agent) doit aussi être compatible avec le protocole IIOP () pour assurer l'interopérabilité entre les différentes plateformes.
3. **Le faciliteur d'Annuaire** (**DF**- Directory Facilitor)

   Agent qui fournit un service de pages jaunes à la plateforme.

   FIPA spécifie aussi le Langage de Communication d'agents (ACL- Agent Communication Language). La communication entre agents ne se fait que par envoi de messages seulement.

Le but de JADE est de simplifier le développement des systèmes multi-agents en conformité avec la norme FIPA pour réaliser des systèmes multi-agents interopérables. Pour atteindre ce but, JADE offre les caractéristiques suivantes :

- La plate-forme multi-agents compatible FIPA, qui inclut le **AMS**, **DF**, **ACC** – voir ci dessus. Ces trois agents sont automatiquement créés et activés à l'activation de la plate-forme.
- La plate-forme d'agents distribuée. La plate-forme d'agents peut être distribuée sur plusieurs machines, à condition qu'il n'y ait pas de pare-feu

entre ces machines. Une seule application Java (Machine Virtuelle Java) est exécutée sur chaque machine. Les agents sont implémentés comme des threads d'exécution Java et les événements Java sont utilisés pour la communication efficace et légère entre agents sur une même machine. Un agent peut exécuter des tâches parallèles et JADE planifie ces tâches d'une manière plus efficace (et même plus simple pour le programmeur) que la planification faite par la Machine Virtuelle Java pour les threads d'exécution.

- Un certain nombre de DF (Facilitateurs d'Annuaire) compatibles FIPA peuvent être activés quand on lance la plate-forme pour exécuter les applications multi-domaines, ou le domaine est logique comme décrit dans FIPA97 Part1.
- Une interface de programmation pour simplifier l'enregistrement de services d'agents avec un ou plusieurs domaines (exemple : **DF**).
- Un mécanisme de transport et une interface pour l'envoi et la réception des messages de et vers les autres agents.
- Le protocole IIOP compatible avec le document FIPA97 pour connecter les différentes plates-formes multi-agents.
- Le transport léger de messages **ACL** sur la même plate-forme d'agents. Dans le but de simplifier la transmission, les messages internes (sur la même plate-forme) sont transférés et codés comme des objets Java et non comme des chaînes de caractères. Quand l'expéditeur ou le récepteur n'appartient pas à la même plate-forme, le message est automatiquement converti en chaîne de caractères spécifiés par la FIPA. De cette façon, la conversion est cachée au programmeur d'agents, qui a seulement besoin de traiter la classe d'objets Java.
- Une bibliothèque de protocoles d'interaction compatibles FIPA.
- L'enregistrement automatique d'agents dans le Système de Gestion d'Agents (**AMS**).
- Un service d'attribution de noms compatible FIPA ; quand on lance la plate-forme, un agent obtient un identificateur unique (Globally Unique Identifier - **GUID**).

- Une interface graphique utilisateur (présenté par figure IV.6) pour gérer plusieurs agents et plates-formes multi-agents en partant d'un agent unique. L'activité de chaque plate-forme peut être supervisée et enregistrée.

