Université Abou Bekr Belkaid
Tlemcen Algérie
جامعة أبي بكر بلقايد
تلمسان الجزائر

République Algérienne Démocratique et Populaire
Université Abou Bakr Belkaid– Tlemcen
Faculté des Sciences
Département d'Informatique

Mémoire de fin d'études

pour l'obtention du diplôme de Licence en Informatique

Thème

CONCEPTION ET IMPLÉMENTATION D'UN SYSTÈME MULTI-AGENT POUR LE TEST DE PRIMALITÉ DE NOMBRE PREMIER

Réalisé par :

- **Guermoudi Mohammed Amine**
- **Benamar Abdeladim**

Présenté le 30 Juin 2011 devant la commission d'examination composée de MM.

- *Belabed Amine* *(encadreur)*
- *Hajila Fethallah* *(Examinateur)*
- *Smahi Mohammed Ismail* *(Examinateur)*
- *Merzoug mohammed* *(examinateur)*

Année universitaire : 2010-2011

Résumé

La détection des nombres premiers est un problème difficile, notamment dans le domaine de la mathématique, car autant que le nombre est grand, il va gaspiller du temps et de nombreuses ressources matérielles et donc financières, pour cela nous avons construit un système multi-agent qui fait l'analyse et le test de la primalité des entiers pour faciliter cette tâche. Les agents du système développé sont programmés en JAVA et sous la plate forme JADE.

Abstract

The detection of prime numbers is a difficult problem, particularly in the field of mathematics, as much as the number is large, it will waste time and many financial and material resources. For that we have built a multi-agent system which makes the analysis and testing the primality of integers to facilitate this task. The agents of our system are developed in Java language under the JADE platform.

ملخص

معرفة إذا كان العدد أوليا يعتبر مشكلة صعبة في مجال الرياضيات لأنه كلما كبر العدد فانه يستغرق وقتا و مصادر مادية و مالية كبيرة, ولهذا أنشأنا نظاما متعدد العملاء يقوم بدراسة و تحليل أولية الأعداد الصحيحة لتسهيل هذه المهمة. حيث أن العملاء مبرمجون باستخدام لغة البرمجة JAVA عن طريق الأرضية JADE.

Remerciement

Nous tenons à remercier :

Allah le tous puissant ;

Mr Belabed Amine, notre encadreur, pour ses conseils, sa disponibilité et son encouragement qui nous ont permis de réaliser ce travail dans les meilleures conditions.

Les jurys pour leurs efforts et leur soin apporté à notre travail. Aux enseignants de notre université et département informatique.

Dédicace

Je dédie ce modeste travail pour mes chers parents par les quelles j'aurais jamais atteindre la place où je suis sans leurs admirables rôles et si encouragent pour accomplir mes études, si ainsi je le dédie pour tous ma familles et mes amis.

Amine

Je dédie ce mémoire :

- *A Ceux qui ont fait de moi l'homme que je suis aujourd'hui : mes très chers parents, que dieu les récompense et les garde, et surtout ma mère qui m'a éclairée mon chemin et qui m'a encouragé et soutenue toute au long de mes études.*
- *A Mon frère et ma sœur.*
- *A M. Belabed et tous les enseignants.*
- *A Tous mes amis et Mes collègues de promotion avec lesquels, j'ai passé mes meilleures années d'études.*

Abdeladim

Table des matières

Table des figures

Introduction générale

Ce projet de fin d'études rentre dans le cadre de la réalisation d'un Système multi agent (SMA) qui analyse la primalité des nombres entiers.

Le principal objectif de ce travail est d'avoir une idée théorique ainsi que pratique sur le paradigme Agents et les systèmes multi-agent. Nous avons choisi comme domaine d'application le test de primalité des nombres entiers, notre choix est justifié par l'importance des nombre premiers dans l'informatique en générale et dans la cryptographie en particulier. Le test de primalité est réalisé à travers le système multi agents que nous avons implémenté.

Nous avons choisie la plateforme JADE pour le de développement des agents et JAVA comme langage d'implémentation.

Le document est organisé comme suite :

Le premier chapitre est divisé en deux parties. La première est consacrée à un rappelle mathématique sur les nombres premiers, dont nous présentons quelques définitions, les types des nombres premiers ainsi que quelques algorithmes pour les nombres premiers et la factorisation d'un nombre en nombres premiers. Dans la deuxième partie, nous parlons de L'utilisation des Nombres premiers en informatique.

Le deuxième chapitre est divisé en trois parties. La première introduit les notions de base du concept agent. La deuxième partie, est consacrée aux systèmes multi-agents dont nous présentons une définition, les moyes et les langages de communication entre les agents ainsi que les plateformes multi-agents. La troisième partie présente quelques avantages des agents et des systèmes multi-agents.

Le troisième chapitre, est consacré à la partie réalisation et implémentation de l'application, il explique en détaille l'architecture et le fonctionnement de l'application réalisée.

La conclusion générale résume l'ensemble de notre travail, et présente quelques perspectives. L'annexe 1 introduit d'une façon plus détaillée l'architecture de la plateforme JADE.

CHAPITRE1 :les nombres premiers

I. Introduction

Les études et les recherche sur les nombres dits premiers se font par une branche des mathématiques que l'on nomme l'arithmétique. L'arithmétique fut longtemps considérée comme la partie noble des mathématiques en ce sens qu'elle était la seule à n'avoir quasiment aucune application concrète, Jusqu'au 19ème siècle [1]. Mais depuis l'arrivé de l'ordinateur au 20ème siècle, les nombres premiers ont trouvé une utilisation pratique. L'utilisation des nombres premiers et leurs propriétés sont nécessaire aujourd'hui pour tous les systèmes de cryptographie, Les codages militaires, bancaires et internet.

Le présent chapitre introduit quelques notions mathématiques relatives aux nombres premiers, leurs caractéristiques ainsi que leurs intérêts dans le domaine informatique.

II. Les nombres premiers : rappelle mathématique

II.1 Définition

Un nombre est dit premier lorsque ce nombre est un nombre entier supérieur ou égal à 2 et divisible uniquement par 1 et lui même. De ce fait, 2 est le seul nombre premier pair.

Euclide donne la définition des nombres premiers, la preuve de leur infinitude dans une démonstration restée célèbre. Il propose aussi la définition du plus grand commun diviseur (PGCD) plus petit commun multiple (PPCM),, et les algorithmes pour les déterminer, aujourd'hui appelés algorithmes d'Euclide.

Le célèbre mathématicien grec Euclide (IIIe av. JC), donne dans le livre VII de ses "Éléments" les définitions d'une unité, d'un nombre, d'un nombre pair et d'un nombre premier [2].

- Définition 1 : L'unité est ce selon quoi chacune des choses existantes est dite une.
- Définition 2 : Un nombre est un assemblage composé d'unités.

- Définition 6 : Le nombre pair est celui qui peut se partager en deux parties égales.
- Définition 12 : Le nombre premier est celui qui est mesuré par l'unité seule.
- Définition 14 : Le nombre composé est celui qui est mesuré par quelque nombre.

II.2 Quelques types de nombres premiers

a) les nombres premiers jumeaux

Deux nombres premiers jumeaux sont deux nombres premiers qui ne diffèrent que de 2. Hormis pour la paire (2, 3), cette distance de 2 est la plus petite distance possible entre deux nombres premiers. Les plus petits nombres premiers jumeaux sont 3 et 5, 5 et 7, 11 et 13.

Si (a, b) un couple de nombres entiers tel que a et b soient tous les deux des nombres premiers, et a < b. On dit que (a,b) forme un couple de nombres premiers jumeaux si b = a + 2.

Il est possible de démontrer que, pour tout entier $m \geq 2$, le couple (m, m + 2) est constitué de nombres premiers jumeaux si et seulement si $4[(m-1)!+1]+m \equiv 0 \mod m(m+2)$.

Cette caractérisation modulaire et factorielle des nombres premiers jumeaux a été découverte par P. A. Clement en 1949 [3]

b) les nombres premiers de Chen

Un nombre premier p est appelé un nombre premier de Chen si p + 2 est soit un nombre premier soit un nombre semi-premier. (C'est-à-dire, si $\Omega(p+2) \leq 2$, où Ωest la fonction grand oméga). En 1966, Chen Jingrun démontra qu'il existe une infinité de tels nombres premiers. Et les premiers nombres premiers de Chen sont :

2, **3**, **5**, **7**, **11**, **13**, **17**, **19**, **23**, **29**, **31**, **37**, **41**, **47**, **53**, **59**, **67**, **71**, **83**, **89**, **101**

Le plus petit membre d'une paire de nombres premiers jumeaux est toujours un nombre premier de Chen. En 2005, le plus grand nombre premier jumeau connu est :

$16869987339975 \times 2^{171960} \pm 1$ [4]

c) Nombres premiers cousins

les nombres premiers cousins sont une paire de nombres premiers qui diffèrent de quatre ; comparer ceci avec les nombres premiers jumeaux, les paires de nombres premiers qui diffèrent de deux, Il découle de la première Conjecture de Hardy-Littlewood que les nombres premiers cousins ont la même densité asymptotique que les nombres premiers jumeaux. [5]

Exemple de nombres premiers de cousin inferieur a 100:

(3, 7), (7, 11), (13, 17), (19, 23), (37, 41), (43, 47), (67, 71), (79, 83)

d) Autres types

- **Auto premier**

Premier ne pouvant pas s'écrire sous la forme d'un nombre ajouté à la somme des chiffres de ce nombre. En base 10 : {3, 5, 7, 31, 53, 97, 211, 233, 277, 367, 389, 457, 479}

- **Mersenne**

Premier de la forme 2^n - 1. Remarque : Dans leur représentation en base 2, ils sont des répunits.

