Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université des Sciences et de la Technologie Houari Boumediene
Faculté d'électronique et d'informatique
Département d'informatique

Rapport de TP

Module : Technologie des agents

Master 1 SII

TP

Implémentation d'un système multi-agents de ventes et achats de produits vestimentaires

- Réalisé par :

  BENHADDAD Wissam
  BOURAHLA Yasser

13-05-2018

Table des matières

Première partie

Système expert et modèle d'inférence

Chapitre 1

Introduction

1.1 Problématique et besoins

Dans le domaine de l'intelligence artificielle, nous somme souvent confrontés à la modélisation des connaissances d'un format brute(e.g langage naturel) en un format interprétable par les machines à travers un formalisme mathématique(logique), une des premières propositions fut l'introduction d'un système de règles logiques (ensembles de conditions et leurs conséquences) accompagné d'un mécanisme de déduction(à l'instar de la façon dont l'homme traite un problème) nommé **Système expert**.

FIGURE 1.1

1.2 Définitions

1.2.1 Base de connaissances

Une base de connaissance est un ensemble de connaissance modélisée de telle sorte à être compréhensibles par un ordinateur, elle peut être sous la forme d'une base de règle de productions(voir point suivant), d'une base de faits [1] ou des deux en même temps.

1.2.2 Règles de production

Les règles de production sont des formules logiques de types :
$A_1 \wedge A_2 \wedge \cdots \wedge A_n \rightarrow B$ où :

- $A_1 \wedge A_2 \wedge \cdots \wedge A_n$ Sont des formules logiques bien formée(de la logique propositionnelle ou celle des prédicats en générale), elle représentent les prémisses(conditions) d'application de la règle.
- B Représente la connaissance déductible des conditions précédentes.
- $\rightarrow$ le symbole de l'implication logique classique

Si toutes les conditions A_i sont vérifiées alors on peut ajouter la nouvelle connaissance B à la base de connaissances.

1.2.3 Moteur d'inférence

Dans un système expert, un moteur d'inférence est un module qui prend en entrée une base de connaissances BC, une base de faits BF et essaye, en appliquant soit l'algorithme de chainage avant ou bien celui du chainage arrière (tout dépendant du besoin ou la nature de la question posé par un utilisateur) de trouver un cheminement logique(i.e une succession d'application de règle de production) de telle sorte à arriver une connaissance B en particulier, ou bien à un ensemble de connaissances dérivable des faits introduit en entrée.

1.2.4 Système expert

Un système expert est un outil dont le but est d'imiter le comportement d'un véritable expert humain dans un domaine particulier, et donc de parvenir a travers l'observation de différents faits à répondre à une question ou de proposer un ensemble de solutions à un problème, c'est un outil d'aide à la décision très utile dans le domaine de l'intelligence artificielle.

1. Ensemble de connaissances que le système considère comme vraies.

Chapitre 2

Implémentation d'un système expert en Java

2.1 Outils utilisés

2.1.1 Langage de programmation :

Nous avons opté pour le langage Java, car il offre une grande flexibilité et facilite l'implémentation qui est due au fait qu'il soit totalement orienté-objet.

2.1.2 IDE :

IntelliJ Idea L'environnement de développement choisit est IntelliJ IDEA, spécialement dédié au développement en utilisant le langage Java. Il est proposé par l'entreprise JetBrains et est caractérisé par sa forte simplicité d'utilisation et les nombreux plugins et extensions qui lui sont dédiées.

2.2 Analyseur de règles de production

Afin de minimiser la modification du code, et pour des soucis de gain de temps, nous avons implémenter un mini-analyseur dont le but est de charger le contenu de deux fichiers(variables,rules) dans la base de faits d'un expert, le format adopté est inspiré des langages balisés, ce module se compose de sous modules :

2.2.1 Analyseur de variables (variableParser)

Il s'occupe de prendre en entré un fichier remplit avec des variables et leurs valeurs possible(dans langage que nous avons mis au point), toute variable est de la forme suivante :

$$<\text{VarName} : \text{Type}>\text{val}_1,\ldots,\text{val}_n,</\text{VarName}>$$

- La liste de valeurs possibles $\text{val}_1,\ldots,\text{val}_n$, peut être vide.
- VarName est le nom de la variable qui servira à l'identifier plus tard dans le programme.
- Type est une structure que nous avons conçu pour supporter entre autre les type primitifs : String,Int,Double.. mais aussi les intervalles([a,b],]a,b],$[-\infty$,b]...)

Quelques examples de variables :

- <Season : String>Summer,Winter,Autumn,Spring,</Season >
- <Temperature : Double>28.5 , 15.5,</Temperature >
- <Price : Double></Temperature >

2.2.2 Analyseur de règles (ruleParser)

Ce module prend en entré un fichier de règles écrite elles aussi dans une langage dédié, toute règle s'écrit comme suit :

$$<\text{VarResult}> = <\text{cond}_1>, \ldots <\text{cond}_n>,$$

Ou VarResult et cond_i ont la même structure suivante :

$$<\text{VarName}/\text{Type}/\text{cond}/\text{value}>$$

- VarName est le nom de la variable qui servira à l'identifier plus tard dans le programme.
- Type est le même type structuré vu dans 2.2.1
- cond est la condtion sur la valeur de la variable courante(= ,< , > != ..) value La valeur suivant le type de la variable

2.3 Modifications apportées

Le système expert de base qui nous a été demandé d'améliorer était très limité(gère les valeurs comme des chaines de caractères uniquement, conditions d'égalité seulement entre les valeurs, ..), nous avons donc décidé d'ajouter certaines fonctionnalités a ce systèmes :

