NetSA : Une architecture multiagent pour la recherche sur Internet

Marc Côté et Nader Troudi

Université Laval
Département d'informatique
Pavillon Pouliot
Ste-Foy, G1K 7P4, Canada.

{mcote, troudi}@iad.ift.ulaval.ca

Notice biographique

Marc Côté:
Marc Côté est diplômé du département d'informatique de l'université Laval. Il est présentement à sa dernière année de maîtrise en informatique. Il travaille sur les agents et les systèmes multiagent sous la direction de M. Brahim Chaib-draa. Rantanplan, un agent pour la surveillance de pages Web sur Internet fût l'un de ses premières réalisation dans ce domaine. L'architecture NetSA occupe présentement son horaire de travail. Une première version de ce système multiagent est prévue pour décembre 1998. Marc tente également de réaliser un agent pour le filtrage des messages indésirables (spam) à l'aide de l'extraction des connaissances dans un texte.

Nader Troudi :
Nader Troudi est un étudiant Tunisien titulaire d'un diplôme en informatique de gestion de l'Institut Supérieur de Gestion (ISG) de Tunis. Il termine sa maîtrise en informatique au département d'informatique de l'Université Laval. Il s'intéresse aux agents logiciels et aux systèmes multiagents. Sous la direction de M. B. Chaib-draa, Nader se concentre sur l'étude des agents intermédiaires et sur les protocoles de négociation entre les agents. Durant ses études de maîtrise il a acquis une grande expérience dans la programmation des agents avec Java.

Résumé

Aussi longtemps qu'Internet poursuivra son évolution, nous continuerons à être submergés par des données, sans que celles-ci soient toutefois structurées. Le recherche d'information dans ce cadre devient une tâche difficile et les méthodes traditionnelles de recherche sur Internet ou sur des bases de données s'avèrent de plus en plus limitées. Les systèmes d'informations coopératifs basés sur les agents logiciels apportent de solutions prometteuses à cet épineux problème. Dans cet article, nous décrivons NetSA : une architecture de système multiagent pour la recherche d'information dans des sources hétérogènes et réparties.

Mots clés : agents, système multiagent, recherche d'informations , bases de données, systèmes répartis, Java.

1. Introduction

Le domaine relatif à l'offre et à la demande d'informations est en train de subir de grands changements. Le premier changement qui a eu lieu il y a déjà quelques années, se rapporte au moyen de transmission de l'information. C'est ainsi que le papier est apparu comme support d'information, et il est encore aujourd'hui fréquemment utilisé. De nos jours cependant, les informations sont de plus en plus disponibles sur des supports électroniques. Un autre aspect de changement concerne la quantité d'informations disponible, le nombre de sources de données et la facilité avec laquelle l'information est obtenue. Ce même aspect a généré des inquiétudes face à la fiabilité des sources d'information et donc à la qualité de cette dernière. Le troisième changement est lié à l'offre et à la demande de l'information. Jusqu'à récemment, l'offre a déterminé le marché de l'information, qui a été approvisionné par un petit groupe de fournisseurs facilement identifiables. Actuellement, le marché est si large, qu'il est presque impossible d'avoir une idée claire sur tous les fournisseurs.

Tous ces changements ont un impact énorme sur le marché de l'information. Mais l'impact le plus important, selon nous, est que ce n'est plus l'offre qui détermine le marché mais bel et bien la demande. Le nombre de fournisseurs a tellement augmenté (et il va augmenter encore) qu'il est devenu difficile de savoir qui offre l'information. Autrement dit, la demande d'information devient l'aspect le plus important dans la chaîne de traitement d'information. De plus, l'information joue un rôle prépondérant dans notre vie quotidienne, ce pourquoi nous disons vivre dans « l'ère de l'information ». Son accès est devenu vital pour la résolution de nos problèmes.

Internet représente aujourd'hui une source d'information inévitable. Contrairement aux moteurs de recherches conventionnels qui utilisent généralement des techniques d'indexation de l'information disponible sur Internet, notre architecture utilise des techniques de raisonnement sur les concepts tout en permettant aux agents d'interagir selon des stratégies coopératives pour trouver la bonne information.

