Entretiens Science et Défense,
Cité des Sciences et de l'Industrie,
Paris, 4-5 novembre 1998

Subsidiarité de l'intelligence dans les grands systèmes

Subsidiarity of intelligence in large systems

Jacques LABETOULLE
Professeur à l'Institut Eurécom[1]

Résumé

Les réseaux informatiques sont devenus des systèmes d'une grande complexité qui deviennent extrêmement difficiles à gérer. C'est donc assez naturellement que le problème de leur gestion intelligente s'est posé. De nombreuses techniques ont été suggérées, sans toutefois apporter de solution réellement satisfaisante. Depuis peu, de nouvelles approches sont apparues, riches de promesses, bâties sur les techniques dites d'Agents Intelligents. Cette contribution cherche à démontrer l'intérêt de ces nouvelles techniques.

Abstract

Computer networks have become extremely complexe systems, difficult to manage. Introducing intelligence in their management is a natural way to overcome this difficulty. Many technics have been suggested, none of them being satisfactory. New technics appeared recently, that appear to be promizing, built on the notion of Intelligent Agents. This contribution aims at demonstrating the interest of such new solutions.

1. INTRODUCTION

Les réseaux prennent une place de plus en plus prépondérante dans notre vie, comme dans celle de l'entreprise. Au-delà de cette remarque, nous devons considérer le fait que non seulement les réseaux (et les services qu'ils supportent) sont devenus indispensables, mais qu'un nombre important d'actes de la vie privée comme de la vie professionnelle repose entièrement sur leur bon fonctionnement. On comprend dès lors pourquoi une attention toute particulière doit être portée à la qualité des services rendus.

[1] La recherche à Eurécom est partiellement supportée par ses partenaires industriels : Ascom, Cegetel, France Telecom, Hitachi, IBM France, Motorola, Swisscom, Texas Instruments, et Thomson CSF

Mais les réseaux doivent en premier lieu être considérés comme des systèmes d'une très grande complexité : ce que je veux dire par là est que, même si on a été capable de concevoir individuellement chacune des briques qui constituent le réseau et participent à son fonctionnement, de les tester, de les valider et de les assembler, on reste souvent incapable de comprendre toutes les réactions induites. On a donc construit des systèmes dont on ne maîtrise plus tous les aspects du fonctionnement, principalement quand des situations anormales se produisent.
L'introduction de l'intelligence dans les réseaux provient (du moins en grande partie) de cette constatation et du désir de se faire aider pour mieux maîtriser des systèmes qui sinon risqueraient d'échapper à tout contrôle humain.

Historiquement parlant, la notion d'intelligence dans les réseaux a été introduite pour faire face à la création de nouveaux services de télécommunication sur le réseau téléphonique. Dans ce domaine, il s'est avéré que les éléments de base de ces réseaux, les autocommutateurs, étaient devenus eux-mêmes des systèmes très complexes (plus d'un million de lignes de code). Modifier, ne serait-ce qu'une seule ligne de code, nécessitait un coût prohibitif : cela est dû non pas à la conception et à l'insertion de ce code, mais à l'énorme travail de test et de validation que cette procédure implique, et donc du délai de mise en service (il n'est pas question de mettre en service sur un réseau un matériel qui ne répondrait pas à des critères extrêmement stricts quant à la qualité). On a donc imaginé une architecture dite de réseau intelligent qui, adjointe à l'architecture traditionnelle des réseaux téléphoniques, permet de mettre en œuvre et de modifier de nouveaux services sans modifier une seule ligne de code sur les autocommutateurs. L'intelligence repose alors sur des machines informatiques qui traitent la logique des services, et sont interfacées avec le réseau traditionnel par le biais d'interfaces bien définies implémentées une fois pour toutes. C'est ainsi qu'un certain nombre de services nouveaux ont pu être bâtis (numéros verts, appels facturés sur cartes de crédit ou sur compte d'abonné, utilisation de terminaux intelligents, ...) et qu'une dynamique de création de services a été mise en place.