{3, 7, 31, 127, 8191, 131071, 524287, 2147483647, 2305843009213693951, 618970019642690137449562111, 162259276829213363391578010288127, 170141183460469231731687303715884105727}

- **Nombre de Carol**

Premier de la forme $(2^n - 1)^2 - 2$: {7, 47, 223, 3967, 16127, 1046527, 16769023, 1073676287, 68718952447}

- **Euclide**

Premier de la forme p_n+ 1. : {3, 7, 31, 211, 2311}

- **Factoriel**

Premier de la forme $n!$ - 1 ou $n!$ + :

{1.2, 3, 5, 7, 23, 719, 5039, 39916801, 479001599, 87178291199}

- **Fibonacci**

Premiers dans la suite de Fibonacci.

{2, 3, 5, 13, 89, 233, 1597, 28657, 514229, 433494437, 2971215073}

- **Gauss**

Entier de Gauss premier. :

{3, 7, 11, 19, 23, 31, 43, 47, 59, 67, 71, 79, 83, 103, 107, 127, 131, 139, 151, 163, 167, 179, 191, 199, 211, 223, 227, 239, 251, 263, 271, 283, 307, 311, 331, 347, 359, 367, 379, 383, 419, 431, 439, 443, 463, 467, 479, 487, 491, 499}

Et il ya autre listes par exemples **:: Bell ; Carré centré ;Chanceux ;Chen ; Cousins ; Cubain ;Cullen ; Double de Mersenne ;Eisenstein ; Étoilé ; Équilibré ; Mills ; Markov….**

II.3 Quelques algorithmes de teste de primalité

a) Test de Fermat

la théorème de Fermat montrer que si p est un nombre premier et si a est premier avec p alors $a^{p-1}-1$ est divisible par **p** ,ceci peut être écrit comme suite : $a^{p-1} \equiv 1 \ (p)$ [4]

Algorithme [6] :

```
tirer au hasard a tel que 1 < a < n
 si pgcd (a; n)≠  1 : (a facteur de n)
→ n pas premier
Sinon, si a^(n-1)  1 |≡[n]
→ n pas premier
 Sinon,
→ n peut-être premier ...
```

b) Test par divisions successives

La méthode la plus simple pour tester la primauté d'un nombre N est de tester sa divisibilité par tous les nombres compris entre 2 et la racine carrée de N.

```
EstPremier(x) =

{

// Inférieur à 2 : pas premier

Si (x<2) Retourne(FAUX);

// Egal 2 : premier

Si (x == 2) Retourne(VRAI);

// Nombre pair autre que 2 : n'est pas premier

Si ((x % 2) == 0) Retourne(FAUX);
// Teste tous les diviseurs inférieur au nombre (de 2 à x-
1)
i = 3;
Tantque(i*i <= x)
{
// Si on peut diviser ce nombre : il n'est pas premier
Si ((x % i) == 0) Retourne(FAUX);
// Passe au prochain diviseur
i += 2;
};
// Aucun nombre compris entre 2 et Racine(x)
// ne le divise : il est premier
Retourne(VRAI);
```

Le problème avec cet algorithme est qu'il est très long car il doit faire énormément de divisions ... Pensons à un nombre ayant une centaine de chiffres. On utilise plutôt cet algorithme

pour vérifier rapidement qu'un nombre est composé en testant sa divisibilité par les nombres entre 1 et 1000

c) Algorithme de Miller-Rabin

Le test de primalité de Miller-Rabin est un test de primalité probabiliste : c'est-à-dire un algorithme qui détermine si un nombre donné est probablement premier, de façon similaire au test de primalité de Fermat

On note $n-1 = 2^s \times d$

Théorème

Si n et premier et a tel que $a^n = 1$, alors :

soit

$a^d \equiv 1[n]$

soit il existe $r \in \{1,\ldots, s-1\}$

$a^{2rs} \equiv -i[n]$

Algorithme **:[6]**

```
Calculer d et s
 tirer au hasard a tel que 1 < a < n
   si pgcd(a, n) ≠1 : (a facteur de n)
→ n pas premier
  sinon, si a^d |≡1 [n] et∀ r ∈{0, 1,…., s − 1},
a^2rs |≡−1 [n]
→ n pas premier
   sinon,
→ n peut-être premier ...
```

II.4 la factorisation d'un nombre en nombres premiers

a) Définition

Consiste à chercher à écrire un entier supérieur ou égal à 2 sous forme d'un produit de nombres premiers . Par définition, un nombre premier ne peut pas être décomposé en produit de plusieurs nombres premiers. On peut dire qu'il est sa propre décomposition.

b) exemples d'algorithmes de factorisation

- **Algorithme naïf**

Il s'agit tout simplement de balayer la liste des nombres premiers en testant si le nombre premier p divise n. Si oui, on recommence l'algorithme pour n/p, en ne testant que les diviseurs premiers encore envisageables. On s'arrête quand le nombre premier à tester devient supérieur à la racine carrée du nombre qu'il est censé diviser.

Exemple :

2088	2	2 divise 2088 le quotient est 1044
1044	2	2 divise 1044, le quotient est 522
522	2	2 divise 522, le quotient est 261
261	3	2 ne divise pas 261, mais 3 divise 261 et le quotient est 87
87	3	3 divise 87 et le quotient est 29
29		ni 3, ni 5 ne divisent 29 et 7^2 est plus grand que 29 (fin)

$2088 = 2^3 \times 3^2 \times 29$

- **Algorithme de p-1 de pollard**

C'est un algorithme pour la factorisation des nombres premiers conçu par « jhon pollard » en 1974.

L'algorithme de base peut être écrit de la façon suivante :

Entrées : ***n*** : un entier composé

Sortie : un facteur non-trivial de ***n*** ou un échec

1. sélectionner un seuil de friabilité ***B***
2. prendre un ***a*** aléatoirement dans $(\mathbb{Z}/n\mathbb{Z})^*$ (note : nous pouvons d'ore et déjà fixer ***a***, une sélection aléatoire ici n'est pas impérative)
3. pour chaque nombre premier $q \leq B$

   $e \leftarrow \left\lfloor \frac{\log B}{\log q} \right\rfloor$

   $a \leftarrow a^{q^e} \mod n$

   (à la fin de cette boucle, on a a^M)
4. $g \leftarrow \mathrm{pgcd}(a-1, n)$

5. si $\mathbf{1} < \boldsymbol{g} < \boldsymbol{n}$ alors retourner $\boldsymbol{g}$
6. si $\boldsymbol{g} = \mathbf{1}$ alors sélectionner un $\boldsymbol{B}$ plus grand et aller à l'étape **2** ou retourner échec
7. si $\boldsymbol{g} = \boldsymbol{n}$ alors aller à l'étape **2** ou retourner échec

Si $\boldsymbol{g} = \mathbf{1}$ dans l'étape **6**, ceci indique que pour tous les $\boldsymbol{p} - \mathbf{1}$ il n'y en a aucun qui était $\boldsymbol{B}$-superfriable. Si $\boldsymbol{g} = \boldsymbol{n}$ dans l'étape **7**, cela indique généralement que tous les facteurs étaient $\boldsymbol{B}$-superfriables, mais dans de rares cas, il pourrait indiquer que $\boldsymbol{a}$ possède un petit ordre modulo $\boldsymbol{p}$. [7].

III. L'utilisation des nombres premiers dans l'informatique

Les nombres premiers sont particulièrement importants en arithmétique, la branche des mathématiques qui traite des nombres entiers.

Les nombres premiers, et la théorie des nombres en particulier, ont longtemps été vus comme un sujet purement mathématique, avec peu ou pas d'applications extérieures. Cela changea d'un seul coup dans les années 1970, quand des nouveaux systèmes de cryptographie basés sur les propriétés des nombres premiers furent découverts.

Ils font également l'objet d'une actualité brûlante dans les nouvelles technologies, en particulier dans la cryptographie, et le codage des informations.

Avec le développement d'internet, le besoin de transmettre des informations confidentielles de façon sécurisée, par exemple des numéros de carte bancaire, est en effet devenu primordial... C'est là notamment qu'intervient l'algorithme RSA, un algorithme de cryptographie basé sur une propriété simple des nombres premiers.

La section suivante présente un bref résumé sur les principes de la cryptographie, et introduit plus en détaille l'algorithme RSA.

III.1 Principes de la cryptologie

La cryptologie, science du secret, englobe la cryptographie — le codage secret d'un message — et la cryptanalyse — le décodage d'un message codé. La cryptologie est une technique très ancienne. Ainsi, Jules César utilisait déjà un algorithme que nous appelons aujourd'hui le chiffrement par substitution, qui consiste à décaler d'une valeur constante les lettres dans l'ordre alphabétique. Mais la cryptologie est aussi une science qui se renouvelle. Depuis les années 1970, elle est devenue un thème de recherche scientifique académique.

Le principe de la cryptographie est de définir une transformation des symboles d'un langage (les lettres ou les mots par exemple) qui soit difficilement inversible, de telle sorte que retrouver le mot original à partir du mot codé devienne une opération difficile à effectuer.

Il existe deux grandes familles d'algorithmes de cryptographie : *les algorithmes symétriques (à clé secrète)* et *les algorithmes asymétriques (à clé publique)*. La clé, en cryptographie symétrique, est l'information qui permet de coder (on dit aussi chiffrer) et de décoder un message. Ainsi, l'algorithme de décalage des caractères utilisé par Jules César est un algorithme à clé privée dont la clé est l'algorithme de codage : si l'on sait que les lettres utilisées ont été décalées d'une valeur constante (c'est-à-dire chaque lettre est remplacée par la n-ième lettre après dans l'ordre alphabétique, avec la convention de bouclage qui dit que la lettre suivant le z est le a) alors on sait aisément décoder le message.

La machine Enigma qui fut utilisée par les Allemands durant la seconde guerre mondiale était également basée sur les substitutions, mais avec un mécanisme beaucoup plus sophistique.

