Ramonaite A., Moreno C., Hutzler G., "BoWeMA : une architecture multi-agent pour l'accélération d'Internet", in Journées Francophones d'Intelligence Artificielle Distribuée et Systèmes Multi-Agents 2001, Montréal (Canada), pp. 347-349, Hermès, Paris, 2001.

BoWeMA : une architecture multi-agent pour l'accélération des flux de données au travers de l'Internet

Ausra Ramonaite[1,2], Carlos Moreno[1] et Guillaume Hutzler[1]

[1]*Laboratoire de Méthodes Informatiques*
Université d'Evry Val d'Essonne
Bd. François Mitterrand, 91025 Evry Cedex
{r amona, moreno, hutzler}@lami.univ-evry.fr

[2]*BoostWorks*
1 rue de Terre Neuve, Bat L, Miniparc du Verger
91967 Les Ulis Cedex
aramonaite@boostworks.com

RÉSUMÉ. *Le grand essor d'Internet a fait apparaître de nouvelles opportunités de développement de systèmes multi-agents (SMA) notamment pour le commerce électronique, la recherche d'information, et récemment l'accélération de réseaux. Dans cet article, nous présentons une architecture multi-agent distribuée, appelée BoWeMA (BoostWeb Multi-Agent) pour l'accélération des flux de données au travers de l'Internet. Cet architecture est basée sur les concepts d'Accélération Intelligente des Réseaux (« Intelligent Network Acceleration »), comprenant l'analyse des données, leur optimisation, leur compression et leur mémorisation dynamique. Le modèle de communication proposé, conforme aux spécifications de FIPA ACL et permet de mettre en place une organisation souple et efficace des communications entre les agents.*

ABSTRACT. *The exponential growth of the Internet has revealed new opportunities for the development of multi-agent systems (MAS) in fields such as electronic commerce, information searching and recently network acceleration. In this paper, we present a distributed multi-agent architecture, called BoWeMA (BoostWeb Multi-Agent), for data flow acceleration through the Internet. It is based on Intelligent Network Acceleration concepts, including data flow analysis, optimisation, compression and dynamic memorisation. The proposed communication model, complying to FIPA ACL standard, offers flexible and efficient messaging, which is very important in this Internet solution.*

MOTS-CLÉS : *Système multi-agent, Accélération.*

KEY WORDS: *Multi-agent system, Internet acceleration*

1. Introduction

L'approche multi-agent [FER 95] constitue une voie prometteuse pour le développement de différents systèmes, tels que des systèmes de télécommunication et de contrôle, diverses applications médicales, commerciales et industrielles, des simulateurs, etc. Le grand essor d'Internet ces dernières années a fait apparaître de nouvelles opportunités pour le développement des systèmes multi-agents (SMA) dans les domaines du commerce électronique, de la recherche d'information, et récemment de l'accélération des flux d'information au travers des réseaux. En effet, aujourd'hui Internet souffre de son succès. Il est souvent saturé et les périodes de latence deviennent de plus en plus longues.

La plupart des solutions actuelles, telles que l'amélioration de l'infrastructure des réseaux (l'installation d'équipements supplémentaires, l'accroissement de la largeur de la bande passante) ou la réécriture des applications logicielles fournissent une certaine flexibilité en résolvant les problèmes, auxquels les utilisateurs d'Internet sont confrontés. Cependant, ils sont coûteux et complexes à déployer et à maintenir.

L'objectif de notre recherche est de proposer une solution complète pour l'accélération des flux d'information au travers de l'Internet, incluant les principes suivants : (i) la distribution pour éviter les goulets d'étranglement du système, (ii) la coopération entre les différents éléments du système pour éviter le chevauchement de leurs actions, et (iii) la communication entre les éléments du système pour assurer la coordination de leur travail. Dans ce contexte, l'approche multi-agent offre les concepts appropriés pour atteindre cet objectif. Nous proposons ici une architecture multi-agent distribuée, appelée BoWeMA (BoostWeb Multi-Agent), pour l'accélération des flux d'information au travers de l'Internet. Le modèle proposé de communication, basé sur les spécifications de FIPA ACL, assure les échanges flexibles et efficaces d'information entre les agents.