Cependant, il faut bien constater que le niveau d'intelligence qui résulte de l'architecture dite «réseau intelligent » reste limité : d'abord d'un point de vue pratique (puisqu'elle ne s'adresse a priori qu'aux opérateurs de réseaux publics) mais surtout par ce qu'elle se révèle, à l'usage, très lourde à mettre en œuvre. Cela résulte de la lourdeur des protocoles mis en œuvre (et du poids très fort qu'impose le processus de normalisation internationale), de la rigidité des réseaux auxquels elle s'adresse (le réseau téléphonique), mais sans doute, et pour beaucoup, du manque de maturité dans les domaines objets répartis au moment où elle a été imaginée. Donc cet aspect de l'intelligence dans les réseaux ne sera pas développé dans cet exposé, et nous focaliserons notre attention sur d'autres concepts, sans doute encore très prospectifs par bien des aspects, mais qui démontrent une tendance que je crois irréversible. Ceux-ci s'appuient sur la notion développée récemment d'agent logiciel ou d'agent intelligent.

Mon exposé sera divisé en trois parties : dans un premier temps, je vais brièvement exposer le concept d'agent intelligent (et des variations qui lui sont traditionnellement associées comme les agents mobiles). Sans avoir la prétention d'être un état de l'art exhaustif sur le sujet, j'essayerai de dégager les principes fondamentaux qui régissent le fonctionnement de ces systèmes. Ensuite, je poserai le problème de la gestion des réseaux et de sa complexité (ou du moins des problèmes et difficultés issus de cette complexité). Enfin j'expliquerai pourquoi la gestion de réseau peut tirer un grand bénéfice de l'utilisation du concept d'agent, et ce pour illustrer mon objectif qui est de démontrer en quoi l'utilisation d'agents intelligents est une solution qui permet de maîtriser la complexité des grands systèmes

2. Le concept d'agent intelligent

Dominer la complexité est sans doute aujourd'hui un des grands challenges de tous les grands systèmes. Parmi les techniques possibles, celles qui s'inspirent de la notion d'agent intelligent (IA) semblent les mieux adaptées et les plus aptes à répondre aux besoins. Je ne serais pas à même d'apporter la preuve de cette affirmation dans cet exposé ; tout au plus, puis-je mettre en lumière un certain nombre d'indices pour étayer mon discours.

Tout d'abord, je donnerai une définition (personnelle car le lecteur intéressé pourra en trouver beaucoup d'autres dans la littérature) :

« un agent est une entité logicielle qui :
- agit pour le compte d'autres entités d'une manière automatique
- effectue ses actions avec un certain niveau de réactivité ou de pro-activité
- met en œuvre des propriétés comme : apprentissage, coopération ou mobilité »

Une autre façon d'aborder cette notion est plus globale, en disant qu'un agent intelligent est une entité logicielle qui possède un certain nombre de propriétés parmi les suivantes :

- autonomie : un agent est dit autonome s'il décide seul des moyens qu'il doit mettre en œuvre pour atteindre les buts ou objectifs qui lui sont assignés,
- communication : un agent possède des moyens pour communiquer avec son environnement ainsi qu'avec d'autres agents avec lesquels il coopère
- collaboration/coopération : un agent peut être amené à coopérer ou ,à collaborer avec d'autres agents dans le but de satisfaire les objectifs qu'il doit atteindre (ou que la communauté d'agents doit atteindre),
- délégation : un agent peut déléguer à d'autres agents des buts (s'il juge ceux-ci plus aptes que lui-même),
- apprentissage : un agent peut acquérir de la connaissance sur son environnement,

- mobilité : un agent peut être amené à ce déplacer pour poursuivre son exécution sur un autre site (dans ce cas, à la fois son code exécutable et ses données propres devront être transférées),
- planification : un agent peut avoir la capacité de planifier l'ensemble des tâches qu'il doit mettre en œuvre pour atteindre ses buts,
- pro-activité : un agent peut anticiper des évolutions de son environnement et agir pro-activement pour le maintenir dans un état souhaité,
- réactivité : un agent peut réagir à certaines situations et définir ses actions en fonction de l'état de son environnement,
- sécurité : un agent peut être doté de mécanismes, soit pour sa sécurité propre, soit pour celle de son environnement.