Le principe de la cryptographie asymétrique (ou à clé publique) est basé sur l'existence d'une fonction dite à sens unique, pour transformer un message en message codé. Il faut que cette fonction soit simple à appliquer à un quelconque message, mais qu'il soit difficile de retrouver le message original à partir du message codé. La cryptographie à clé publique permet de coder un message secret et aussi d'authentifier l'émetteur d'un message. [8][9]

III.2 Système RSA

Le protocole de cryptographie RSA se fonde sur le résultat suivant :

Soit p et q deux nombres premiers.

On pose **n = pq.**

Si **e** est un nombres entier premier avec **(p-1)(q-1)** ,alors il existe un nombre entier **d** . Positif, telle que, pour tout **A, A^{ed} –A** soit divisible par **n**

(**X** et **Y** sont dits premiers entre eux si leur seul diviseur commun est **1**).

Création des clés :

1. Choisir p et q, deux nombres premiers distincts

2. Noter n leur produit, appelé « module de chiffrement » : $n=pq$
3. Calculer l'indicatrice d'Euler de n : $\varphi(n)=(p-1)(q-1)$.
4. Choisir e, un entier premier avec $\varphi(n)$, appelé « exposant de chiffrement ».
5. Comme e est premier avec $\varphi(n)$, il est, d'après le théorème de Bachet-Bézout[9], inversible $\bmod \varphi(n)$, c'est-à-dire qu'il existe un entier d tel que $ed \equiv 1 \pmod{\varphi(n)}$.

L'entier d est l'exposant de déchiffrement. Le couple (n,e) est appelé clé publique, alors que le couple (n,d) est appelé clé privée.[6]

IV. Conclusion

Dans ce chapitre nous avons présenté une vue générale sur les nombres premiers, leurs principales caractéristiques ainsi que quelques algorithmes relatifs aux testes de primalité et la factorisation. Le chapitre a montré également l'utilisation pratique de ces nombres dans le domaine informatique et plus précisément dans la cryptographie à clé publique.

Chapitre2 : système multi-agent

I. Introduction

De nos jours, le mot « agent » est utilisé dans plusieurs domaines et, de ce fait, plusieurs sens lui sont attachés. D'ailleurs, même à l'intérieur du domaine de l'informatique, plusieurs chercheurs ont défini le concept d'agent de manières différentes. Comme ce concept est fondamental pour la suite de ce chapitre, nous allons nous attarder à bien le cerner tout au long de ce chapitre. La plupart du temps, un agent n'est pas seul dans son environnement, il y a d'autres agents présents autour de lui. Les agents doivent, par conséquent, être capables d'interagir entre eux.

Un système où évoluent plusieurs agents est appelé système multi-agent et il possède généralement plusieurs caractéristiques intéressantes, comme le parallélisme, la robustesse et l'extensibilité. Ce chapitre se veut une introduction au concept d'agent pour établir les bases de ce mémoire. Il débute par une définition du concept d'agent. Il présente par la suite quelques architectures d'agents. Il introduit finalement les systèmes multi-agents brièvement, car ils feront l'objet d'une présentation plus approfondie dans les chapitres suivants.

II. La notion d'agent

II.1 Définition

Un agent peut être défini comme entité capable d'agir sur elle-même et son environnement. Disposant d'une représentation partielle de cet environnement. Pouvant communiquer avec d'autres agent et dont le comportement est la conséquence de ses observation .de sa connaissance et des interactions avec les autres agents. Pour Weiss (1999), un **agent** est une "entité computationnelle", comme un programme informatique ou un robot, qui peut être vue comme percevant et agissant de façon autonome sur son environnement. On peut parler d'autonomie parce que son comportement dépend au moins partiellement de son expérience. [10] donc l'agent est :

Entité (physique ou abstraite) caractérisée par :

- Son autonomie dans la prise de décision,
- Ses connaissances sur lui même et sur les autres,
- Sa capacité d'agir.

✓ ***Remarque*** :

Un agent ne peut exister sans environnement. L'environnement est une structure dans laquelle l'agent évolue. Un agent va agir sur son environnement et l'environnement va agir sur l'agent.

✓ **Les caractéristiques d'un agent**

Agent est une entité qui possède les caractéristiques suivantes **[JW, 1999]**:[11]

- **Autonome :** il prend des décisions motivées par son état interne sans intervention extérieure.
- **Réactif :** Un agent est situé dans un environnement. Il est capable de percevoir cet environnement et de réagir aux changements qui interviennent par ses actions.
- **Social :** il est capable d'interagir avec d'autres agents.
- **Proactif :** il ne fait pas que réagir à son environnement mais il est capable de lui-même de produire des actions motivées par des buts.

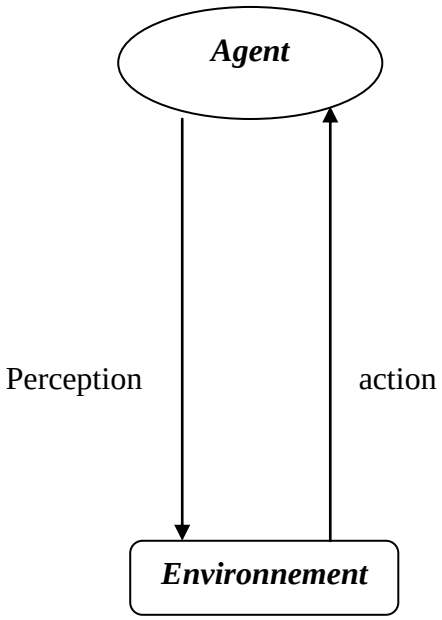

Figure 1 : « ***Un Agent avec son environnement*** **»**

II.2 Les types d'agents [12]

- *Agent cognitif* :

  Il est intelligent par lui-même c'est-à-dire qu'il effectue un certain raisonnement pour choisir ses actions. Un tel raisonnement peut se faire soit en se basant sur les buts de l'agent, soit sur une certaine fonction d'utilité.

- *Agent réactif :*

  Comme son nom l'indique, un agent réactif ne fait que réagir aux changements qui surviennent dans l'environnement. Autrement dit, un tel agent se contente simplement d'acquérir des perceptions et de réagir à celles-ci en appliquant certaines règles prédéfinies.

  Étant donné qu'il n'y a pratiquement pas de raisonnement, ces agents peuvent agir et réagir très rapidement.

- *Agent hybride* :

  Chaque agent hybride est caractérisé par la notion de couches et chaque couche représente soit les agents cognitifs, soit les agents réactifs. Donc l'agent hybride combine entre les deux comportements (comportement réactif et comportement cognitif).

III. Les systèmes multi-agent

III.1 Définition

La définition d'un système multi-agent (avec son acronyme SMA, et MAS pour

« Multi-agent system » en anglais) est plus immédiate : « un système multi-agent est un ensemble organisé d'agents ». Nous ne faisons que suivre ici la définition usuelle du terme système : « un ensemble organisé d'éléments ». Cela signifie que dans un système multi-agent, il existe une ou plusieurs organisations qui structurent les règles de cohabitation et de travail collectif entre agents. Un SMA est défini comme un système dans lequel plusieurs entités (ou agents) autonomes et intelligentes interagissent ensemble dans le but de réaliser un ensemble de buts ou de tâches [13]

Les caractéristiques d'un SMA :

Le SMA est un système et tous les systèmes ont des caractéristiques et des propriétés. Un SMA a les caractéristiques suivantes :

Ouvert : les agents y entrent et en sortent librement (ex: une application de commerce électronique, etc.).

Fermé : l'ensemble d'agents reste le même.

Homogène : tous les agents sont construits sur le même modèle.

Hétérogène : des agents de modèles différents, de granularités différentes.

Mixte (ou non) : les agents « humains » sont partie intégrante du système, comme le serait un groupe de travail représenté par des agents assistants (implique ouvert et hétérogène).

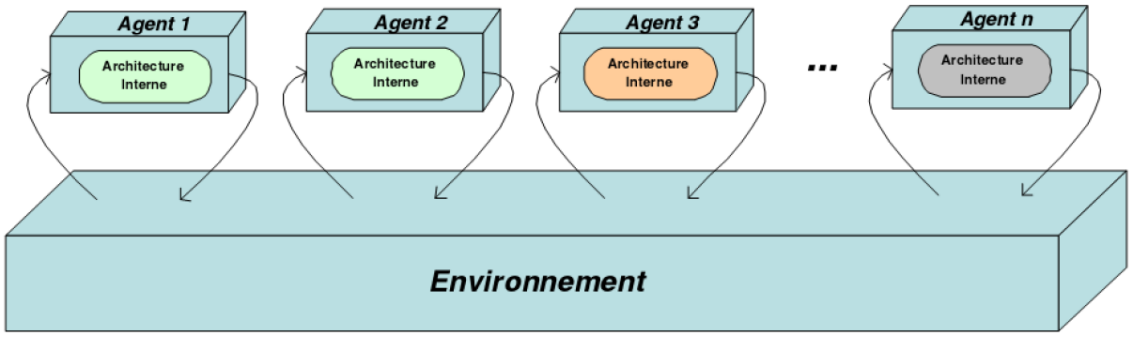

Figure 2 : « *Architecture d'un SMA »[15]*

III.2 Communication et langage de communication entre les agents

La communication est la base de la résolution coopérative des problèmes. Elle permet de synchroniser les actions des agents et résoudre les conflits de ressources et de buts par la négociation. Dans les systèmes multi-agents, les agents ne disposent d'aucune mémoire commune. La communication entre les agents repose explicitement sur des mécanismes d'envoi de message, de réception et de synchronisation. [12]

a) Architecture de communication :

La communication entre les agents dans les SMA peut être organisée suivant trois schémas différents :

- **Réseaux en anneau :** Les interactions dans ce genre d'organisations ont trop lentes, un message destiné à un agent doit transiter par n-1 agents (figure 3) :

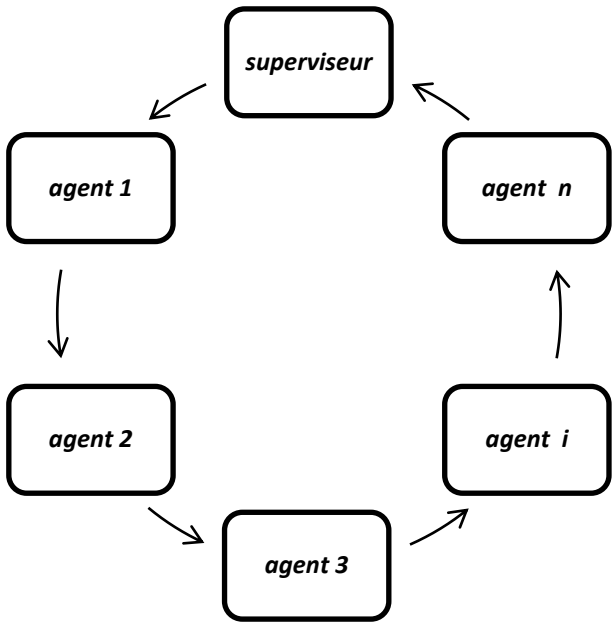

Figure 3 : «*Réseau en anneau* »

- **Réseaux en étoiles :** Ils présentent l'avantage de l'accès rapide entre le superviseur et les autres agents. La nature bidirectionnelle des connexions rend complexe la gestion des interactions inter-agents (figure 4).