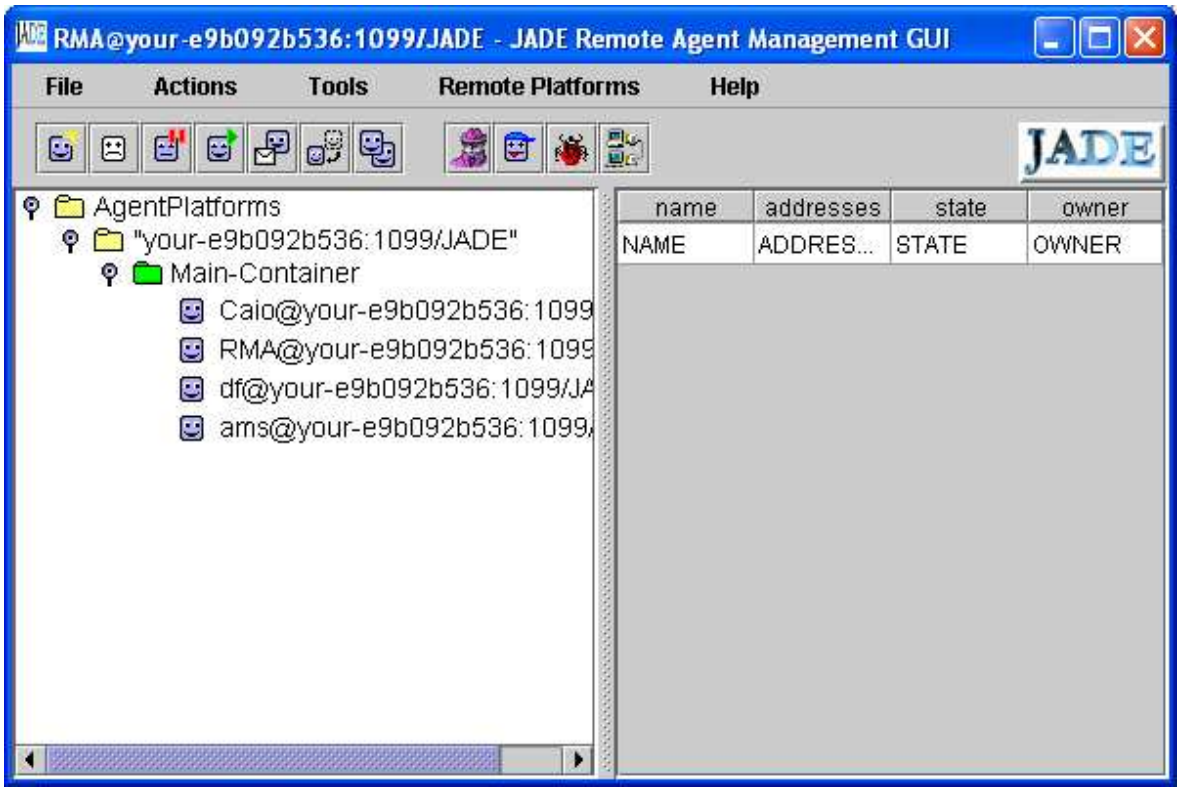


Figure IV.6 : L'interface graphique de la plate forme JADE

IV.8.3 Choix du langage de communication utilisé : *FIPA ACL*

On a besoin d'un langage de communication pour exprimer les messages échangés entre les agents de telle sorte qu'il soit compréhensible par tous les agents de la même manière (langage standard).

Dans le champ des Systèmes Multi-Agents à base d'agents cognitifs, le besoin d'un modèle de communication standard se fait sentir et les efforts se multiplient dans ce sens. Les technologies agents et multi-agents permettent de concevoir et de développer des applications complexes. La caractéristique fondamentale de celles-ci dans le paradigme actuel de l'informatique répartie est l'habileté des agents à communiquer entre eux de manière utile à leurs objectifs tant individuels que collectifs. Les spécifications de FIPA ACL, que nous allons utiliser dans notre projet se composent d'un ensemble de types de

message et de la description de leur pragmatique. Les spécifications décrivent chaque acte communicatif avec une forma narrative et une sémantique formelle basée sur la logique modale. Elles fournissent également la description normative d'un ensemble de protocole d'interaction de haut niveau, y compris la demande d'action, l'établissement de contrat (contract net) et plusieurs genres de ventes aux enchères.

Les agents communiquent entre eux de message ce qui peut représenter des actes de langages, et qui sont encodés dans le langage de communication d'agent ACL (Agent Communication Language)

IV.8.4 Liste de performatives de communication FIPA ACL

Nous listons toutes les communications possibles que FIPA ACL propose et les regroupons selon leurs actions [Fipa 00]:

Actions	**Syntaxe**	**Définition – Sens**
Accept Proposal	accept-proposal	Communication de l'accord de l'expéditeur d'effectuer une action qui lui a été préalablement soumise.
Agree	Agree	Communication de l'accord de l'expéditeur pour effectuer une action, sans doute dans le futur.
Cancel	Cancel	Communication de l'annulation de l'accord donnée préalablement par l'expéditeur pour effectuer une action.
Call for Proposal	Cfp	Communication par l'expéditeur d'une demande d'effectuer une certaine action.
Confirm	Confirm	Communication par l'expéditeur de la confirmation de la validité (selon les règles de l'agent) de la proposition préalablement reçue.
Disconfirm	Disconfirm	Communication par l'expéditeur de la confirmation de la non validité (selon les règles de l'agent) de la proposition préalablement reçue.
Failure	Failure	Communication par l'expéditeur de l'échec d'une action essayée.
Inform	Inform	Communication par l'expéditeur d'une proposition, pensée vrai par celui-ci.
Inform If	inform-if	Communication par l'expéditeur d'une proposition (pensée vrai par celui-ci), et demande au receveur une confirmation ou une non-confirmation. Macro-action impliquant l'usage de "request".
Inform Ref	inform-ref	Communication par l'expéditeur d'une demande de l'objet qui correspond à une description envoyée. Macro-action impliquant l'usage de "request".
Not Understood	not-understood	Communication par l'expéditeur d'une non compréhension d'une action effectuée par le destinataire.
Propagate	Propagate	Communication par l'expéditeur d'un message à propager à des agents dont la description est fournie. Le destinataire du message traite le sous-message à propager comme s'il lui était directement destiné et envoie le message "propate" au agent qu'il a identifié
Propose	Propose	Communication par l'expéditeur d'une proposition d'action conditionnée à certaines préconditions données.
Proxy	Proxy	Communication par l'expéditeur d'une demande d'une transmission