Un agent possédera en général qu'un sous-ensemble de ces propriétés, et il est difficile de dire à partir de quel niveau il s'agit effectivement d'un agent intelligent. Le concept est n'est donc défini que d'une manière approximative. Ceci dit, le concept d'objet, apparu il y a presque 20 ans dans le domaine informatique, ne possède pas non plus de frontières très strictes et cela n'a empêché ni son développement ni l'affirmation de sa pertinence pour facilité la programmation de problèmes complexes. L'agent est quelque chose de différent de l'objet (la notion est d'une certaine manière plus abstraite).

Dans les applications et les expériences menées dans le domaine des agents, on distinguera deux types : celui des agents autonomes (un seul agent, doté d'objectifs pour lui seul) et celui des agents coopératifs.

Le premier type d'agent (les agents autonomes) sont souvent assimilés à ce que l'on appelle des agents d'interface. Ils sont associés, à un instant donné, à un utilisateur et vont effectuer des actions décidées en fonction de la satisfaction de cet utilisateur. Leur rôle est donc d'assister l'utilisateur en communiquant avec le reste du système. Pour cela, il devra acquerir une connaissance sur le profil de l'utilisateur et sélectionnera les fonctionnalités du système à afficher. Ce type d'agent a été particulièrement développé pour faire du filtrage d'information, de la recherche d'information. Ce type d'agent est souvent appelé PDA (pour Personal Digital Agent). Parmi les exemples d'utilisation : filtrage de courriers électronique, de «news » (dans ces deux cas, le rôle de l'agent est passif dans le sens où il n'interagit pas avec le réseau), ou recherche d'information pour le compte de l'usager (et dans ce cas le rôle est actif, car l'agent peut être amené à lancer des actions de recherche d'information). Dans leur utilisation la plus sophistiquées, ce type d'agent construit un modèle de l'usager ainsi qu'un modèle du domaine dans lequel il agit pour mieux répondre aux objectifs (utilisation dans le domaine du commerce électronique).

Le domaine des agents coopératifs est beaucoup plus complexe, mais également beaucoup plus riche en potentialités. Il est largement issu du domaine de la recherche en intelligence artificielle, quand sont apparues les limites de ce domaine. Une des solutions envisagées pour faire face aux

limitations des techniques d'IA a été de distribuer l'intelligence entre des modules, chacun en charge d'une partie du problème. Ce sont ces techniques de distribution de l'intelligence artificielle qui ont donné naissance aux agents intelligents.

Pour donner une image simpliste, on peut considérer un système d'agents comme une population dans laquelle chaque individu est programmé pour une tâche précise et que c'est l 'ensemble des individus qui par leur travail collectif aboutit au résultat recherché (un peu comme dans une fourmilière ou une termitière).

Dans le domaine des agents distribués, deux philosophies peuvent être distinguées :

- celle des agents coopératifs : l'ensemble des agents a été programmé d'une manière homogène, pour un but à atteindre. Dans ce cas leurs coopérations seront elles-mêmes coopératives. Un exemple typique d'application de ce type a été développé dans le contrôle de trafic aérien. On parle de systèmes CMAS (cooperative multi-agent systems).

- celle des agents dits « self interested » : dans ce cas, chaque agent poursuit un but qui lui est propre et qui peut être conflictuel avec les buts des autres agents. Les négociations entre agents peuvent alors être compétitives, et on est souvent amené à utiliser des techniques de type théorie des jeux pour résoudre les conflits. Un exemple typique est celui du commerce électronique où l'agent du client cherche à minimiser le prix alors que les agents des fournisseurs cherchent à maximiser un profit. Ce type d'agents se rencontre dans le domaine des réseaux et services et plus précisément de leurs gestions. On parle alors de systèmes de type SMAS (Self Interested Multi-agent Systems).

Les agents mobiles

Parmi les propriétés déjà évoquées des agents, une est particulièrement importante, au moins par la popularité qu'elle suscite. Il s'agit de la mobilité. Un agent mobile est un agent qui possède d'une part certaines propriétés des agents intelligents (autonomie, communication, réactivité, pro-activité,...) et d'autre part la capacité de migrer d'un site à un autre. Pour assurer cette dernière caractéristique, il doit contenir un ensemble de modèles :

- un modèle agent : ce modèle définira la structure interne de l'agent et les caractéristiques d'agent intelligent qu'il transportera avec lui.
- un modèle de cycle de vie : ce modèle définira les étapes (sur le plan de l'état interne) par lequel devra passer l'agent, en particulier afin de définir à quels moments il pourra migrer. Par exemple le système des Aglets d'IBM définit un agent mobile par un ensemble de tâches et de conditions de transitions entre les tâches. Bien sûr, l'agent ne pourra se mouvoir que s'il

est dans un état endormi (pas de code en cours d'exécution). Il reprendra son activité après son arrivée sur un autre site.