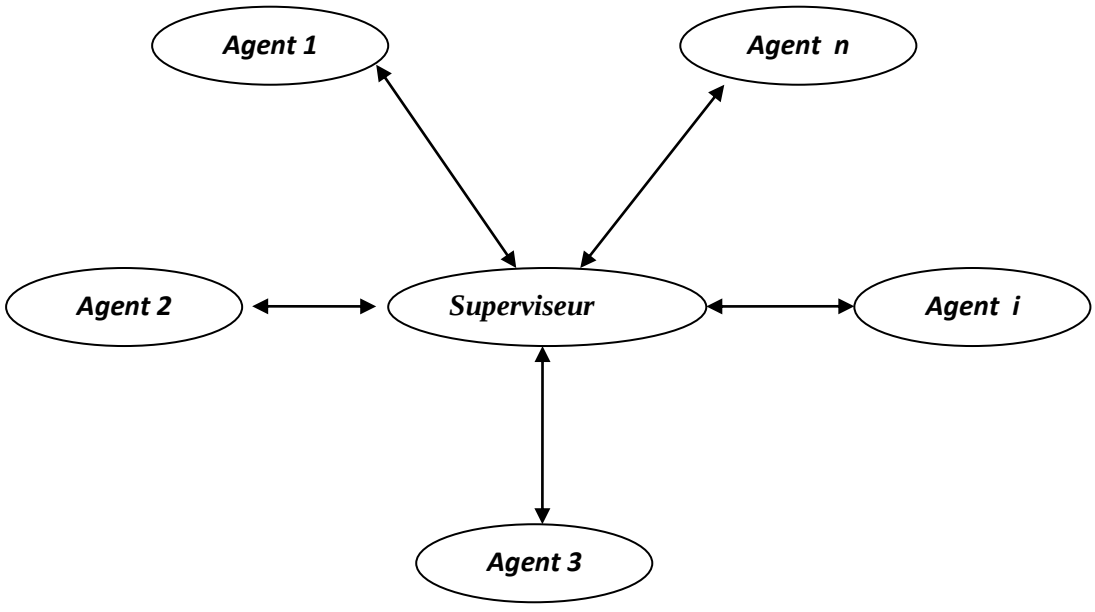

Figure 4 : «*Réseau en étoile* »

- **Réseaux hybrides :** Ce type d'organisation est une solution hybride reliant deux types d'organisation : un réseau en bus et un réseau en étoile (figure 5).

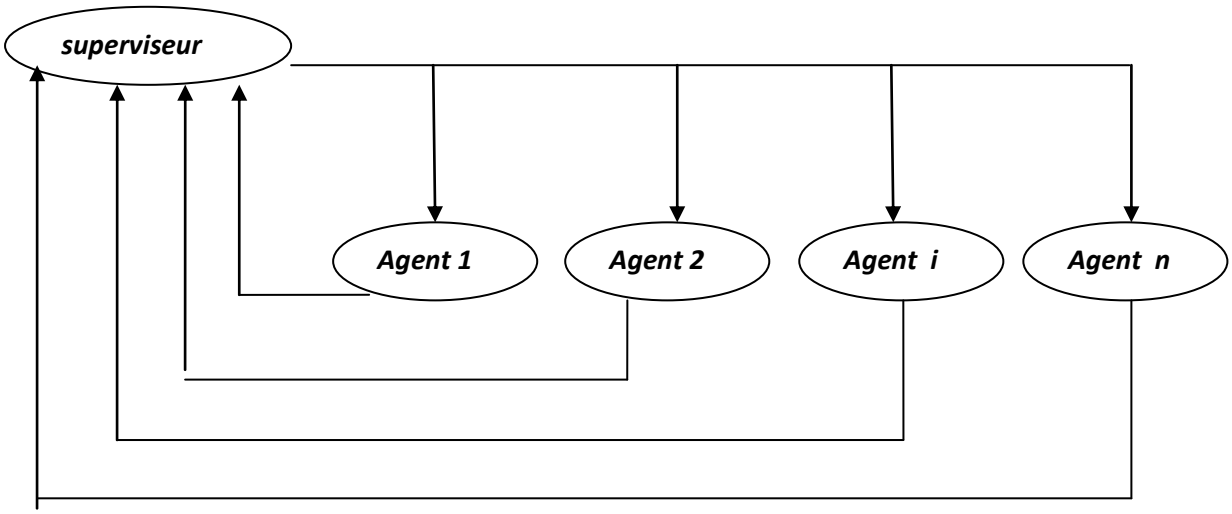

Figure 5 : « *Réseau hybride* »

b) Mode de communication

- **Communication par partage d'informations**

Les composants ne sont pas en liaison directe mais communiquent via une structure de données partagée, ou on trouve les connaissances relatives à la résolution (état courant du problème) qui évolue durant le processus d'exécution. Cette manière de communiquer est l'une des plus utilisées dans la conception des systèmes multi-experts. L'exemple parfait d'utilisation de ce mode de communication est l'architecture de blackboard (tableau noir) , on parle plutôt de sources de connaissances que d'agents. Ce mode de communication n'existe pas dans les systèmes multi-agent où l'on dispose que d'une vision partielle du système alors que la communication par partage d'informations suppose l'existence d'une base partagée sur laquelle les composants viennent lire et écrire.[14]

Communication via le tableau noir : l'architecture tableau noir se compose des trois éléments suivants:

- Les connaissances
- Le tableau noir
- Le mécanisme de contrôle

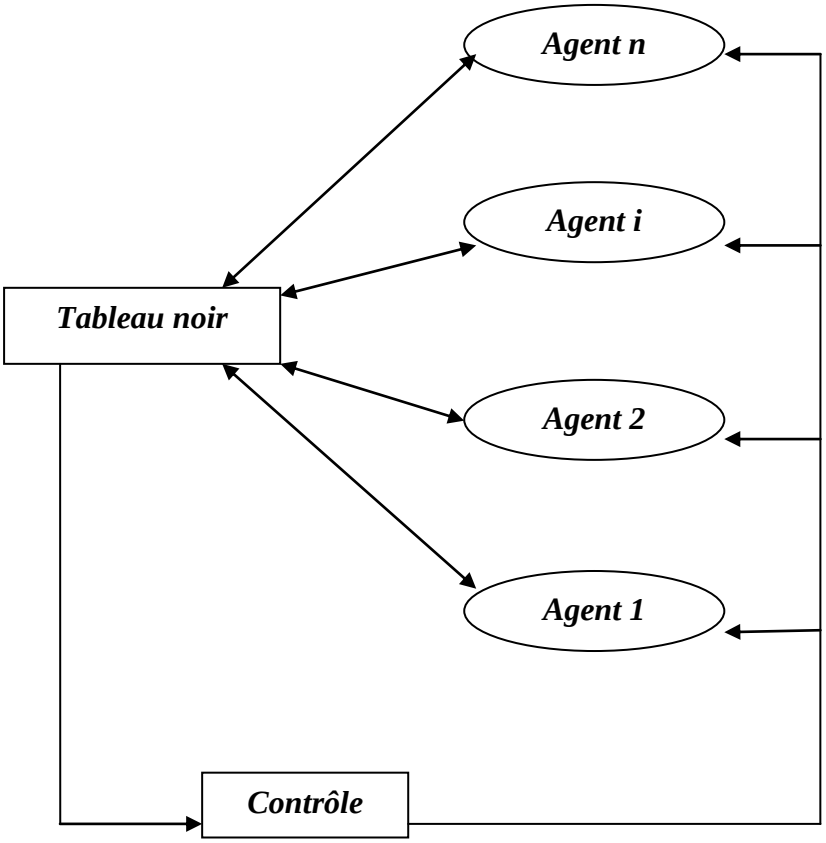

Figure 6 : «*architecture Tableau noir* »

Dans cette architecture la communication n'est pas directe et l'interaction entre agents se fait via le partage d'un même espace de travail (figure 6).

- **Communication par envoi de messages :**

Les agents sont en liaison directe et envoient leurs messages directement et explicitement au destinataire. La seule contrainte est la connaissance de l'agent destinataire. Les systèmes fondés sur la communication par envoi de messages relèvent d'une distribution totale à la fois de la connaissance, des résultats et des méthodes utilisées pour la résolution du problème.

Architecture à contrôle distribué :

Distribution totale des connaissances et du contrôle (figure 7).

.