		d'un message à des agents dont la description est donnée.
Query Ref	query-ref	Communication par l'expéditeur d'une demande par l'expéditeur de l'objet référencé par une expression.
Refuse	Refuse	Communication par l'expéditeur de son refus d'effectuer une action donnée, et en donne les raisons.
Reject Proposal	reject-proposal	Communication, pendant une négociation, par l'expéditeur de son refus d'effectuer des actions.
Request	Request	Communication par l'expéditeur d'une demande au destinataire d'effectuer une action.
Request When	request-when	Communication par l'expéditeur d'une demande, au destinataire, d'effectuer une action quand une proposition donnée devient vrai.
Request Whenever	request-whenever	Communication par l'expéditeur d'une demande, au destinataire, d'effectuer une action dès qu'une proposition donnée devient vrai, et à chaque fois que celle-ci redevient vrai.
Subscribe	Subscribe	Communication par l'expéditeur d'une demande d'un objet donnée par une référence envoyé par l'expéditeur, et de renotifier l'agent ayant souscrit dès que l'objet en question change.

Tableau IV.1 Communication FIPA ACL

IV.8.5 Structure des messages de communication simple

Le message minimum type (syntaxe de ce message) du FIPA ACL contient tout d'abord:

- ✓ L'expéditeur du message,
- ✓ Le destinataire du message,
- ✓ Le contenu du message.

Cependant, ces messages minimums ne suffisent pas toujours pour communiquer: on peut avoir besoin, pour la compréhension du message et pour la rapidité de celle-ci ainsi que la rapidité de traitement du message, d'indiquer d'autres informations telles que:

- Le langage utilisé dans le contenu du message ("language ...") : Plusieurs langages peuvent être utilisés pour la description du contenu des messages échangés tels que :
  - ✓ Le langage KIF;
  - ✓ Le langage sémantique (SL);
  - ✓ Prologue;
  - ✓ Le langage XPDL (XML Process Markup Language);
  - ✓ Le langage XML, …

- le protocole utilisé,
- l'ontologie auquel le message se rattache ("ontology ..."),
- la référence d'un message antérieur auquel le message actuel se rattache ("in-reply-to ..."), ou la référence d'un message ultérieur attendu en retour ("reply-with ...").
- la référence de la conversation.

Exemple [Fipa 00]:

L'agent A veut informer l'agent B du temps qu'il fera demain, selon ses prévisions:

(inform
:sender (agent-identifier : name_A)
:receiver (set (agent-identifier : name_B))
:content
"weather (tomorrow, raining)"
:language Prolog)

❖ **Création de l'agent en JADE**

Pour créer un agent (par exemple agent firme), nous définissions une classe qui hérite de la classe " **jade.core.agent** "(Figure IV.7), puis nous implémentons la méthode "**Setup()**" au niveau de laquelle, nous spécifions l'ensembles des opérations qui doivent être fait lors de l'initialisation de l'agent comme par exemple l'enregistrement de l'agent dans le **DF**.