Cet article est organisé comme suit. Dans la section 2, nous donnons un brève aperçu des travaux du domaine. Dans la section 3, nous détaillons l'architecture proposée BoWeMA ainsi que son modèle de communication. Nous terminons cet article par la présentation des perspectives sur ce travail.

2. Travaux du domaine

Avec la croissance rapide des flux d'information et du nombre des utilisateurs, l'accès à l'information sur l'Internet devient de plus en plus difficile et entraîne notamment des temps de latence de plus en plus élevés. La plupart des solutions traditionnelles résolvant ce problème de latence, se focalisent sur l'installation de nouveaux équipements sur le réseau, l'ajout de fonctionnalités de compression sur les routeurs ou la réécriture des applications logicielles pour en optimiser les performances (par exemple, la réplication du contenu, l'utilisation du cache, etc.) [ALL 99]. Ces solutions résolvent bien certains des problèmes, auxquels les utilisateurs sont confrontés. Cependant, elles sont coûteuses et complexes à déployer et à maintenir.

L'optimisation d'un réseau basée sur des outils logiciels permet d'améliorer les performances de l'Internet en réduisant le trafic HTTP d'un bout à l'autre du système. Un outil d'optimisation peut produire des résultats immédiats en améliorant les performances des réseaux par l'accélération des temps de connexion et de réponse et en diminuant le volume du trafic. L'amélioration des performances allège la pression sur les équipements de réseau (les modems, les proxies et les routeurs). BoostWorks[1] a développé tel outil entièrement logiciel, appelé BoostWorks Optimizer [BOO 00], qui permet d'accélérer l'affichage des pages Web en réduisant le trafic sur les réseaux. Il accélère les performances en tirant avantage des hauts volumes de trafic redondant dans les architectures de réseaux et ceci sans modification de la configuration existante. L'INA (« Intelligent Network

[1] Start-up française, spécialisée dans la création de logiciels, basés sur la technologie d'Accélération Intelligente de Réseaux, qui permet d'accélérer la performance de réseaux tout en préservant l'intégrité et la qualité des données.

Acceleration ») constitue la technologie d'optimisation sous-jacente pour ce produit. Elle permet d'analyser les données traversant le réseau, de les optimiser et d'appliquer à ces données la compression la plus adaptée, tenant en compte les habitudes de l'utilisateur. Les différentes fonctionnalités du traitement de flux de données (la compression, l'optimisation) sont réalisées de manière centralisée, ce qui peut poser problème lorsque des traitements longs sont nécessaires, et non-intelligente (par exemple, la vérification de l'utilité de la compression des données n'est pas assurée).

Nous proposons dans cet article une version distribuée de cet outil d'optimisation qui doit permettre le traitement simultané d'un grand nombre de requêtes, donc d'obtenir une meilleure régulation des flux de données en évitant de bloquer le système lorsqu'un traitement long doit avoir lieu. Dans ce cadre, nous proposons d'utiliser un ensemble d'agents plus ou moins spécialisés et possédant collectivement l'ensemble des compétences nécessaires au traitement du flux de données, ces agents devant fonctionner ensemble pour se répartir le travail à chaque fois qu'une nouvelle requête est reçue.