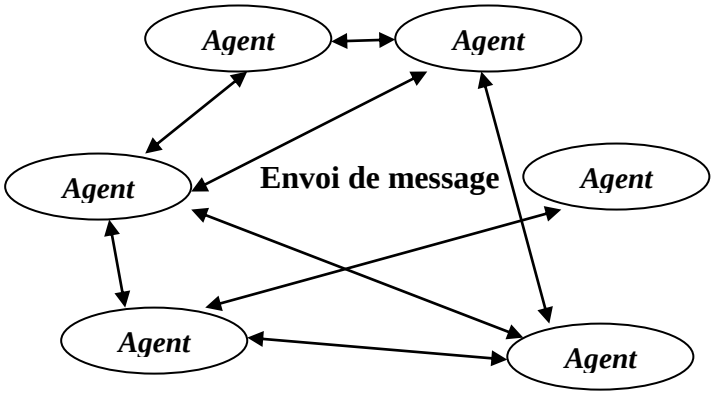

Figure 7 : « *architecture à contrôle distribué* »

c) Les langages de communication

Lorsque la notion de groupe d'agents est apparue, les recherches sur des mécanismes permettant aux agents de communiquer entre eux se sont amorcées. Celles-ci ont mené à trois langages de communication et de représentation de l'information qui sont devenus des « standards » en SMA : KQML (**Knowledge Query Manipulation Language)**. KQML détermine un format pour les messages et un protocole pour la réception et l'envoi de ces derniers. KQML permet aux agents de partager des informations avec les autres agents du système afin de coopérer pour résoudre un problème [13]. Il y a d'autre langage qui s'appelle Le langage (FIPA ACL), est un langage semblable à KQML auquel des protocoles d'interaction comme le *contractnet* et autre protocole populaire ont été ajoutés.

III.3 Les plateformes multi-agents [15]

Des plates-formes typiques incluent l'architecture d'un ordinateur, le système d'exploitation, programmant des langues et l'interface utilisateur liée (des bibliothèques de système de temps d'exécution ou l'interface utilisateur graphique).

Une plate-forme est un élément crucial dans le développement logiciel. Une plate-forme pourrait être simplement définie comme un endroit pour lancer le logiciel. Le

fournisseur de plate-forme offre un engagement au développeur de logiciels que le code de logique exécutera successivement tant que la plate-forme fonctionne en plus d'autres plates-formes.

Plateforme est Intergiciel pour le développement d'applications pair à pair d'agents intelligents pour ce domaine de mémoire de système multi agent.

- **JADE**

(Java Agent DEvelopement) est un Framework de développement de systèmes multi-agents, open-source et basé sur le langage Java. Jade fournit des classes qui implémentent « JESS » pour la définition du comportement des agents.

L'outil possède trois modules principaux (nécessaires aux normes FIPA). Le DF « Director Facilitor » fourni un service de pages jaunes à la plate-forme. L'ACC « Agent Communication Chanel » gère la communication entre les agents. L'AMS « Agent Management System » supervise l'enregistrement des agents, leur authentification, leur accès et utilisation du système. Les agents communiquent par le langage FIPA ACL. Un éditeur est disponible pour l'enregistrement et la gestion des agents. Aucune autre interface n'est disponible pour le développement ou l'implémentation. (Voir l'annexe 1)

- **CORMAS**

(COmmon Resources Multi-Agent System) est un Framework de développement de systèmes multi-agents, open-source et basé sur le langage de programmation orientée objet Small Talk.

- **MAGIQUE**

Est une plate-forme pour agents physiquement distribués écrite en Java et fournissant un modèle de communication original d'appel à la cantonade. Dans MAGIQUE, les compétences sont dissociées des agents. L'architecture des agents et les différentes compétences sont développées séparément. Les compétences sont ensuite greffées comme plugin dans les agents au gré du concepteur.

IV. Les avantages des agents et les systèmes multi-agents

Ils possèdent également les avantages traditionnels de la résolution distribuée et Concurrente de problèmes :

– ***La modularité*** permet de rendre la programmation plus simple. Elle permet, de plus, aux systèmes multi agents d'être facilement extensibles, parce qu'il est plus facile d'ajouter de nouveaux agents à un système multi agent que d'ajouter de nouvelles capacités à un système monolithique.

– ***La vitesse*** est principalement due au parallélisme, car plusieurs agents peuvent travailler en même temps pour la résolution d'un problème.

– ***La fiabilité*** peut être également atteinte, dans la mesure où le contrôle et les responsabilités étant partagés entre les différents agents, le système peut tolérer la défaillance d'un ou de plusieurs agents. Si une seule entité contrôle tout, alors une seule défaillance de cette entité fera en sorte que tout le système tombera en panne.

- ***Et d'autres avantages :*** Distribué ; Parallèle ; Flexible ; Adaptatif ; Récursif et facile.

V. Conclusion

Dans ce chapitre, nous avons présenté une vue générale sur le domaine des Systèmes multi-agents. Le concept de base de ce domaine est la notion d'agent, qui représente une entité autonome capable de percevoir, de se représenter et d'agir sur son environnement.

En plus les avantages déjà cités dans ce chapitre, les caractéristiques des SMA permettent de respecter les normes du génie logiciel dans l'élaboration des systèmes, à savoir : la modularité, la fiabilité et la réutilisation ce qui explique l'utilisation croissante des SMA dans les systèmes informatique.

On peut dire aussi que les systèmes multi-agents se trouvent à la croisée de nombreux autres domaines tels que l'Intelligence Artificielle Distribuée, la théorie des actes de langages, mais aussi la théorie des organisations et bien évidemment le génie logiciel, ce qui rend ce domaine un vaste champ de recherche.

Chapitre 3 : application

I. Introduction

Ce chapitre est consacré à la partie pratique de notre travail. Il décrit l'architecture du système réalisé, son fonctionnement ainsi que l'ensemble des outils utilisés dans la phase d'implémentation.

II. Le But du travail

Le but de notre travail est de construire un SMA qui permet de manipuler les nombres entiers et savoir si un entier est premier ou non.

III. Architecture

III.1 Architecture générale.

L'application implémentée se compose d'un système multi-agents (SMA) et d'une interface de communication entre le SMA et l'utilisateur (figure 8).

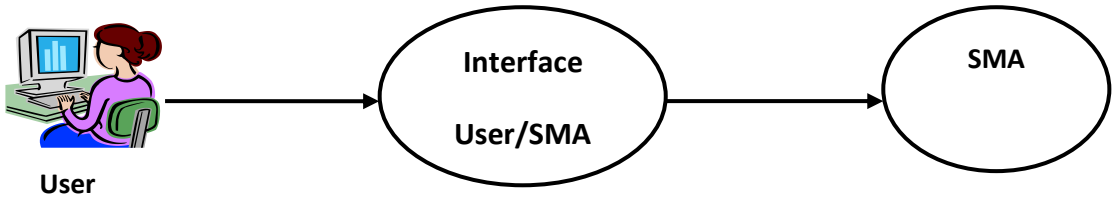

Figure 8 : « Architecture général de l'application »

(Figure 9) on donne un aperçu de l'interface graphique de notre application. A travers cette interface, l'utilisateur lance la plateforme multi agents, contrôle le nombre d'agents participants à l'opération de calcul (test de primalité) et fournit bien sur le nombre a testé au système multi-agents.

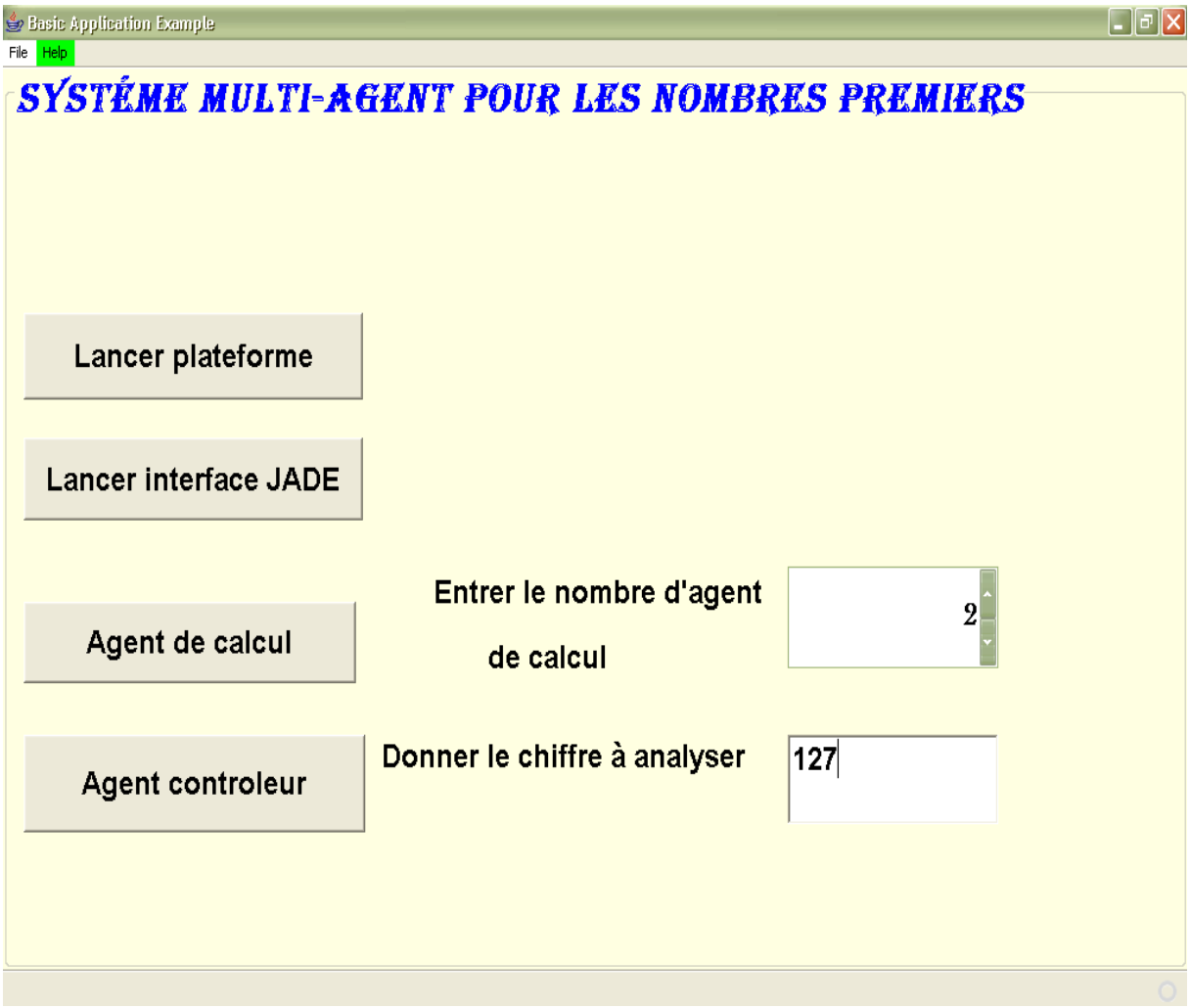

Figure 9 : « aperçu le l'interface graphique de l'application »