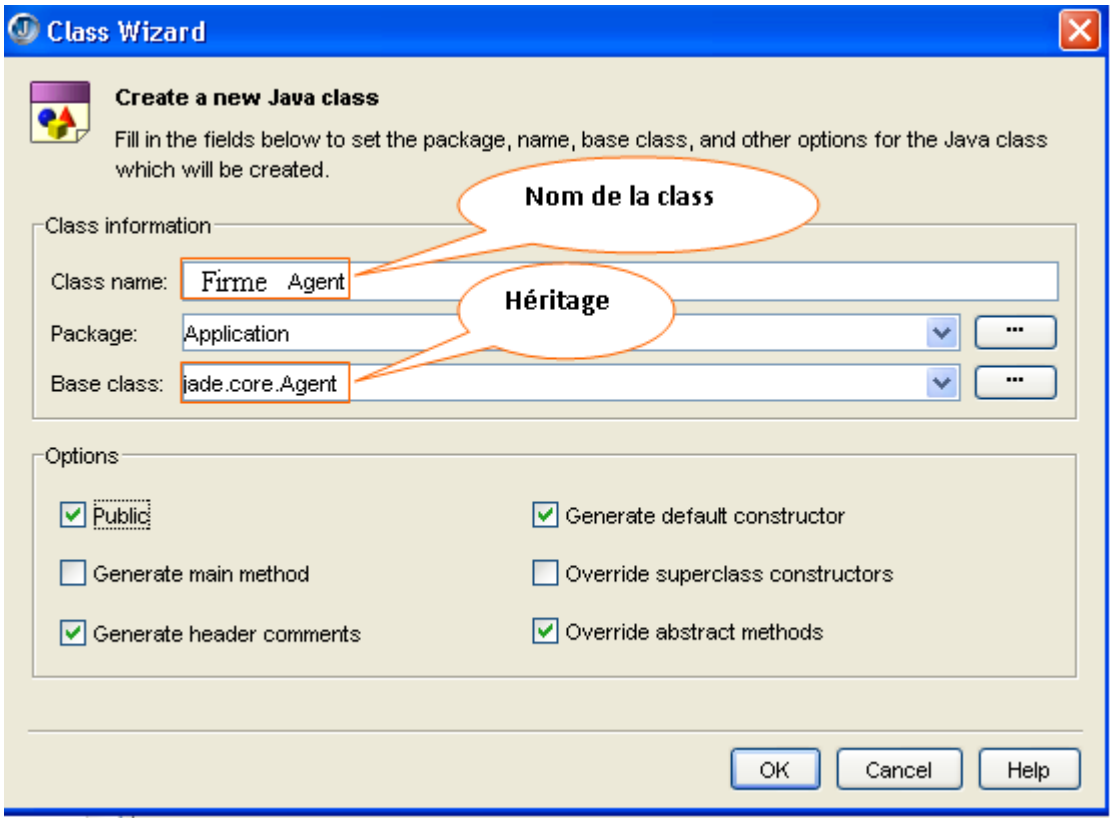


Figure IV.7: Création d'une class à partir de la super class Agent.

La figure IV.8 illustre la portion du code java qui permet de faire l'instanciation à partir de la classe **Agent** ainsi que l'implémentation de la méthode **Setup()** dans lequel l'agent s'inscrit dans le **DF** lorsqu'il rentre dans le système (initialisation de l'agent).

```
import jade.core.agent;
Public class FirmeAgent extends Agent{
      Protected void setup()   {
          DFAgentDescription dfd= new DFAgentDescription();
          Dfd.setName(getAID());
          ServiceDescription sd=new ServiceDescription();
          Sd.setName(serviceName);
          Sd.addLanguage("XML");
          Sd.setType(" la Médiation ");
          Sd.addProtocols("fipa-contract-net);
          Sd.addServices(sd);
          Try
          {  DFService.register(this, dfd);}
          Catch (FIPAException fe) {}
    } }
```

Figure IV.8 : Création d'une classe nommé FirmeAgent hérité de la classe Agent.

Notre agent nommé " FirmeAgent " étendu donc de la classe "Agent", cette dernière présentée une super-classe commune pour tous les agents qui peuvent être définis. Cela permet à l'agent d'hériter les aspects fondamentaux cachés (qui traitent les facilités fournies par la plate-forme, telles que le clonage, la communication, etc…), ainsi que qu'un ensemble de méthodes qui peuvent être appelées pour implémenter les tâches spécifiques à l'agent, comme par exemple l'envoi des messages, la suspension et la prise de certain tâches, etc…

IV.9 Résultats expérimentaux

Lorsqu'un utilisateur se connecte à notre système, la fenêtre ci-dessous s'affiche. Cette interface permet de modifier les règles à appliqués afin d'estimer le prix de micro portable ou bien de lancer l'estimation.