Le présent article est structuré comme suit : dans la prochaine section nous identifierons les problèmes relatifs à la recherche d'informations. Par la suite, dans la troisième section, nous présenterons les solutions possibles et comparerons les moteurs de recherche avec les agents logiciel. Une nouvelle approche pour la recherche d'information, celle basée sur les systèmes multiagent, sera décrite dans la section 4. Dans la section 5, nous détaillerons notre système multiagent NetSA, en présentant tout d'abord l'architecture générale suivie des composantes de cette architecture.

2. Les problèmes relatifs à la demande d'informations

La satisfaction d'une demande d'information est devenue à la fois plus facile et plus compliquée. Elle est devenue plus facile dans la mesure où grâce à l'émergence de nouvelles sources de données, comme le réseau international appelé Internet, chacun, en principe, peut avoir l'accès à une source d'informations inépuisable. Cependant, la masse énorme d'informations disponibles sur Internet, même sur un intranet ou un Data Warehouse, qui, à première vue, semble être sa force majeure, est en même temps l'une de ses faiblesses. La quantité d'informations à la disposition de l'utilisateur, généralement un décideur, est trop grande : l'information recherchée est probablement disponible quelque part, mais il arrive souvent qu'une seule partie soit retrouvée, et parfois même rien du tout. Les méthodes de recherche d'information conventionnelles se sont avérés incapables de résoudre ces problèmes. Ces méthodes supposent que nous connaissons d'avance quelle information est valable et où exactement elle peut être trouvée. De telles méthodes sont utilisées de la manière suivante : les systèmes d'informations, comme les bases de données, sont approvisionnés avec des indices qui fournissent ces informations aux usagers. Grâce à ces indices, l'utilisateur peut, à tout moment, vérifier si certaines informations sont offertes par la base de données, si elles sont disponibles, et où il peut les trouver. Avec les nouvelles technologies notamment Internet, mais aussi intranet/extranet et Data warehouse, ces stratégies ne sont plus applicables. Les raisons à cela sont les suivantes :

- La nature dynamique d'Internet : aucune supervision centrale ne s'applique quant au développement d'Internet. Toute personne qui désire l'utiliser et/ou offrir des informations ou des services est libre de le faire. Ceci a crée une situation où il est devenu très difficile d'avoir une idée claire sur la taille réelle d'Internet ;
- La nature dynamique des informations : les informations qui ne sont pas disponibles aujourd'hui peuvent être disponibles demain et le contraire s'applique aussi ;
- L'information est hétérogène : l'information est offerte sous plusieurs formats et de plusieurs façons. Ceci complique la recherche automatique d'une information donnée, puisque chaque format et chaque service nécessitent une approche particulière.

3. Les solutions possibles

Plusieurs solutions existent pour résoudre les problèmes identifiés précédemment. La plupart sont des solutions ad hoc. C'est ainsi qu'en utilisant des programmes qui circulent sur Internet, nous pourrons gérer des méta-informations concernant tous les documents disponibles. L'information collectée, est caractérisée par un ensemble de mots-clés, est sauvegardée dans des bases de données de grande taille. Toute personne qui désire chercher des informations peut les localiser en donnant un ou plusieurs mots-clés à ce moteur de recherche. Bien que les moteurs de recherche fournissent des services plus ou moins bons, ils possèdent plusieurs inconvénients (qui seront plus évidents dans le futur).

Une solution totalement différente pour ce problème est l'utilisation des agents logiciels. Pattie Maes définie un agent logiciel comme étant un programme informatique autonome qui assiste l'utilisateur dans l'exécution de ses

tâches et qui communique avec d'autres agents [MAES 1994]. Si en plus de ces caractéristiques l'agent peut manipuler des symboles ou des abstractions, peut agir en temps réel, peut apprendre et peut s'adapter aux préférences de l'usager, nous parlons alors dans ce cas des agents logiciels intelligents.