III.2 Le système multi agents

Notre SMA se compose de deux types d'agents : un agent contrôleur et un ou plusieurs agents de calcul. Le diagramme de classes de la figure 10 représente la structure ce système. Le rôle de chaque agent est décrit comme suite :

l'Agent contrôleur

C'est l'agent central, son rôle est de récupérer le nombre fourni par l'utilisateur, de diviser ce nombre en intervalles en fonction de nombre d'agents de calcul, d'envoyer l'entier à traiter et l'intervalle de calcul à chaque agent concerné, et enfin de recevoir les résultats des tests fournis par les agents de calculs et prévient l'utilisateur du résultat global (nombre premier ou non).

L'agent de calcul

Le SMA contient un ou plusieurs agents de ce type. Le rôle principal de cet agent est de vérifier si un nombre admet un diviseur dans un intervalle donné en suite renvoi le résultat de cette vérification à l'agent contrôleur. Dans notre application le nombre et l'intervalle sont fournis par l'agent contrôleur.

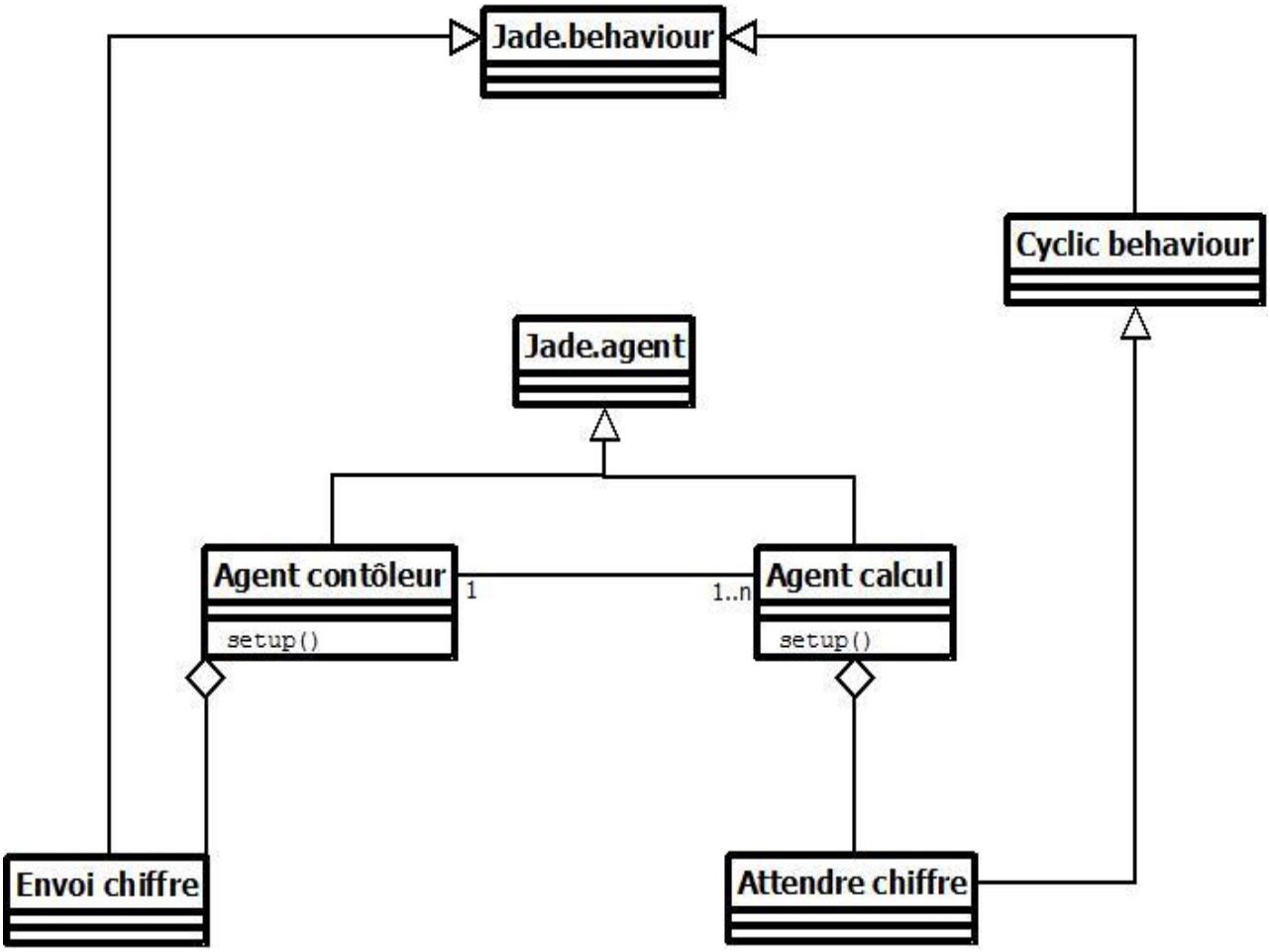

Figure 10 : « Diagrammes de classes »

IV. fonctionnement

La figure 11 présente un exemple simplifié du fonctionnement de notre système. Le SMA dans l'exemple se compose d'un agent contrôleurs et de deux agents de calculs.

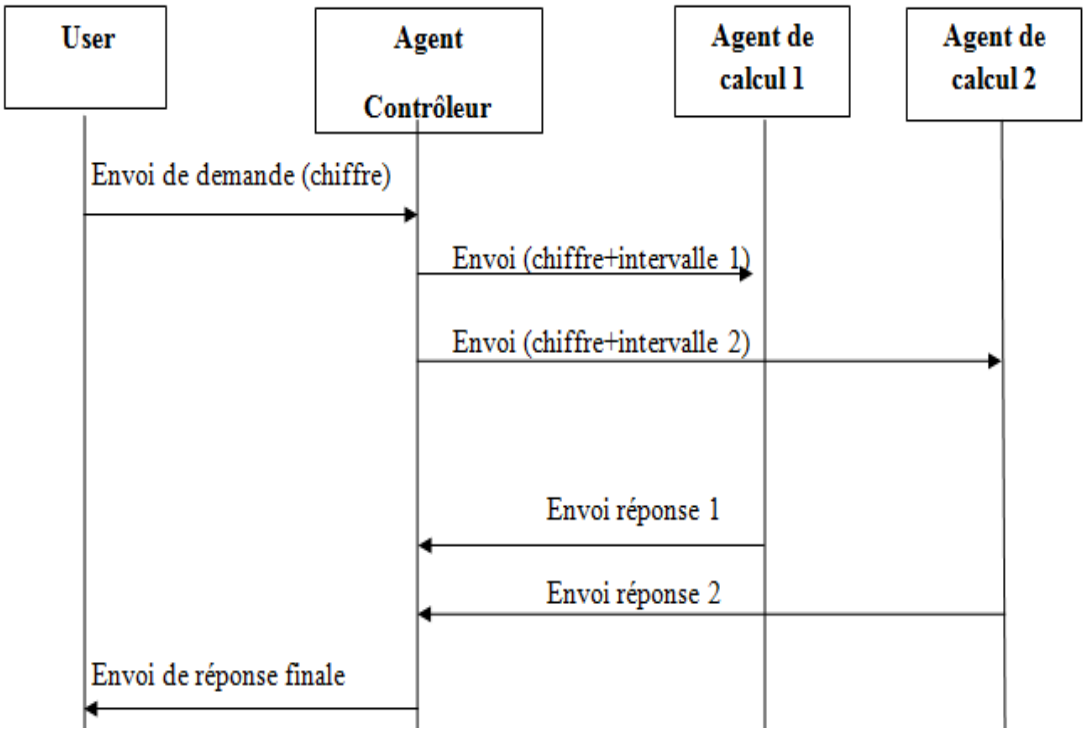

Figure 11 : « : diagramme de séquence »

Après que l'agent contrôleur récupère le nombre fourni par l'utilisateur, il fait la construction de deux intervalles en utilisant ce nombre. Il envoi ensuite a chaque agents de calcul, le nombre à tester ainsi que l'un des intervalles construits. Chaque agent de calcule test dans l'intervalle reçu s'il existe un diviseur de ce nombre et renvoi le résultat du teste à l'agent contrôleur. Selon les résultats fournis par les agents de calcul, l'agent contrôleur peut décider si le nombre est premier ou non, Il envoi en suite le résultat de cette décision à l'utilisateur.

La figure suivante montre la communication entre les agents durant un exemple d'exécution de notre application. Cette figure est extraite à partir de l'outil « *agent sniffer* » de la plateforme JADE.