3. BoWeMA : Système multi-agent distribué pour l'accélération d'Internet

Le système BoWeMA est conçu pour faire l'interface entre un client, utilisateur de données (par exemple, un navigateur Web), et un serveur, fournisseur de ces mêmes données (par exemple, un serveur HTTP). Pour ce faire, BoWeMA filtre les échanges de données entre les deux de manière à les optimiser. Les requêtes effectuées par un navigateur Web sont en effet très redondantes, les mêmes données pouvant être redemandées plusieurs fois consécutivement. Dans la plupart des navigateurs Web, c'est rôle du cache de conserver en mémoire les données accédées récemment de manière à pouvoir les resservir le plus rapidement possible et en économisant la bande passante. Avec le système BoWeMA, nous proposons une architecture multi-agent distribuée de gestion de cette mémoire cache qui doit permettre d'en accélérer et d'en optimiser le fonctionnement.

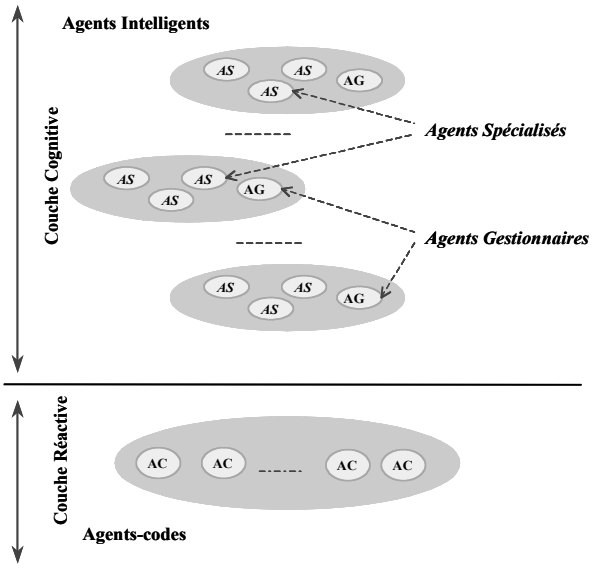

Figure 1. Architecture générale de BoWeMA.

3.1. Architecture du système

L'architecture de BoWeMA (BoostWeb Multi-Agent), basé sur les concepts d'INA de BoostWorks, comprend deux couches (une couche réactive et une couche cognitive), intégrant deux types d'agents (Figure 1) : *les agents-codes*[2] (AC) dont la tâche principale consiste à exécuter des actions prédéfinies, telles que la compression ou l'optimisation de flux des données, et *les agents intelligents* (*les agents spécialisés* (AS) et *les agents gestionnaires* (AG)) qui effectuent l'analyse des données reçues ainsi que leur traitement « intelligent » de manière distribuée.

Les agents spécialisés sont divisés en plusieurs groupes selon la spécificité des tâches qu'ils ont à effectuer, et leur degré de compétence. Cette compétence n'est pas forcement supérieure ou inférieure à celle des agents d'un autre groupe, mais elle est indispensable pour réaliser l'objectif du groupe d'agents en question.

Chaque groupe d'agents possède un agent gestionnaire, permettant de contrôler les états des agents de manière indirecte. Sa tâche consiste à stocker les informations concernant les changements d'états des agents du groupe, transmis par les agents eux-mêmes, et à gérer l'occupation du groupe en question.