L'utilisation des agents logiciels pour la recherche d'informations offre certains avantages par rapport aux méthodes courantes tels que les moteurs de recherches. Le Tableau 1 récapitule ces avantages.

	Les moteurs de recherches	Les agents
Critères de recherche	La recherche d'informations est faite en se basant sur un ou plusieurs mots-clés. Ceci suppose que l'usager est capable de formuler exactement ses mots-clés. Dans le cas contraire, plusieurs informations non pertinentes seront retournées et des informations pertinentes ne seront jamais retrouvées.	Les agents sont capables de chercher l'information d'une façon plus intelligente, par exemple en cherchant selon des concepts. Les agents sont également capables de corriger les requêtes de l'usager, en se basant sur le modèle de ce dernier ou sur d'autres informations.
Indexation	L'indexation d'information est faite par collection de méta-informations sur les informations et sur les documents disponibles sur le web. C'est une méthode coûteuse (en temps et en ressources), inefficace et qui ne correspond pas bien à la nature dynamique de l'Internet.	Les agents peuvent créer leurs propres bases de connaissances qui sont mises à jour après chaque recherche. Si l'information change de site, les agents sont capables de la trouver et, par la suite, s'adapter à ce changement. En plus, les agents sont capables de communiquer et coopérer entre eux (et c'est là leur vraie force), ce qui accélère et facilite la recherche.
Interface usager	La recherche d'information est souvent limitée à quelques services (WWW). Trouver l'information offerte par d'autres services (des bases de données) oblige souvent l'usager à se débrouiller seul.	Les agents peuvent débarrasser l'utilisateur de certains détails, comme la façon avec laquelle un service doit être manipulé. L'usager se concentre seulement sur ce qu'il cherche, l'agent s'occupe du reste.
accessibilité	Les moteurs de recherches ne sont pas toujours accessibles, faute de connexion ou de congestion. L'usager sera alors oblige d'utiliser un ou plusieurs autres moteurs de recherche ce qui nécessitera probablement une autre façon de procéder.	Etant donné que l'agent réside sur la machine de l'utilisateur, il est toujours à la disposition de ce dernier. Un agent peut exécuter plusieurs tâches jour et nuit, et parfois même il pourra les exécuter en parallèle. L'avantage d'un tel agent réside aussi dans le fait qu'il est intelligent et qu'il peut par conséquent essayer d'éviter les heures de pointe.
adaptabilité	L'information sur le roseau est très dynamique, souvent les moteurs de recherches font références à des informations dont la localité a changé. Les moteurs de recherches n'apprennent pas et ne s'adaptent pas aux usagers. En plus, l'utilisateur ne peut pas recevoir les mises à jour des informations. Faire de la recherche d'informations d'une telle façon est très coûteux.	Les agents s'adaptent aux préférences et aux souhaits de chaque usager. Ils peuvent ainsi apprendre de leurs recherches précédentes et par la suite comprendre mieux les besoins des utilisateurs.

Tableau. 1: Comparaison entre les moteurs de recherches et les agents logiciels [d'après, HERMANS 1997]

4. La recherche coopérative basée sur les systèmes multiagents

Comme d'autres technologies, l'évolution d'Internet est continue. Le volume des données sera trop grand et trop varié de telle façon qu'il sera impossible pour l'être humain de suivre ce qui se passe. Le pire, c'est que prochainement les logiciels conventionnels ne seront plus capables de maîtriser la situation, par conséquent une nouvelle structure pour la recherche d'informations s'avère dès aujourd'hui nécessaire. Une telle structure facilitera la tâche et fera abstraction des différentes techniques. Ce type d'abstraction est comparable à celui avec lequel les langages de programmation de haut niveau ont débarrassé les programmeurs de tous les problèmes de bas niveau (registres et appareils).