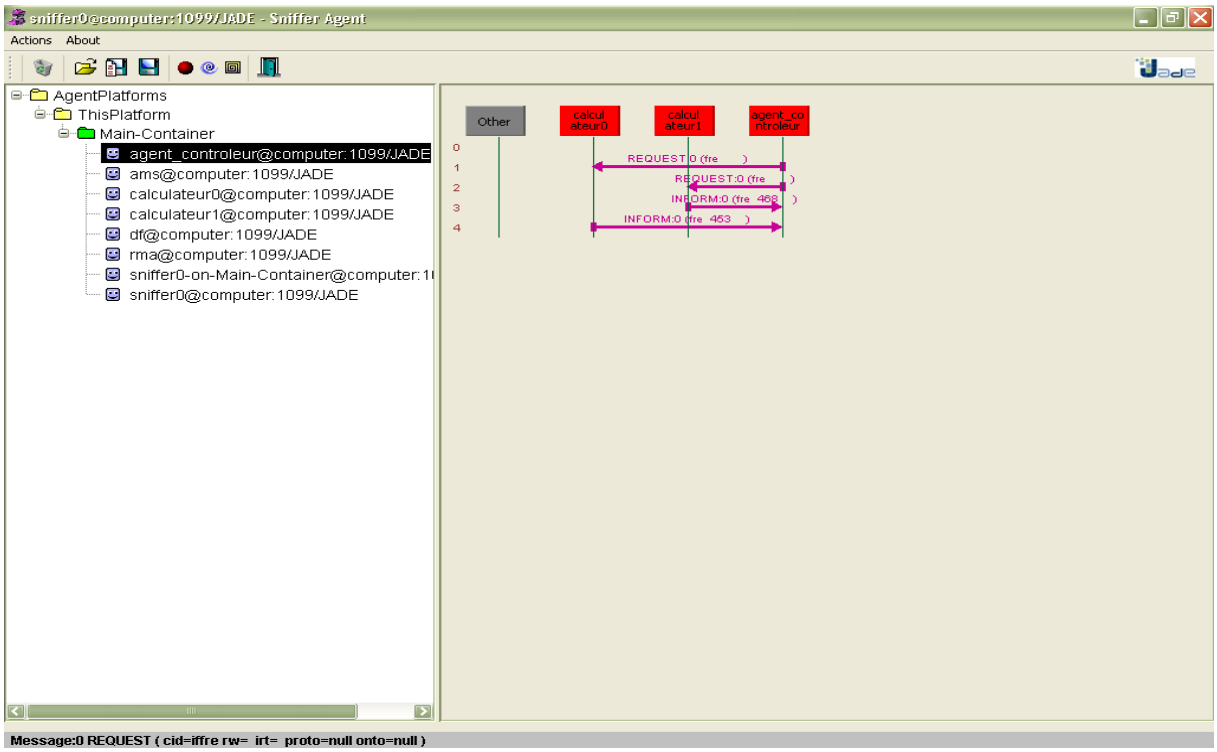

Figure 12 : « échange de message entre les agents du système »

V. Les outils de développement

Notre application est implémentée en utilisant le langage JAVA, sous l'environnement de développement ***NetBeans***. Pour le système multi agents nous avons utilisé la plateforme **JADE**.

Le choix de langage JAVA a été motivé par les raisons suivantes :

- Les agents développés sous la plate-forme JADE, sont entièrement écrits en Java. Ce langage est donc imposé comme étant une conséquence de notre précédent choix en terme de plate-forme de développement d'SMA;
- Java assure une totale indépendance des applications vis-à-vis de l'environnement d'exécution : c'est à dire que toute machine supportant Java est en mesure d'exécuter un programme sans aucune adaptation (ni recompilation, ni paramétrage de variables d'environnement) ;
- JAVA possède une bibliothèque immense d'objets prêts à l'emploi, ce qui facilite pleinement la procédure d'implémentation.

V.1 NetBeans

NetBeans est un EDI (Environnement de Développement Intégré), un environnement libre et facile à utiliser. Il supporte plusieurs langages de programmation à savoir JAVA, C++, PHP, et bien d'autres. Il fourni plusieurs outils tel qu'un éditeur de texte doté d'un pré-compilateur avancé, un gestionnaire de projets. Ainsi que des outils de débogage et de test des programmes. C'est un outil qui facilite énormément la phase de développement et des testes.

La figure 13 présente une capture écran de Netbeans 6.0, utilisé dans l'implémentation .

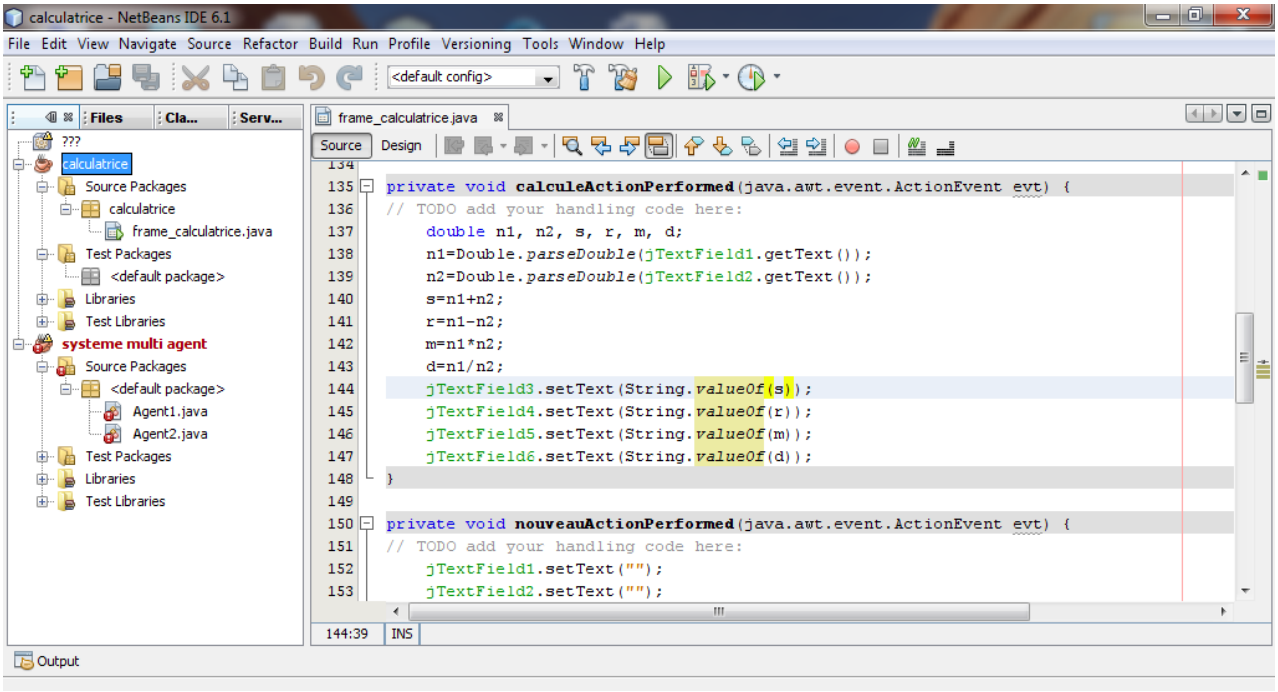

Figure 13 : « interface de NetBeans IDE 6.0 »

V.2 La plateforme JADE (Java Agent Development Framework)

JADE est une plate-forme de développement d'agents gratuite et Open Source développée par CSELT et qui résulte principalement des activités de recherche (voir annexe I pour plus de détaille).

Le choix de cette plateforme est justifié par les points suivants:

- JADE simplifie l'implémentation d'un SMA à travers un Middleware répondant aux spécifications de la FIPA une librairie de classes que les utilisateurs peuvent utiliser et étendre ainsi un ensemble d'outils graphiques qui permettent le débogage et l'administration du SMA à concevoir ;
- JADE assure une communication transparente par l'échange de messages dans le langage normalisé FIPA-ACL ;

- JADE diminue l'effort de programmation car elle implémente deux agents (DF et AMS) dont les fonctionnalités sont utiles à notre application.

La figure 14 présente une capture écran de l'interface graphique de la Plateforme JADE.

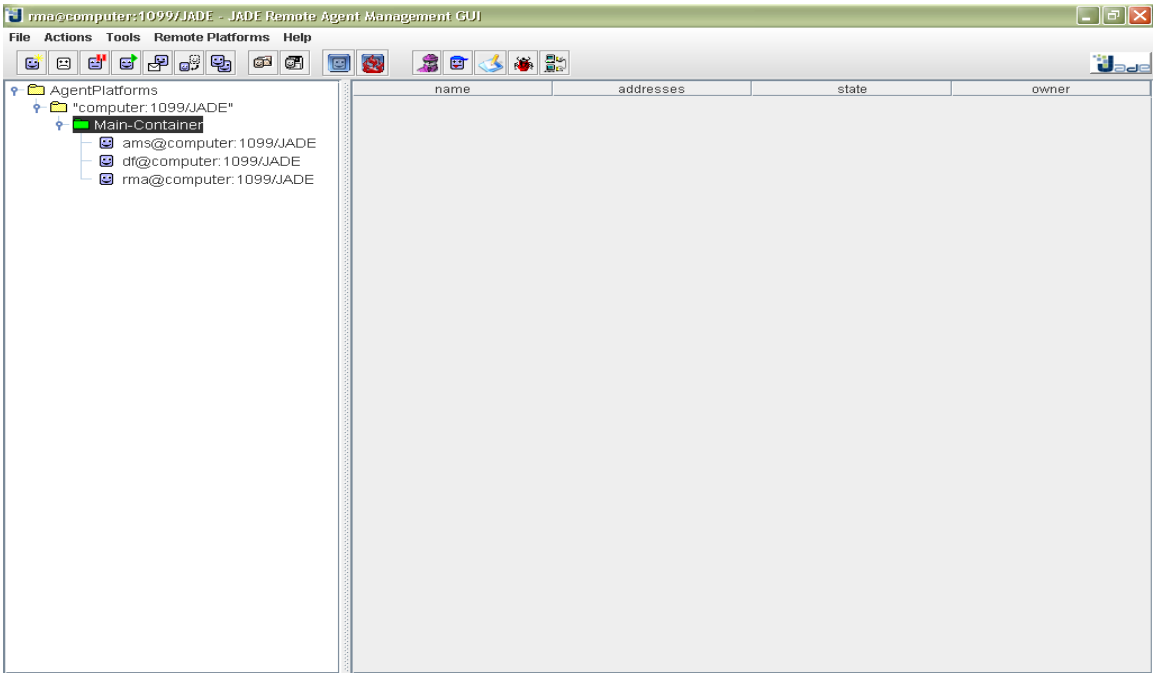

Figure 14 : « interface graphique de la plateforme JADE »

VI. Conclusion

Dans ce chapitre nous avons présenté l'architecture et l'implémentation de notre application. Cette application exploite la technologie multi-agent, pour le test de primalité des entiers naturels. Les tests de l'application implémentée sont bien déroulés, notre application a montré une parfaite collaboration entre les différents agents pour atteindre le but final. Les résultats fournis sont fiables, vu que les tests ont été faits sur des nombres bien connus comme premiers ou non. On peut dire que le résultat notre travail est acceptable par rapport à notre première expérience dans ce domaine.