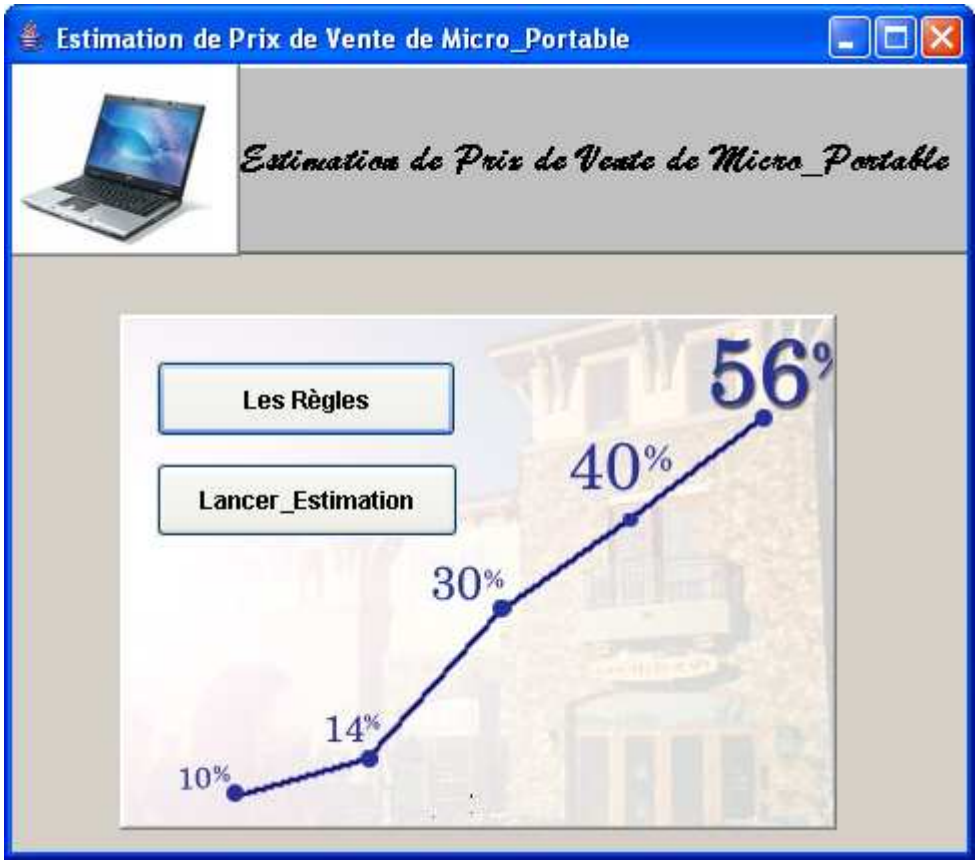


Figure IV.9: Interface graphique du logiciel d'implémentation de notre système.

Lorsque l'utilisateur choisit de modifier les règles de choix de stratégies, la fenêtre ci-dessous s'affiche.

Figure IV.10: Interface graphique modification de règles.

Et quand l'utilisateur choisit de faire l'estimation de prix l'agent firme se déclenche et donc son interface apparaitre (Figure IV.11).

Figure IV.11: Interface graphique Agent Firme.

Par la suite l'agent firme demande un état global du marché et l'agent marché va lui répondre (figure IV.12)

Figure IV.12: Interface graphique Agent marché.

IV.10 Conclusion

Nous avons décri et détaillé, le long de ce chapitre, l'implémentation de l'approche formelle proposée pour l'adaptabilité d'un agent avec une étude de cas dans les systèmes économique plus particulièrement la stratégie de fixation de prix dans une firme pour la vente des micros portables. Nous avons essayé de mettre en œuvre l'ensemble des idées qui caractérise l'architecture proposée.

Notre architecture est bien implémentée en utilisant le langage JAVA avec la plate-forme JADE, qui permet le développement des agents.

Conclusion Générale

Tout au long de ce mémoire, nous avons présenté les différents concepts nécessaires pour proposer une approche formelle pour l'adaptabilité d'un agent. Donc nous nous sommes intéressés à la technologie d'agents adaptatifs et à son utilité dans les environnements dynamiques.

L'objectif atteint par ce projet est la modélisation du principe d'adaptabilité de comportement chez l'agent par un outil formel et faire une validation.

Pour concevoir un agent adaptatif on a proposé une architecture méta niveau; afin de développé une structure décisionnelle de base ou un méta comportement chargée de choisir le comportement approprié à la situation au quelle se trouve confronté l'agent en fonction du contexte.