Etant donné cependant, que le processus de réflexion relatif à ces idées est très récent, aucun standard n'est établie. Une idée prometteuse a cependant émergée ces dernières années. Cette idée consiste en un ensemble d'agents intelligents qui communiquent en vue de collaborer entre eux. C'est ce que nous appelons les systèmes multiagents. La définition, la conception, les composantes et les caractéristiques d'une telle architecture seront présentées dans la partie suivante.

5. NetSA

Aujourd'hui il y a lieu des architectures mutiagents pour la recherche sur Internet de par le monde (HUHNS 1997). La plupart de ces architectures ont toutefois été développées pour une application donnée et de ce fait, elles ne peuvent être réutilisées pour d'autres applications. Il faut savoir que la construction d'un système multiagent n'est pas simple en soi et que parfois même elle peut s'avérer être très complexe [WOOLDRIDGE 1998]. De ce fait, le coût de production d'un système multi-agent est énorme en personne/mois tant au niveau de la conception, que de la réalisation et de la vérification (déverminage). Il nous faut donc utiliser des moyens pour réduire la charge de travail et par le fait même le coût de production. Une des nombreuses solutions apportées est la réutilisation d'une architecture de système multiagent qui soit portable et universelle. NetSA (Networked Software Agents) tente de répondre à cette demande en proposant une architecture complète de système multiagent. Dans les lignes qui suivent, nous donnerons un aperçu de NetSA et de son architecture.

5.1. Historique et technologies

Le projet NetSA a débuté en Janvier 1998 au laboratoire DAMAS (Data-minning agents et multigents) du département d'informatique de l'Université Laval, sous la direction du Pr. B. Chaib-draa. Il a été initié par un contrat entre NCR- le centre allemand d'intelligence artificielle (DFKI) et DAMAS en vue d'élaborer un prototype pour les systèmes financiers. Le projet est en cours de développement et une première version est prévue pour la fin de 1998. Le groupe de recherche en charge du projet NetSA a opté pour les technologie standardisées telles que :

- KQML (Knowledge Query and Manipulation Language) [FININ 1997] ;
- KIF (Knowledge Interchange Format) [GENESERETH 1995];
- Java [GOSLING 1996] ;
- HTML (Hyper Text Markup Language) [POWEL 1998];
- CGI (Commun Gateway Interface) [IVLER 1997] ;

Ceci procure au système NetSA un haut niveau d'ouverture, de flexibilité et d'extensibilité.

5.1.1. KQML et KIF

KQML est un langage d'interrogation et de manipulation des connaissances. C'est un langage basé sur les actes du langage naturel qui est utilisé surtout pour la communication entre agents. Bien que ce langage ne soit pas normalisé, est devenu une norme officieuse et ce, grâce à sa forte utilisation. De son côté, KIF est utilisé pour construire le corps du message KQML. C'est un langage déclaratif de type logique, utilisé pour l'échange de connaissance entre différents programmes. Ainsi les requêtes peuvent être exprimées par ce format. La figure 1, montre un exemple de requête qui utilise KQML et KIF. Dans cet exemple, l'expéditeur (Sender) appelé «l'agent d'exécution» s'adresse à l'agent courtier («agent courtier») en lui recommandant (recommend-one) un agent ressource qui travaille sur la bourse (le nom de l'ontologie est «Bourse») et pour qui le prix a payé soit inférieur à 100. Les trois types d'agents ainsi que l'ontologie seront expliqués plus loin dans cet article.

```
(recommend-one

      :sender « AgentExecution »
      :receiver « AgentCourtier »
      :ontologie « NetSA »
      :content
      :langage KIF
       (and
            (agent ?a)
            (name ?a ?name)
            (address ?addr)
            (address ?a ?addr)
            (host ?addr ?host)
            (port ?addr ?port)
            (protocol ?addr ?protocol)
            (type ?a « AgentRessource»)
            (langage ?a SQL)
            (ontology ?ont)
            (ontology ?a ?ont)
            (name ?ont « Bourse »)
            (name ?s « Prix »)
            (contraint ?c)
            (contraint ?s ?c)
            (expression ?c (< ?s 100))
      ) )
```