Conclusion générale

Le travail présenté dans ce mémoire tourne autour des systèmes à base d'agents. L'objectif était de donner une vue générale sur ce paradigme en introduisant quelques concepts et notions théoriques relatives. Nous avons pu acquis une expérience pratique en développant une petite application. Cette application se compose d'un ensemble d'agents qui collaborent pour tester la primalité d'un entier naturel. La mise en œuvre de cette application est effectuée en utilisant le langage JAVA sous la plate forme JADE.

Bien que l'application développée est loin d'être utilisée sur le plan pratique, L'implémentation nous a permis :

- De faire connaissance avec le langage Java, un langage en plein essor et de plus en plus utilisé dans presque touts les domaines.
- de voir en pratique des concepts de programmation plus avancés tel que le développement à base d'agent et l'utilisation pratique de d'une plateforme multi-agents.

Cette Expérience est un bon complément de notre formation de base, elle nous a permis d'enrichir nos connaissances théoriques et pratiques, et constitue la base de départ pour des futurs travaux.

Nos perspectives étant de continuer dans ce domaine (les SMA), de bâtir une base solide pour pouvoir développer des applications plus consistantes, et pourquoi pas de faire des contributions dans ce domaine.

Annexe

Annexe 1

1- **La plate forme jade**

2- **Langage de communication de la plate-forme JADE**

3- **Outils de débogage de JADE**

4- **Conclusion**

1- La plateforme JADE

Le meilleur moyen pour construire un système multi-agent(SMA) est d'utiliser une plate-forme multi-agent. Une plate-forme multi-agent est un ensemble d'outils nécessaire à la construction et à la mise en service d'agents au sein d'un environnement spécifique. Ces outils peuvent servir également à l'analyse et au test du SMA ainsi créé. Ces outils peuvent être sous la forme d'environnement de programmation (API) et d'applications permettant d'aider le développeur JADE (Java Agent DEvelopement framework) est une plate-forme multi-agent créé par le laboratoire TILAB et décrite par Bellifemine et al. Dans [17][16]. JADE permet le développement de systèmes multi-agents et d'applications conformes aux normes FIPA [18]. Elle est implémentée en JAVA et fourni des classes qui implémentent « JESS » pour la définition du comportement des agents. JADE possède trois modules principaux (nécessaire aux normes FIPA).

- DF « Director Facilitor » fournit un service de « pages jaunes» à la plate-forme ;
- ACC «Agent Communication Channel » gère la communication entre les agents ;
- AMS « Agent Management System » supervise l'enregistrement des agents, leur authentification, leur accès et l'utilisation du système.

Ces trois modules sont activés à chaque démarrage de la plate-forme.

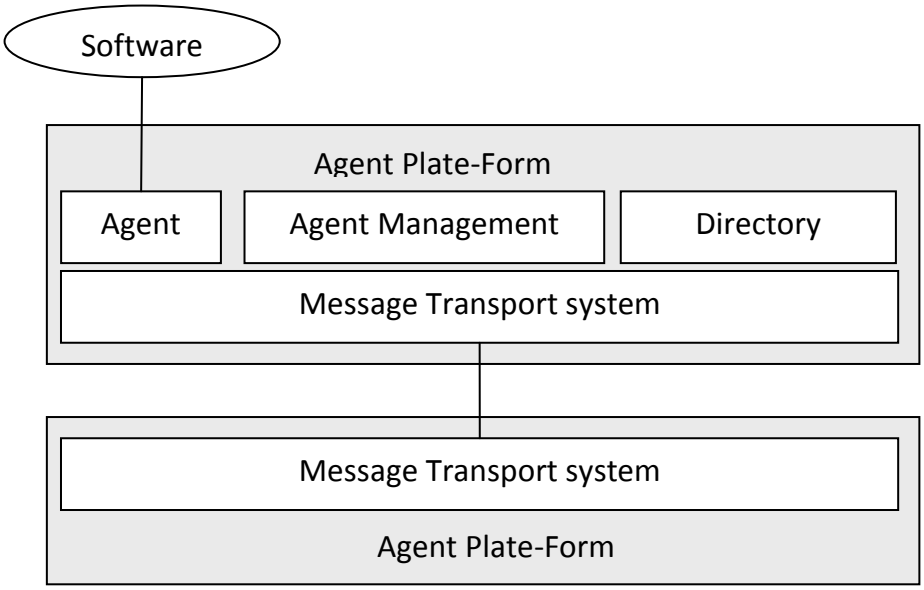

Figure 15 : « «Architecture logiciel de La plate-forme JADE »

2- Langage de communication de la plate-forme JADE

Le langage de Communication de la plate-forme JADE est FIPA-ACL (Agent Communication language). La classe ACLMessage représente les messages qui peuvent être échangés par les agents. La communication de messages se fait en mode asynchrone. Lorsqu'un agent souhaite envoyer un message, il doit créer un nouvel objet ACLMessage, compléter ces champs avec des valeurs appropriées et enfin appeler la méthode send(). Lorsqu'un agent souhaite recevoir un message, il doit employer la méthode receive() ou la méthode blockingReceive().

JADE utilise deux protocoles de communication :

- Les protocoles FIPA-Query et FIPA-Reques.
- Le protocole Contract-Net de JADE

3- Outils de débogage de JADE

Pour supporter la tâche difficile du débogage des applications multi-agents, des outils ont été développés dans la plate-forme JADE. Chaque outil est empaqueté comme un agent, obéissant aux mêmes règles, aux mêmes possibilités de communication et aux mêmes cycles de vie d'un agent générique (agentification de service).

- Agent Dummy.

- Agent Sniffer.

- Agent Inspector.

4- Conclusion

Les technologies d'agent attirent, aujourd'hui beaucoup d'attention, particulièrement pour des applications largement distribuées et ouvertes. Ces applications présupposent l'interopérabilité des agents issus de différentes infrastructures, ce qui mène naturellement à définir des normes qui standardisent leur communication.

Références

- [1] http://www.cnrs.fr/Cnrspresse/math2000/html/math10.htm
- [2] Les Éléments d'Euclide", traduites littéralement par F. Peyrard, Librairie scientifique et technique Albert Blanchard, 1993, page 180.
- [3] P.A. Clement Congruence for sets of primes, American Mathematical Monthly n° 56 (1949), pp. 23-25

- [4] http://fr.wikipedia.org/wiki/Nombres_premiers_cousins

- [5] http://fr.wikipedia.org/wiki/Nombre_premier_de_Chen

- [6] Alexandra Bruasse-Bac (primalité et factorisation 2002-2003)
- [7] Johannes Buchmann : la factorisation des grands nombres (pour la science 1998)

- [8] http://interstices.info/jcms/c_30225/nombres-premiers-et-cryptologie-lalgorithme-rsa.

- [9] Jonathan Touboul (Chercheur) :nombre premier et cryptologie rithme-rsa
- [10] http://cormas.cirad.fr/fr/demarch/sma.htm

- [11]: M. WOOLDRIDGE, N. R. JENNINGS, K. SYCARA. “A roadmap of agent research and development”. Int Journal of Autonomous Agents and Multi-Agent Systems, 1(1):7- 38, 1999
- [12] Guillaume Hutzler / Tarek Melliti Intelligence Artificielle Distribuée Systèmes Multi-Agents(http://www.ibisc.univ-evry.fr/~hutzler/Cours/SMA.html)

- [13] francois Bourdon, « systèmes multi agents cours exposé dans le cadres du DEA IAA » ,2001
- [12] F.Vernadat and P.Azéma. prototypage de système d'agents communicants .In journée système multi-agents PRC-GDR intelligence artificielle . Nancy.decembre 1992.
- [13] [FIN, 94]: T. FININ, R. FRITZON, D. MCKAY, R. MCENTIRE, “KQML as an agent communication language”, In Proceedings of the Third International Conference on Information and Knowledge Management (CIKM'94), ACM Press, 1994.
- [14] P. Nii. Blackboard systems. In Handbook of Artificial Intelligence. Volume IV. Addison- Wesley : Reading, MA, 1989.
- [15]"Ben-delhoum soheib" Hadji brahim Gestion des emplois du temps par les Systèmes Multi Agents 2007/2008
- [16] Bellifemine F., Giovani C., Tiziana T. Rimassa G., "Jade Programmer's Guide"(http://sharon.cselt.it/projects/jade/), 2000.
- [17] Bellifemine F., Poggi A., Rimassa G., "JADE -- A FIPA-compliant agent framework", 1999.
- [18] Bellifemine F., Caire G., Trucco T., Rimassa G., "JADE : Programmer's Guide", 2002.