- un modèle d'exécution : c'est la description de la machine (réelle ou virtuelle) supportant l'exécution du code de l'agent. Aujourd'hui, une forte tendance consiste à utiliser du code JAVA, du fait de la disponibilité quasi universelle de la machine virtuelle associée à ce langage.
- un modèle de sécurité : ce modèle est nécessaire pour deux raisons. La première est de protéger les machines hôtes contre les agents malicieux (vol ou altération d'information, vol de ressources ou tout acte de type vandalisme) ; la deuxième est de protéger les agents contre des machines malicieuses (il est à noter que ce dernier domaine est encore en phase d'études, et qu'il n'y a pas de solution à disposition).
- un modèle de communication : les agents ont besoin de communiquer entre eux. Cela exige l'utilisation d'un protocole, mais qui est d'un type particulier. Le problème est de transporter des messages qui sont liés au caractère cognitif de l'information utilisée. Si on peut envisager l'utilisation de moyens de communication classiques (email, RPC,...), il est très souvent fait appel à des langages comme KQML (Knowledge Quiery Manipulation Language) qui répondent à ce besoin.
- un modèle de navigation : c'est ce modèle qui permettra à l'agent (en fonction de son état propre, de sa perception de son environnement,...) de décider quand et vers quel site il doit se déplacer.

Bien entendu, pour pouvoir implémenter des agents mobiles, il faut au préalable installer sur les machines susceptibles de les accueillir un logiciel support qui puisse les recevoir et leur permettre de s'exécuter. Il existe aujourd'hui sur le marché un certain nombre de tels systèmes (ils s'appuient sur des langages comme TCL ou JAVA), tels Telescript de General Majic, les Aglets d'IBM, Voyager et beaucoup d'autres.

A chaque transfert, à la fois le code de l'agent et ses données (les données collectées, l'état de l'agent, en liaison avec son cycle de vie) devront être transportés. Il est parfois possible que le code d'un agent soit résident sur un site si celui-ci est fréquemment visité, conduisant à l'économie du coût de son transfert.
La grande question concernant les agents mobiles est de déterminer soigneusement quand les employer. Typiquement, ils sont intéressants dès qu'il y a une information à compiler à partir d'éléments répartis sur un ensemble de sites, et nécessitant un ensemble de traitements et de tris. Il faut que le coût du transfert se révèle économique par rapport au coût du transport de l'information qu'il serait nécessaire de rapatrier sur un site pour obtenir le même résultat. Cette situation est souvent favorable dans le domaine des réseaux et de leur administration car (voir chapitre suivant) l'information de base (les éléments de MIBs) sont par nature répartis sur un grand ensemble de sites géographiques (l'ensemble de tous les éléments de réseau).

Le lecteur intéressé trouvera des synthèses sur le monde des agents logiciels (intelligents ou non) dans les références [1] et [2], disponibles sur le Web.

3. La gestion de réseaux

Le processus de la gestion de réseau implique un grand nombre de tâches, traditionnellement classées en cinq catégories (ou aires de gestion) : gestion de la configuration, des performances, des fautes, de la sécurité et la gestion financière. Les grands principes de la gestion de réseau ont été mis en place à la fin des années 80 et reposent sur un modèle client-serveur :