Figure. 1: Exemple de requête en KQML et KIF

5.1.2 Java

Dans tout système informatique le choix d'un langage de programmation est très important. Bien que les systèmes multiagent ne sont pas limités au langage Java, ce dernier a été récemment fortement utilisé. En ce qui nous concerne, la portabilité et l'universalité étant des facteurs prédominants dans la conception de notre système multiagent, l'utilisation de Java a été un choix quasi évident. Java est un langage du type compilé une fois et exécuté sur toutes machines. De plus, Java est un langage robuste qui surveille ces accès mémoire continuellement.

5.1.3. HTML et CGI

Pour concevoir des interfaces usagers de haute qualité et faire des formulaires à l'aide de Java, nous avons pensé utiliser des applets. Il réside cependant une inconsistance au niveau des contrôles graphiques entre les différentes plates-formes. Par exemple la qualité graphique des stations Sparcs provoque un rétrécissement des champs textes programmés sous Windows. De plus, les niveaux de sécurité implantés sur les différents navigateurs Internet nous compliquent grandement la tâche. Nous avons donc opté pour des pages HTML contenant des CGI écrits en Javascripts [COHEN 1997]. Ces pages sont expédiées et dirigées par un servlet [VOSS 1998] selon les besoins de l'utilisateur.

5.2. L'architecture NetSA

L'architecture de NetSA est divisée en trois unités jouant chacune un rôle bien déterminé. Comme le montre la figure 2, ces unités sont :

- l'unité de communication avec l'utilisateur,
- l'unité de traitement d'informations,
- l'unité d'interrogation et d'extraction d'informations.

Ces trois unités sont fortement modulaires puisque chacune est spécialisée dans un ensemble de fonctions. Elles sont aussi faiblement couplées puisqu'elles ne sont pas très dépendantes. Ces deux caractéristiques sont souvent recherchées dans tout système informatique. Dans ce qui suit nous détaillons chaque unité et nous étudions sa relations avec les autres unités du système.

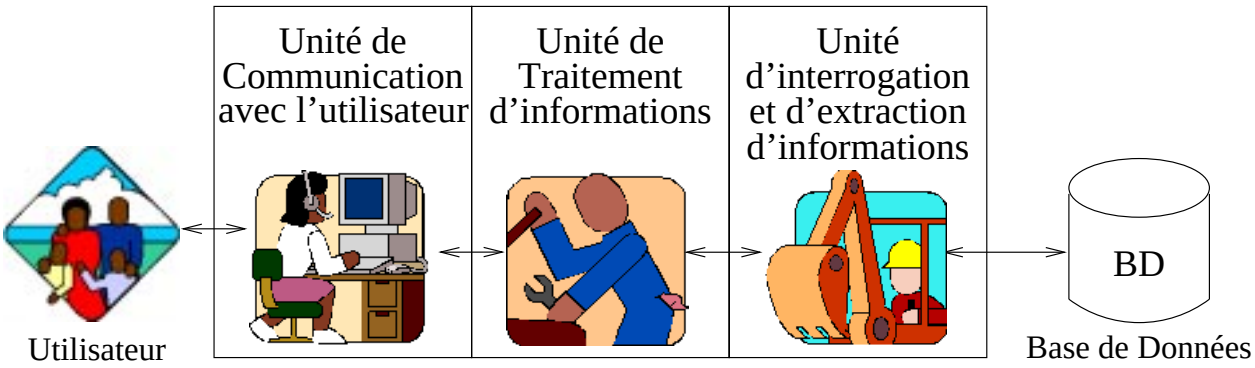

Figure. 2: Les unités de NetSA

5.2.1. Unité de communication avec l'utilisateur

Cette unité est chargée des communications entre NetSA et l'usager. Elle comprend des agents-utilisateur interagissant avec l'usager pour l'aider à réaliser une tâche bien précise. Cette interaction se traduit par une transformation des requêtes de l'usager, qui transformées en des performatives KQML facilitent la communication avec les agents de l'unité de traitement.