Ce méta comportement doit d'être doté d'un ensemble de méta règle qui sont construites graduellement. En effet on peut ajouter d'autres méta-règles que le concepteur décide leur importance au modèle étudié ou même éliminé ceux qui sont moins adoptés.

Grâce à sa modularité, l'architecture proposée aide à décomposer le comportement arbitrairement complexe d'un agent en un ensemble de petits comportements spécialisés, et éventuellement en dirigeant ces divers comportements par un méta-comportement. Les avantages de cette architecture correspondent aux différentes propriétés recherchées d'un système multi-agent :

- **Multi-granularité :** des agents de diverses granularités (taille, comportements internes, connaissance) peuvent être implémentés avec l'architecture proposée. Cette propriété est très importante pour la conception des systèmes complexes.
- **Adaptabilité et ouverture :** les agents peuvent adapter leur comportements au changement de l'environnement. De nouveau méta règles peuvent être créés et intégrés par le méta-comportement d'un agent.
- **Réflexion :** notre architecture met en application un modèle d'agents adaptatifs basé sur la réflexion dans lequel chaque agent a son propre méta-comportement qui régit ses divers comportements, par exemple afin de prendre des décisions appropriées au sujet de contrôle ou adapter ses comportements à de nouvelles circonstances.

Néanmoins, il faut noter qu'une telle architecture est souvent complexe à réaliser pour le programmeur, car il doit prendre en compte touts les cas possible (états possible de l'agent et de son environnement) et touts les comportements de base. Ce travail n'est réalisable dans de bonnes conditions que pour des agents extrêmement simple où tous les stimuli sont connus a priori et ne sont pas très nombreux.

Dans le cas des systèmes complexes, les variations de l'environnement ne peuvent être connues a priori. Pour éviter ce problème, nous proposons comme perspective de doté notre agent par des capacités d'apprentissage où l'adaptation sera traité dynamiquement.

Bibliographie

- [Abdessemed 06] Mohamed rida Abdessemed « Proposition d'une méthode de classification dans un environnement de robotique collective », Université de Batna, 2006

- [Camps et Gleizes 96] Valérie Camps, Marie-Pierre Gleizes, « Attitudes coopératives individuelles pour l'adaptation collective, Actes des quatrièmes Journées Francophones IAD&SMA, Port Camargue, Editions Hermès, Université Toulouse III – Paul Sabatier,1996

- [Chefrour 02] Chefrour D., André F., « ACEEL : modèle de composants auto-adaptatifs Application aux environnements mobiles », Journées Systèmes à composants adaptables et extensibles, 2002.

- [Drieu 01] Benjamin Drieu, « L'intelligence artificielle distribuée appliquée aux jeux d'équipe situés dans un milieu dynamique : l'exemple de la RoboCup », Université Paris 8, 2001

- [David 02] David P.-C., Ledoux T., « An Infrastructure for Adaptable Middleware », DOA'02, Springer-Verlag, LNCS 2519, 2002.

- [Erceau, 93] J. Erceau, « Intelligence Artificielle Distribuée et Systèmes Multi Agents : de la théorie aux applications », 23ème Ecole Internationale d'Informatique de l'AFCET, Neuchâtel, 1993.

- [Ferber 95] Jacques Ferber, « Les Systèmes Multi Agents: vers une intelligence collective », Inter Edition, 1995.

- [Fipa 00] FIPA, IEEE Foundation for Intelligent Physical Agents, http://www.fipa.org/ ,2000

- [Gleizes et Al 01] Marie Pierre Gleizes, Carole Bernon, Valérie Camps, Pierre Glize « La conception de systèmes multi-agents adaptatifs : contraintes et spécificités » Atelier de Méthodologie et Environnements pour les Systèmes Multi-Agents (SMA 2001), Plate-forme AFIA, Université Toulouse III – Paul Sabatier, 2001.

- [Guessoum 03] Zahia Guessoum, « Modèles et architectures d'agents et de systèmes multi-agents adaptatifs », Université Paris 6, 2003.

- [Guessoum et Al 02] Zahia Guessoum ,Thomas Meurisse et Jean-Pierre Briot « Construction modulaire d'agents et de systèmes multi-agents adaptatifs en DIMA » *Laboratoire d'informatique de Paris VI (LIP6)*2002 .