- chaque équipement réseau (pris ici au sens matériel ou logiciel) contient un agent. Il s'agit d'une entité logicielle qui maintient une base de données locale contenant des informations sur l'équipement lui-même, et qui sait répondre aux messages qu'on lui envoie (pour lire ou modifier cette base de données). L'agent joue donc le rôle de serveur et se contente de répondre aux sollicitations des gestionnaires (sa seule possibilité d'action non sollicitée est d'envoyer des messages d'alerte quand des conditions exceptionnelles surviennent) : en aucun cas un tel agent ne peut être considéré comme intelligent. Il ne faut bien entendu pas confondre cette notion d'agent (ce vocabulaire est dédié dans le domaine de la gestion de réseaux) avec la notion d'agent (intelligent ou non) telle qu'elle a été introduite dans le chapitre précédent.
- la fonction de gestion est assurée par des machines qui supportent un logiciel appelé manager. Celui-ci exécute des applications ad-hoc de gestion et son système d'information est constitué par l'ensemble des variables fournies par l'ensemble des agents (plus des variables qui lui sont propres). Il obtient des valeurs à jour de ces variables en effectuant des requêtes (périodiques ou à la demande) sur les différents agents.
- Un protocole doit être utilisé pour transporter les informations entre les agents et les managers. Deux solutions existent : une solution OSI (le protocole CMIP – Common Management Information Protocole), et une solution IETF pour les réseaux Internet (le protocole SNMP – Simple Network Management Protocol).

Le logiciel de gestion est donc représenté par le manager, qui en général tourne sur une plate-forme logicielle qui offre deux capacités : celle de faire tourner les applications (en offrant un certain nombre de services de base), et celle de permettre d'en développer de nouvelles (en offrant des API – Application programming Interface).

Sans rentrer dans les détails des architectures support et des applications qui sont disponibles sur le marché, mon but ici est de mettre en lumière les difficultés rencontrées avec les solutions telles qu'elles existent aujourd'hui. Pour simplifier, je dirai qu'elles sont de deux natures : fonctionnelles (liées au fonctionnement de la gestion de réseaux) et conceptuelles (liées aux problèmes de développements).

- Les applications de gestion de réseaux doivent utiliser un nombre impressionnant de variables (de l'ordre de centaines de milliers pour un réseau un tant soit peu important).
- Ces variables sont d'un très faible niveau d'abstraction, obligeant le concepteur à de gros efforts pour structurer ses programmes en fonction de ces variables. Même si la notion d'objet a été introduite dans le monde OSI, elle n'est pas suffisante pour aborder le problème de la conception d'une manière intellectuellement satisfaisante.
- Les APIs de développement sont d'une extrême complexité, nécessitant des informaticiens de haut niveau et une très longue période de formation et d'apprentissage. Les aides méthodologiques pour faciliter leur approche conceptuelle sont inexistantes ou inadaptées.
- Les logiciels de gestion de réseaux sont très centralisés (les approches plates-formes distribuées existent mais sont très difficiles à exploiter en système distribué). Cette propriété a deux inconvénients majeurs : les logiciels de gestion sont énormes, donc difficiles à concevoir, à maintenir et à faire évoluer ; cette architecture nécessite d'énormes quantités d'échanges d'information entre les agents et les gestionnaires, introduisant de ce fait une utilisation importante de la bande passante du réseau et des délais non négligeables avant de pouvoir faire un diagnostic si un incident se produit.
- Les hommes qui administrent les réseaux doivent être des experts hautement qualifiés car les logiciels traditionnels ne permettent pas suffisamment la maîtrise des réseaux et de leur complexité.

4. Les agents intelligents et la gestion de réseaux

Si j'ai mis en évidence dans le chapitre précédent les natures des lacunes que je juge les plus importantes pour la gestion de réseaux, c'est parce que ma conviction est qu'il faut promouvoir une solution radicalement différente, et qui permettra de répondre positivement aux critiques énoncées, à la fois sur le plan fonctionnel et sur le plan conceptuel. Sur quoi donc doit être bâtie cette solution idéale ?

- La solution doit être hautement distribuée pour permettre des analyses locales d'information (donc avec peu d'échanges à travers le réseau)
- La solution doit permettre l'introduction d'intelligence dans le système (tout type de moteur d'intelligence artificielle), car l'homme ne peut pas tout analyser assez rapidement
- La solution doit être hautement évolutive et adaptable car chaque réseau est un cas particulier, et chaque administration de réseau (en fonction des besoins particuliers de ses utilisateurs) doit pouvoir s'adapter
- La solution doit permettre au système d'administration de réseaux d'être réactif, voire pro-actif, pour pouvoir réagir aux situations anormales avec le maximum d'autonomie et garantir la meilleure qualité de service possible