5.2.2. Unité de traitement d'informations

L'unité de traitement d'information reçoit de l'unité de communication les requêtes à satisfaire. Elle décompose ces requêtes et les transforme en sous requêtes plus simples qui sont envoyées vers unité d'interrogation et d'extraction d'information. Cette unité fournit également la définition de l'ontologie qu'il convient d'utiliser et agit comme un patrouilleur qui dirige les agents vers la ressource désirée.

5.2.3. Unité d'interrogation et d'extraction d'informations

Unité d'interrogation et d'extraction d'information est une interface entre les bases de données et unité de traitement d'information. Elle transforme les requêtes KQML reçues et les traduit en requêtes SQL pour interrogation des bases de données. De ces bases de données, elle retire l'information pertinente et la redirige vers l'unité de traitement de l'information sous un format KQML.

Nous allons maintenant détailler les différents agents qui comportent ces unités.

5.3. Les types d'agents

Le système multi-agent NetSA comporte différents types d'agents, comme indiqué en figure 3, dans des concentrations variables. Ces différents agents sont :

- plusieurs agents-utilisateurs,
- au moins un agent courtier ,
- au moins un serveur d'ontologie,
- au moins un agent d'exécution,
- plusieurs agents-resources internes ou externes.

Considérons maintenant ces types d'agents.

5.3.1. Agent utilisateur

L'agent utilisateur, situé dans l'unité de communication avec l'utilisateur, est la porte d'entrée des requêtes externes au système. Il fournit à l'utilisateur le bon formulaire HTML/CGI qui lui permettra de faire une requête facilement. L'agent traduit ensuite le formulaire soumis dans le protocole KQML/KIF en vue de son utilisation dans le système. L'agent utilisateur est persévérant et autonome dans le sens où il est capable de garder le profil de l'utilisateur au fur et à mesure que celui-ci utilise le système. Il est capable de stocker de l'information pour l'utilisateur et d'agir comme un agent ressource. Évidemment il y a autant d'agents utilisateurs qu'il y a d'utilisateurs. Chaque agent s'occupe de l'utilisateur auquel il est rattaché. Cet agent est une application Java développée en tant que servlet.

5.3.2. Agent courtier (broker *)*

L'agent courtier ou broker, situé dans l'unité de traitement de l'information, associe aux différentes requêtes les agents qui sont capables d'y répondre [SYCARA 1996]. Les agents ressources peuvent s'y inscrire en envoyant un message en KQML qui avertira l'agent broker du type de service fourni par le nouvel agent ressource. Lorsqu'un agent demande à l'agent broker qui peut faire sa requête, il lui répond en lui donnant le nom de ou des agents aptes à faire la requête. On peut rendre le système plus robuste en ajoutant un second broker qui prendra la relève du premier en cas de défaillance ou qui en allégera la charge.