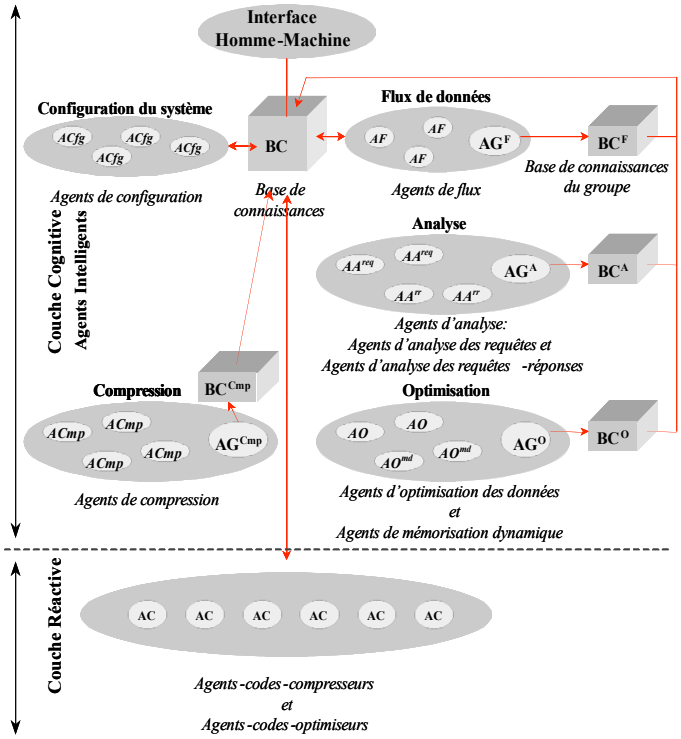

Figure 2. *La structure organisationnelle de BoWeMA.*

[2] D'après [BER 00]

3.2. *Structure organisationnelle*

Chaque fois qu'une donnée est demandée par le navigateur, le traitement correspondant requière la coopération de différents agents aux compétences variées qui se transmettent mutuellement les requêtes ou les données afin de satisfaire à la requête initiale du navigateur. Cela permet d'identifier une succession de traitements élémentaires dans laquelle les agents spécialisés analysent les flux de données de manière de plus en plus précise jusqu'à déterminer finalement le type d'agent de compression ou d'optimisation nécessaire.

La structure organisationnelle de BoWeMA distingue différents rôles, qui sont attribués aux agents en tenant en compte de leurs fonctions spécifiques au sein du système. Nous définissons les rôles suivants (voir Figure 2 ci-dessus) :

- **Configuration du système** : les agents de configuration (ACfg) gèrent les différents paramètres du système, qui sont accessibles aux autres agents par l'intermédiaire d'un base principale de connaissance (BC).
- **Flux de données** : la tâche principale des agents de flux (AF) est d'identifier la nature du flux (requête du navigateur Web ou réponse du serveur HTTP) et de le rediriger vers les agents ou les acteurs du système (le navigateur ou le serveur HTTP) correspondants. Les requêtes sont soit des requêtes ordinaires, qui ne contiennent pas de données (par exemple, la demande de chargement d'une page Web envoyée par le navigateur), soit des requêtes accompagnées des données (par exemple, la demande de mise à jour d'une page Web effectuée par un navigateur). De même, les réponses peuvent être ou non accompagnées de données. Les réponses avec données correspondent généralement aux envois par le serveur HTTP des pages Web demandées par le navigateur ; les réponses sans données, correspondent aux réponses négatives du serveur HTTP concernant les pages Web demandées.
- **Analyse**, comprenant deux sous-rôles (**Analyse des requêtes et Analyse des requêtes-réponses**) : les agents d'analyse des requêtes (AA^{req}) traitent les requêtes ordinaires, envoyées par le navigateur et relayées par les agents de flux. Leur tâche principale est de vérifier si l'information demandée par le navigateur est stockée dans la mémoire cache. Les agents d'analyse des requêtes-réponses (AA^{rr}) traitent deux types de flux : les requêtes avec les données, envoyées par le navigateur, et les réponses, envoyées par les serveur HTTP, contenant les données. Ils identifient le type de flux (HTML, JPEG, GIF, etc.) et déterminent le traitement qui doit lui être associé.;
- **Optimisation**, comprenant deux sous-rôles (**Optimisation des données** et **Mémorisation dynamique**) : l'optimisation des données vise à obtenir des flux d'information plus homogènes, en éliminant les éléments redondants et insignifiants et en préservant le format original. En ce qui concerne la mémorisation dynamique, il s'agit de la mise en mémoire cache des données les plus fréquemment requises dans leur version originale et optimisée. Ceci permet de servir ces données plus rapidement si elles sont demandées à nouveau.
  Les agents d'optimisation des données (AO) identifient la nature de l'information reçue (le texte, l'image, etc.) et la redirigent vers l'agent-code-optimiseur correspondant (l'agent-code, capable de réaliser l'optimisation d'un certain type de données). Les agents de mémorisation dynamique (AO^{md}) placent les données dans leur version originale et optimisée dans la base de données, qui joue le rôle de « cache ».
- **Compression** : La compression des données inclut la compression des fichiers de type texte ou binaire ainsi que la dégradation des images (GIF, JPEG, PNG) par rapport à la qualité souhaitée.
  La tâche principale des agents de compression (ACmp) est d'identifier la nature d'information reçue (le texte, l'image, etc.) et de la rediriger vers l'agent-code-compresseur correspondant (l'agent-code, capable de réaliser la compression du type de données concerné).