- Le développement d'applications doit être aisé, pour faciliter les développements nouveaux et les adaptations mais également pour faciliter la maintenance
- La solution doit pouvoir rester compatible avec la base installée d'équipements et d'agents, donc compatible avec les normes existantes (SNMP et CMIP)

C'est à ce prix que l'on pourra maîtriser la complexité du système de gestion. Il apparaît bien sûr que la définition des agents intelligents telle qu'elle a été donnée dans le paragraphe 2 semble bien répondre au problème, mais il faut maintenant mieux expliciter comment mettre en œuvre les concepts associés. Pour cela, plusieurs aspects doivent être examinés. Ils concernent :
- le système d'information
- l'architecture d'agents distribués
- les protocoles inter-agents
- l'architecture même de l'agent

4.1 Le système d'information

Les systèmes agents ou multi-agents appuient souvent leur système d'information sur des entités abstraites, construites à base de systèmes de connaissance (cela est une résultante de l'orientation intelligence artificielle, à l'origine de beaucoup de tels développements). Ces entités portent génériquement le nom de catégories mentales, et sont de différents types :

- belief : représente la connaissance d'un agent à propos du monde qu'il observe
- goal : représente la valeur cible que l'on souhaite atteindre ou maintenir pour une connaisance donnée
- motivation : description des objectifs que l'agent doit atteindre
- desires : le monde tel qu'il devrait être selon le point de vue de l'agent
- intention : les désirs que l'agent va réellement chercher à satisfaire
- ...

Dans les systèmes dits intelligents, les enchaînements entre les catégories mentales (comment une motivation génère les goals et les beliefs nécessaires, ...) sont considérés comme devant être automatisés. Dans le cas de la gestion de réseau, le niveau d'abstraction que fournissent les catégories mentales est suffisant pour grandement faciliter la conception des applications qui les utilisent, permettant au concepteur de se dégager de bien des contraintes liées à la rigidité des variables traditionnelles (issues des MIBs), grâce à la souplesse qu'elles autorisent. Plutôt que d'appuyer systématiquement sur des moteurs d'intelligence artificielle, je préconise pour ce type d'applications de définir avec le moins de rigidité formelle possible, des formats relativement libres pour définir les catégories mentales utiles et d'écrire (dans un langage de programmation approprié, et JAVA est aujourd'hui le meilleur candidat) les programmes qui réalisent ces relations ou enchaînements. La pratique a montré que le simple fait de raisonner sur des entités moins formelles et donc

plus abstraites offrait un potentiel énorme pour développer très rapidement des applications très sophistiquées et ce dans un temps relativement court.

4.2 Architecture distribuée d'agents

Dans une architecture efficace d'administration de réseaux, il est souhaitable que le traitement des informations se fasse le plus près possible des équipements fournisseurs de ces informations. Avec une architecture agents, on a une grande liberté pour situer les machines supports des agents là où elles auront accès au plus grand nombre d'équipements, avec des délais de connexion les plus courts. La proximité immédiate des routeurs ou commutateurs haut débit ainsi que les tronçons Ethernet sont donc des endroits privilégiés. Il ne faut pas hésiter à multiplier le nombre de machines, tout en veillant à ce qu'elles gardent sous leur contrôle un nombre suffisant d'équipements. Le prix n'est pas excessif dans la mesure où les machines support sont de simples stations de travail, la majorité d'entre elles pouvant ne pas avoir de clavier/écran.

Le système ainsi construit est résistant aux fautes dans la mesure où une machine peut facilement prendre le relais d'une machine défaillante. Il est donc souhaitable de mettre en place une politique de supervision des agents, effectuée par les agents eux-mêmes, qui se testent mutuellement, et, grâce à un algorithme distribué, peuvent détecter les pannes et réaffecter dynamiquement les domaines de responsabilité. Un tel mécanisme a été testé, et le lecteur intéressé trouvera une description dans [3].