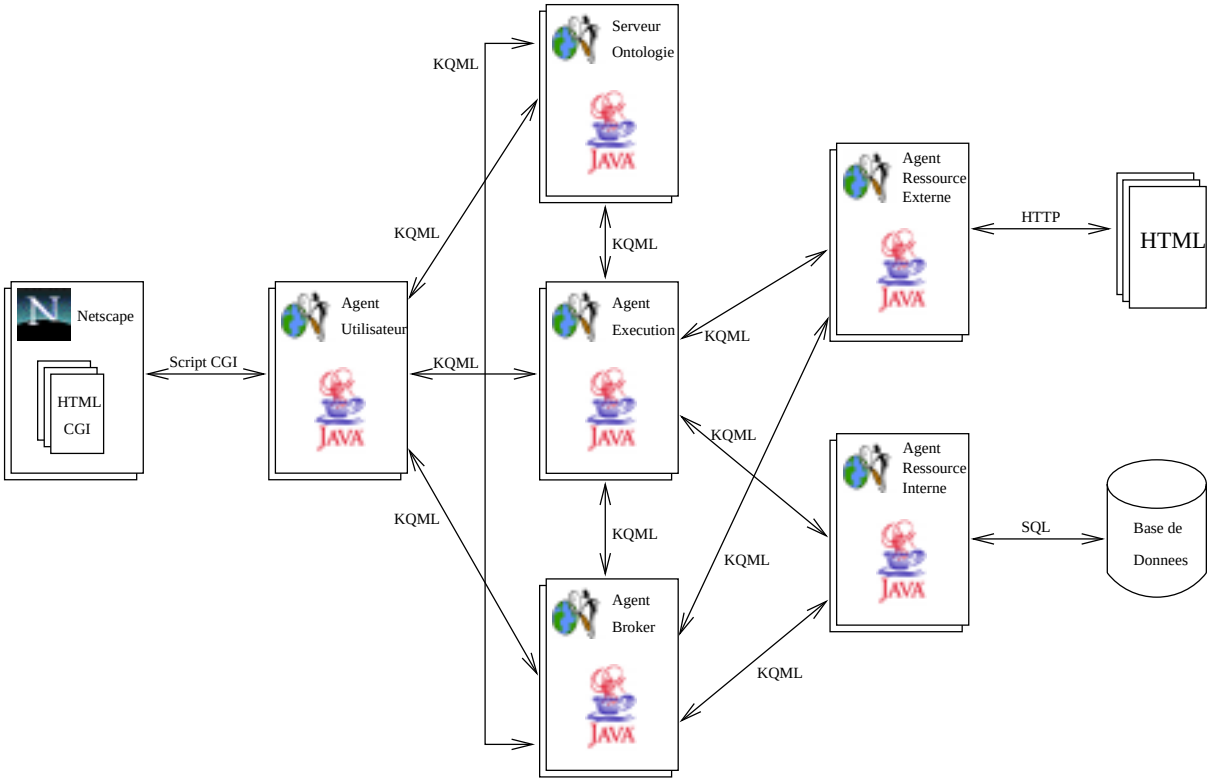

Figure. 3: Les agents de NetSA

5.3.3. Serveur d'ontologie

Une ontologie est un ensemble de vocabulaires qui décrit l'univers du discours d'un domaine donnée. Ce vocabulaire sera ensuite partagé par les agents pour assurer une bonne communication. Notre serveur ou agent d'ontologie, situé dans l'unité de traitement de l'information, a pour responsabilité de gérer la création, la mise à jour et la formation de requêtes d'une multitude d'ontologies [GRUBER 1993]. Le format KIF est utilisé aussi bien pour interroger le serveur d'ontologie que pour exprimer les résultats de l'interrogation. Les différents formats d'ontologies sont exprimés via un méta-modèle d'ontologie. Un modèle qui accepte et unifie toutes les représentations (orientée objet, relationnelle, etc) d'une même ontologie.

5.3.4. Agent d'exécution

L'agent d'exécution est responsable de l'exécution haut niveau des requêtes basées sur l'ontologie. Il utilise des techniques de décomposition de tâches ainsi que des techniques de planification.Il décompose les requêtes en sous requêtes et les distribue aux agents ressources pouvant répondre aux besoins de ses sous requêtes. Il réunit ensuite les réponses de ces sous requêtes pour former la réponse à la requête d'origine.

5.3.5. Agent ressource

L'agent ressource est aux données ce que l'agent utilisateur est à l'utilisateur. Cet agent reçoit des requêtes formulées en KQML et les transforme en requête SQL afin d'extraire l'information requise des bases de données.

L'information trouvée est ensuite traduite en KQML en réponse à la requête du demandeur. Comme pour l'agent utilisateur au niveau de l'usager, l'agent ressource garde un contexte des ressources permettant des recherches incrémentielles [DECKER 1997]. Principalement, un agent ressource gère une seule ressource. De toute évidence, plus le nombre d'agents ressource est grand, plus nous aurons un a accès à une information complète et diverse. L'agent ressource utilise JDBC [HAMILTON 1997] comme passerelle pour accéder à différents types de bases de données.