3.3. *Exemple : Procédure du transfert des données avec BoWeMA*

Pour mieux illustrer le fonctionnement du système BoWeMA, nous analysons l'exemple suivant : le navigateur envoie une requête ordinaire demandant le chargement d'une page Web. Le système BoWeMA reçoit cette requête et la transmet au serveur HTTP. Il reçoit en retour les données demandées (la page Web), les traite et puis les transmet au navigateur.

L'exemple décrit ci-dessus peut être décomposé en plusieurs étapes [RAM 00] :

- **Etape 1 : La réception d'un requête, provenant du navigateur** : le navigateur ouvre la connexion vers le système BoWeMA et lui envoie sa requête. Celle-ci est reçue par un des agents de flux (AF) inoccupés. Ce dernier informe l'agent gestionnaire du groupe (AG) à propos du changement de son état, puis analyse l'information reçue grâce à sa base de connaissance et décide du traitement qui doit lui être appliqué. Dans ce cas précis, il s'agit d'une requête ordinaire. Donc l'agent AF la transmet à un des agents des requêtes (AA^{req}) et informe de nouveau l'agent AG à propos du changement de son état.
  Il faut préciser que tous les agents spécialisés informent leurs agents gestionnaires à propos de chaque changement de leurs états (en général, dans les cas suivants : (i) après avoir reçu une information provenant d'un autre agent ou d'un acteur du système (le navigateur Web ou le serveur HTTP), et (ii) après avoir transmis l'information traitée à un autre agent) ;
- **Etape 2 : L'analyse d'une requête** : après la réception de la requête ordinaire, provenant de l'agent AF, l'agent AA^{req} vérifie si l'information demandée est stockée dans le « cache ». Pour ce faire, il envoie une demande de vérification, à un des agents de mémorisation dynamique (AO^{md}). Dans ce cas précis, nous considérons que le « cache » ne contient pas l'information demandée. L'agent AA^{req} reçoit la réponse négative de l'agent AO^{md}, et il transmet la requête reçue auparavant à un des agents AF inoccupés.
- **Etape 3 : L'envoi d'une requête au serveur HTTP** : puisque la données n'a pas pu être trouvée dans le cache, l'agent AF transmet la requête au serveur http concerné après avoir établi une connexion avec lui.
- **Etape 4 : La réception d'une réponse, provenant du serveur HTTP** : en réponse à cette requête, soit la donnée est disponible et le serveur HTTP la délivre en retour à un des agents AF, soit elle ne l'est pas et il renvoie une réponse négative. Dans ce cas précis, nous supposons qu'elle était disponible et il s'agit alors d'une réponse contenant des données (la page Web) que l'agent AF transmet donc à un des agents d'analyse des requêtes-réponses (AA^{rr}).
- **Etape 5 : L'analyse d'une réponse** : à la réception d'une réponse, provenant d'un agent AF, l'agent AA^{rr} effectue plusieurs actions d'analyse (par exemple, la vérification de la nature de l'information reçue ou la vérification des capacités du navigateur de décompresser des données) pour pouvoir décider du traitement de l'information reçue. Dans ce cas précis, il s'agit d'une réponse contenant des données de type texte (la page HTML), et le navigateur est capable de décompresser les données de ce type. L'agent AA^{rr} transmet donc la réponse à un des agents d'optimisation des données (AO).
- **Etape 6 : Le traitement d'une réponse** : lorsque l'agent AO reçoit l'information, il envoie la copie des données reçues (leur version originale) à un des agents AO^{md}, en demandant de les stocker dans la mémoire cache. L'agent AO analyse ensuite les données de manière à sélectionner les agents d'optimisation puis de compression les mieux adaptés. Dans ce cas précis, l'agent AO fait appel à l'agent-code optimiseur, capable de réaliser l'optimisation des données de type texte. L'agent AO envoie ensuite la version optimisée des données à un des agents de compression (ACmp) qui fera appel à l'agent-code compresseur spécialisé correspondant (adapté aux données de type texte).
  Une fois la compression effectuée, l'agent ACmp envoie une copie des données optimisées et compressées à l'agent AO^{md}, pour stockage dans la mémoire cache. Ces données sont ensuite transmises à un des agents AF, en indiquant de les transmettre au navigateur, qui a fait la demande initiale.