4.3 Les protocoles inter-agents

Le but de ces protocoles est d'une part d'assurer le transfert entre les agents de la connaissance (portée par les « beliefs » suivant le système d'information préconisé), et d'autre part de permettre la délégation d'objectifs (les goals). Il existe des langages qui permettent ce type d'échange, en particulier KQML. Ce type de langage, hérité de la théorie des langages acteurs, permet de transmettre la connaissance, d'interroger sur la présence d'une connaissance, de demander l'exécution d'une opération, ... sous forme de requêtes contenant des instructions telles que « inform », « request », « ask for » .... Pour que la communication ait un sens, il faut que les agents aient au préalable conscience du type d'information échangée, et pour cela définir la sémantique de l'information définie à l'aide d'une ontologie.

4.4 L'architecture de l'agent

Le problème de l'architecture interne de l'agent est critique car de lui dépend la réponse aux deux questions fondamentales portant sur la capacité

fonctionnelle et sur la facilité de développement des applications. Il pose en fait deux problèmes.

Le premier porte sur le caractère générique ou non de cette architecture. Par générique, on entend que l'architecture est indépendante des applications qui seront implémentées sur l'agent. Dans la plupart des expériences en terme d'agents ou d'agents distribués, l'application et l'agent sont développés conjointement et comme un tout, ce qui limite fortement les évolutions possibles. D'un autre côté, vouloir développer une architecture totalement générique, capable de s'adapter et d'être efficace pour tout type de problème est totalement illusoire. Par contre, si on s'attaque à une nature d'applications particulière (comme c'est le cas si on limite ses ambitions à un domaine donné comme l'administration de réseaux) cette approche devient possible. Mon expérience personnelle du problème m'en a du moins apporté la certitude.

Le deuxième problème porte sur la nature de cette architecture. Là, trois approches sont possibles :

- Une architecture d'agent réactif : de tels agents ont un comportement de type réflexe, c'est-à-dire qu'ils sont programmés pour réagir à certains événements ou situations détectés. Une telle approche est assez limitée et ne permet pas d'envisager le problème de l'administration des réseaux d'une manière assez efficace.
- Une architecture d'agent délibératif : dans ce cas, l'agent raisonnera sur les connaissances qu'il a acquises en vue de résoudre les problématiques (objectifs) qui lui sont soumises et planifiera la suite des opérations résultantes. Ce type d'agent est assez fréquent dans la littérature traitant de systèmes d'agents intelligents et le plus souvent s'appuie sur des moteurs d'intelligence artificielle. Par rapport à l'administration de réseaux, une telle architecture est possible mais risque de poser des problèmes d'efficacité (du moins en terme de temps de réponse) quand des problèmes urgents se posent.
- Une architecture hybride, comprenant une partie réactive et une partie délibérative : dans ce cas, l'agent comprend deux couches correspondant aux deux capacités (délibérative et réactive). C'est sans aucun doute la solution la plus efficace pour la gestion de réseaux car une telle solution combine les avantages des deux premières solutions.

Dans notre cas, nous avons donc adopté une architecture hybride. La partie délibérative traite des catégories mentales et s'appuie (sur le plan des connaissances) sur les beliefs. Des modules de traitement sont programmés, qui permettent de déduire des connaissances nouvelles à partir de connaissances élémentaires provenant du réseau lui-même, dans le but de satisfaire les objectifs visés (en terme de connaissances ou d'actions à effectuer). La partie «générique » de cette partie délibérative contient un ensemble d'outils qui permettent d'automatiser les opérations sur les catégories mentales utilisées. La partie réactive comprend trois types d'éléments :

- Des «sensors » : ils permettent d'interagir avec le réseau pour remonter les informations nécessaires à l'élaboration des beliefs. Ils permettent donc d'acquérir de l'information sur le réseau et ses éléments, indépendamment des protocoles utilisés.
- Des «effectors » : ils permettent des actions sur le réseau. A priori, ils permettent tous les types d'opérations permises par les protocoles en agissant directement sur les agents au sein des éléments de réseau, mais ils élargissent les possibilités en autorisant d'autres types d'actions (interactions système ou par telnet, lancement de scripts, et pourquoi pas lancement d'agents mobiles). Cette caractéristique, tout en restant compatible avec les réseaux et les bases de leur administration, permet de s'affranchir des limites des systèmes classiques
- Des «reactors » : ces éléments permettent de programmer des comportements réflexes au sein même de la partie réactive, ce qui autorise une très forte réactivité du système d'administration de réseaux.