6. Conclusion

Nous avons présenté NetSA, une architecture multiagent dédiée à la recherche d'informations dans des sources de données distribuées. Cette architecture est en cours de développement et elle devrait à un premier prototype, fin 1998 qui doit tourner autour des aspects financiers comme la gestion des prêts bancaires. L'architecture a été développé en ayant à l'esprit : ouverture, généralité et simplicité et de ce fait elle devrait être réutilisable dans bien d'autres domaines. Les lecteurs intéressés par cette architecture peuvent contacter le professeur en charge de ce projet B. Chaib-draa (email : chaib@ift.ulaval.ca).

Références électroniques

http://www.ift.ulaval.ca/~chaib : Notre groupe de recherche, DAMAS (Data-mining, Agents et MultiAgentS).
http://132.203.114.20 :Page de Marc coté.
http://www.ift.ulaval.ca/~troudi/ : Page de Nader Troudi.
http://www.cs.umbc.edu/agents/ : Page agent de UMBC, page de KQML et de KIF.
http://www.dfki.de/mas : DFKI, Centre de recherche allemand sur les systèmes multiagents.
http://www.javasoft.com : Java.
http://www.agent.org/ : Une page très intéressante sur les agents et les systèmes multiagents.

Bibliographie

COHEN Yosef., *Javascript, Cookbook*, wiley, ISBN 0-471-18145-5, 1997, 608 p.

DECKER Keith, SYCARA Katia, PANNU Anandeep et WILLIAMSON Mike., «Designing Behaviors for Information Agents», dans *Proceedings of the First International Conference on Autonomous Agents (AGENTS-97*), Février 1997.

FININ Timothy, LABROU Yannis et MAYFIELD Jim., «KQML as an agent communication language», dans *Software Agents , J.M. Bradshaw, editor, Software Agents*. AAAI Press, 1997.

GENESERETH Michael R. et FIKES Richard E., *Knowledge Interchange Format Version 3 Reference Manual*, Logic-92-1, Stanford University Logic Group, 1995.

GOSLING James, JOY Bill et STEELE Guy., *The Java Language Specification*, ADDISON-WESLEY, ISBN 0-201-63451-1, 1996, 720 p.

GRUBER Thomas., «A translation approach to portable ontology specifications», Knowledge Acquisition, 5(2):199-220, 1993.

HAMILTON Graham et CATTEL Rick., *JDBC™ : A Java SQL API ,Specification Manual Version 1.20,* JavaSoft, Janvier 1997.

HERMANS Bjorn., «Intelligent software agent on the internet: an inventory of currently offered functionality in the information society and a prediction of (near-) future developments». *Master's thesis*, Tiburg University, Tiburg, The Netherlands, Mars 1997.

HUHNS Michael N et SINGH Munindar P., *Readings in AGENTS,* Morgan Kaufmann Publishers, Inc, ISBN 1-55860-495-2, 1997, 523 P.

MAES Patie, «Agents that reduce work and information overload», *Communications of the ACM, vol. 37, no. 7,* juillet 1994, p. 31-40.

IVLER Jean M., *CGI, Developer's Resource*, Prentice Hall Canada, ISBN 0-13-727751-2, 1997, 597 p.

POWEL Thomas A., *HTML : the complete reference*, McGRAW-HILL, ISBN 0-078-82397-8 1998, 912 p.

SYCARA Katia, DECKER Keith et Williamson., <<Matchmaking and Brokering>>, dans *Proceedings of the Second International Conference on Multi-Agent Systems (ICMAS-96)*, Décembre 1996.
VOSS Greg., <<JavaServer™ Technologies, Part II>>, *Technical article JavaSoft*, http://developer.javasoft.com/developer/javaInDepth/java-server2/java-server2.html, 1998.
WOOLDRIDGE Michael et JENNINGS Nicholas R. <<Pitfalls of agent-oriented development>>, *Departement of Electronic Engineering, Queen Mary & Westfield College*, University of London, 1998.