2.3.1 Typage des variables

L'inconvénient en travaillant seulement avec des String est le manque de flexibilité des valeurs, nous avons donc subdivisé le typage en 4 groupes :

- StringVariableValue : Type basique, généralement la plus part des variables ont ce type, il peut désigner un catégorie, un nom, une propriété nominale ... etc.
- IntegerVariableValue : Type basique pour désigner les valeurs discrètes telle que le nombre l'age.
- DoubleVariableValue : Type basique pour désigner les valeurs continues(beaucoup plus fréquentes) telles que le prix, la température ... etc
- IntervalUnion : Type complexe pour désigner tout séquence de valeur non dénombrable comme l'union de plusieurs intervalles([0,2]∪[5,22]∪[55,69[), une valeur minimum(value>min) our maximum ... etc

2.3.2 Introduction de nouvelles conditions

Les conditions utilisables n'étant que l'égalité et l'inégalité(qui ne fonctionnait pas à la base), nous avons opté pour une restructuration du système d'évaluation, avec l'introduction du typage vu précédemment nous avons pu introduire deux nouveaux opérateur de conditions qui sont le $>$ ($\geq$) et le $<$ ($\leq$), ces deux opérateur nous ont permis ainsi de délimiter sous forme d'intervalle les valeurs de certaines variables.

2.4 Interface modifié

L'interface de base n'étant pas appropriée à notre besoin, nous avons décide de la modifier et la rendre adaptée au domaine choisi, plus de détails seront donnée dans la partie suivante

2.5 Conclusion

Les systèmes experts malgré leur simplicité ont prouvé leur efficacité durant les débuts de l'IA, cependant leur manque d'autonomie et de réaction avec le monde extérieur restait encore un obstacle majeur pour atteindre un niveau d'intelligence semblable à celui de l'homme, une extensions des systèmes expert à donc était développée, appelé système à agent-intelligents.

Deuxième partie

Système multi-agents dans le domaine commercial

Chapitre 3

Introduction

3.1 Problématique

3.2 Définitions

3.2.1 Agent intelligent

Un agent est un logiciel qui agit de façon autonome. C'est un programme qui accomplit des tâches à la manière d'un automate et en fonction de ce que lui a demandé son auteur, en revanche un **agent intelligent** est une entité autonome capable de percevoir son environnement grâce à des capteurs et aussi d'agir sur celui-ci via des effecteurs afin de réaliser des buts1. Un agent intelligent peut également apprendre ou utiliser des connaissances pour pouvoir réaliser ses objectifs.

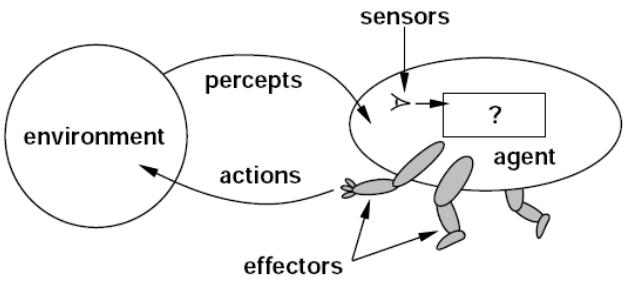

FIGURE 3.1 – Agent intelligent interagissant avec le monde extérieur

3.2.2 Système multi-agents

un système multi-agents (SMA) est un système composé d'un ensemble d'agents (un processus, un robot, un être humain, etc.), situés dans un certain environnement et interagissant selon certaines relations. Un agent est une entité caractérisée par le fait qu'elle est, au moins partiellement, autonome.

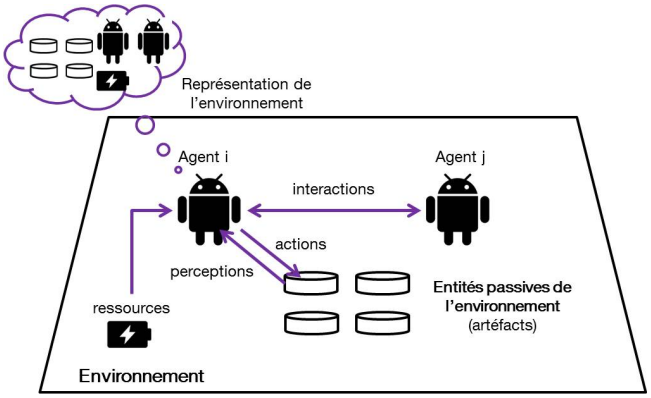

FIGURE 3.2 – Système multi-agents en coopération

Chapitre 4

Implémentation d'un système multi-agents avec la plateforme JADE

4.1 Outils utilisés

4.2 Schéma globale or samih kima habit xD

4.3 Communication entre les agents

Chapitre 5

Application dédiée

5.1 Interface homme-machine

5.2 Backend(Base de données)

Troisième partie

L'interpréteur JASON

Chapitre 6

Introduction

6.1 Problématique et besoins

6.2 Définitions

6.2.1 Architecture CDI (Croyance-Désir-Intention)

6.2.2 AgentSpeakL

6.2.3 La plateforme JASON

Chapitre 7

Exemples de communication entre agents avec JASON

7.1 Environnement de travail

7.2 Inférence locale

7.3 Scénario simple de communications

Chapitre 8

Comparaison avec la plateforme JADE

8.1 Principales différence

8.2 Le critère de décision

Table des figures

Liste des tableaux