Les concepts décrits ci-dessus ne sont pas les seuls que l'on rencontre dans la littérature traitant de la gestion de réseaux par agents intelligents. Un article synthétique sur les différentes techniques utilisées est référencé dans [4].

4.5 La réduction de la complexité

Résumons les caractéristiques principales des agents intelligents tels qu'ils ont été introduits, et essayons de comprendre en quoi cette approche permet de mieux maîtriser la complexité inhérente à l'administration des réseaux.

- L'utilisation de données possédant un haut niveau d'abstraction (les catégories mentales) permet au concepteur de construire ses applications sans tenir compte les contraintes particulières des protocoles. Il a la possibilité de définir des éléments de connaissance très appropriés aux problèmes qu'il a à résoudre et donc de faciliter son approche conceptuelle. La façon dont seront instrumentées les variables représentant la connaissance devient un problème séparé du problème initial, et l'utilisation des « sensors » permet, à travers la couche réactive de l'agent, d'assurer la bonne alimentation en information des « beliefs ».
- La possibilité de traiter des problèmes localement (donc d'une manière isolée) permet grâce aux techniques de délégation, de hiérarchiser un problème initial complexe et le décomposant en un ensemble de sous-problèmes beaucoup plus simples.
- La technologie distribuée permet des traitements proches des sources d'information, limitant ainsi le trafic dans le réseau et permettant de traiter une plus grande quantité de données sans pénaliser la bande passante.
- Un système de gestion bâti sur le principe agent permet une grande souplesse de mise en place, et en particulier assure une sécurité et une résistance aux fautes irréalisables avec un système centralisé.

- L'introduction de techniques d'intelligence artificielle, dont l'efficacité a été démontrée pour certains types de problèmes (comme le filtrage d'alarmes) est parfaitement naturelle dans un environnement agents.
- La technologie de base (celle qui est utilisée par un agent générique) est infiniment plus simple qu'une plate-forme classique d'administration de réseau.

Les arguments que je donne ci-dessus ne peuvent en fait être démontrés que par l'expérience. Dans la pratique, les techniques évoquées ont effectivement été expérimentées, et exploitées pour réaliser certaines fonctions de gestion de réseau (détection des agents défectueux et reconfiguration du système agents, gestion de connexions sur réseau haut débit, contrôles d'admission et détection d'intrus). Toutes ces expériences se sont révélées relativement faciles à définir et à mettre au point, et d'une manière beaucoup plus aisée que si on avait adopté une approche classique.

5. Conclusion

Les techniques présentées ici n'ont certes pas encore fait leur preuve d'une manière indiscutable. Il apparaît cependant qu'elles apportent un bénéfice certain, bien que difficilement mesurable, à un problème crucial. Des études restent à mener pour conforter et prouver plus formellement les certitudes affichées ici (en particulier pour une meilleure mesure de l'efficacité en terme de facilité de conception, de capacité fonctionnelle et de performance). Il semble également souhaitable de développer des méthodes de conception adaptées au monde agents comme il en existe pour le monde objets.
La technologie agents a, par de nombreuses expériences, montré son efficacité pour aborder des problèmes spécifiques. Nous avons montré ici qu'elle était une alternative crédible mais aussi efficace pour un problème de nature très complexe comme celui de la gestion de réseaux. Ma conviction est qu'elle peut également se révéler pertinente pour d'autres classes de problèmes complexes, et nécessite d'être prise en compte dès que la nature des difficultés rencontrées atteint les limites du savoir-faire actuel.

Bibliographie

[1] Nwana H.S., « software Agents : an overview ». *Knowledge Engineering Review*, Vol. 11,n°3, pp. 205-244, October/November 1996. http://www.cs.umbc.edu/agents/introduction/ao/.

[2] Green S. and all « Software Agents : a review », Technical report, May 97, http://www. cs.tcd.ie/research_groups/aig/iag/pubreview.ps.gz

[3] Marcus K. « Diagnosing Intelligent Agents for Network management » submitted to IM'99

[4] Cheikhrouhou M., Conti P., Labetoulle J., « Intelligent Agents in Network Management, a State-of-the-Art », *Networking and Information Systems*, Vol1 – n° 1/ 1998, pp. 1-29