- [Hamdane 08] Hamdane Mohamed El Kamel, « Architecture basée agents pour la prise en charge de l'adaptabilité dans un système de workflow » Université Larbi Tèbessi de Tébessa, 2008

- [Hach] Hachette, Le dictionnaire universel francophone en ligne, http://www.francophonie.hachette-livre.fr

- [Jensen 97]K. Jensen: *A Brief Introduction to Coloured Petri Nets*. Lecture Notes in Computer Science Vol. 1217, Springer-Verlag 1997, 203-208.

- [Jensen 98]K. Jensen: *An Introduction to the Practical Use of Coloured Petri Nets*, Lecture Notes in Computer Science vol. 1492, Springer-Verlag 1998, 237-292.

- [Labidi et al 93] S. Labidi, W. Lejouad, « De l'Intelligence Artificielle Distribuée aux Systèmes Multi-Agents », Rapport de recherche n°2004 , 39 pages, 1993

- [Ledoux 01] Thomas Ledoux, « Projet RNTL ARCAD : D1.1-État de l'art sur l'adaptabilité », Ecole des Mines de Nantes, 2001

- [Mandiau et al 02] René Mandiau, Emmanuelle Grisling Lestrugeon. « Systèmes Multi-agents». Techniques de l'ingénieur, traité Informatique Industrielle S7216. 2002

- [Marjorie 02] Marjorie Le Bars, « Un Simulateur Multi-Agent pour l'Aide à la Décision d'un Collectif : Application à la Gestion d'une Ressource Limitée Agro-environnementale », Université Paris IX-Dauphine, 2002

- [Marcia 96], Groupe Marcia, « Auto-organisation évolution de structure(s) », Actes des 4ème journées Nationales du PRC-IA sur les Systèmes Multi-Agents, PRC IA, Toulouse, France, 1996.

- [Marcoux et Al 98] Marcoux A, Maurel C., Migeon F., Sallé P., « Generic operational decomposition for concurrent systems : semantics and reflection », Parallel and Distributed Computing Practices, vol. 1, no 4, 1998, p. 49-64, Nova Science Publishers inc.

- [Pavon 06] Juan Pavon « INGENIAS : Développement Dirigé par Modèles des Systèmes Multi-Agents », Université Pierre et Marie Curie, 2006

- [Rodin 04] Vincent Rodin, « Contribution à l'utilisation de l'informatique en biologie », Université de Rennes I, 2004

- [Soltani 07] Leila Soltani « Les Systèmes Multi-Agent pour le Contrôle de Production », Université Hadj Lakhdar Batna,2007

- [Sabas 01] Arsène Sabas, « Système multi agent : une analyse comparative des méthodologies de développement Vers la convergence des méthodologies de développement et la standardisation des plateformes SMA », Université du Québec à Trois-Rivières, 2001

- [Sébastien et Arcangeli 04] Sébastien Leriche et Jean-Paul Arcangeli , « Une architecture pour les agents mobiles adaptables » *IRIT – UPS Toulouse Cedex 4,2004*

- [Sansonnet et Fegas 05] J-P.Sansonnet et Fegas Mounir Systèmes Multi-Agents Master Recherche Informatique – LRI – Université Paris Sud XI 2005

- [Tadao 89] TADAO MURATA, Petri Nets: Properties, Analysis, and Applications, in Proceedings of the IEEE, Vol. 77, N80. 4, pages 541-580. April 1989.

- [Tagne et Al 05] Elie Fute Tagne ,Emmanuel TONYE ,César VIHO et Alain AKONO ; « Modélisation d'une Architecture Multi-agents d'un Système de Télécommunication par Réseaux de Pétri » ; 2-9525435-0 © IEEE SITIS2005

- [Tran 07] Trung Hau Tran, « Approches évolutionnaires pour le comportement adaptatif d'entités autonomes », Université Toulouse III – Paul Sabatier, 2007

- [Valette 02] R.Valette : *Les Réseaux de Petri*, LAAS-CNRS Toulouse, septembre 2002.

- [Wooldridge et al 98] Wooldridge, Jennings, Sycara, « Roadmap of Agent Research and Development ». 1998

- [Wooldridje et Jennings 95] M.Wooldridje & NR.Jennings, « Intelligent Agents : Theory and Practice »,1995.

- [Zernadji 09] Zernadji Tarek « Une approche de modélisation des logiciels à base de composants par les réseaux de Petri » Université El Hadj Lakhdar – BATNA,2009.