- **Etape 7 : L'envoi d'une réponse au navigateur** : à la réception d'une réponse, l'agent AF cherche le navigateur, qui a fait une demande, et lui transmet la réponse.

3.4. *Modèle de communication multi-agent*

Dans l'architecture de BoWeMA, nous considérons la communication comme une action d'échange d'information entre les différents agents par l'envoi de messages point à point. Deux types de communication sont identifiés :

- La communication « verticale » entre les agents spécialisés ayant les différents niveaux de compétence (par exemple, entre l'agent d'analyse des requêtes-réponses et l'agent d'optimisation des données) ou entre les agents spécialisés et les agents-codes. Dans le premier cas, l'agent spécialisé communique avec un autre agent inoccupé (en général, le premier agent attendant dans la queue des agents inoccupés du groupe). Dans le deuxième cas, l'agent spécialisé communique directement avec un agent-code particulier.
- La communication « horizontale » entre les agents spécialisés et les agents gestionnaires (par exemple, entre l'agent d'analyse des requêtes et l'agent gestionnaire du groupe).

Pour que les agents puissent communiquer efficacement, il est nécessaire qu'ils utilisent un langage de communication commun [LAB 00]. Dans le domaine de l'accélération d'Internet, les agents hétérogènes doivent être capables d'échanger les messages standardisées d'une manière flexible. Les agents doivent réagir rapidement sans perdre trop de temps pour des actions de communication. Dans l'architecture BoWeMA, tous les agents communiquent en envoyant des messages asynchrones, conformes à la spécification de FIPA ACL [FIPA 97] [OBR 98] [FIPA 00]. Possédant une sémantique claire et indépendante de la structure des agents, ce langage de communication permet de mettre en place des interactions primitives aussi bien que des interactions complexes, qui nécessitent l'établissement de protocoles élaborés.

La description des messages des agents dans l'architecture BoWeMA correspond à la grammaire suivante :

```
<MESSAGE>=<Type_de_message><Paramètres_de_message>
```

Le champ `<Paramètres_de_message>` inclut les identificateurs de l'agent émetteur et de l'agent récepteur, l'expression signifiant une action antérieure, pour laquelle ce message est une réponse, le contenu du message, la description du langage employé ainsi que l'identificateur du protocole. Le champ `<Type_de_message>` identifie le type d'acte communicatif, qui se réfère à une action que l'on demande à l'agent récepteur de réaliser. Dans l'architecture BoWeMA les agents utilisent les actes communicatifs suivants :

- « `inform` » : ce type de message permet à l'agent spécialisé soit d'informer l'agent gestionnaire du groupe à propos de son changement d'état, soit d'envoyer la réponse concernant les résultats d'exécution d'une action demandée auparavant (par exemple, l'agent de mémorisation dynamique informe l'agent d'analyse des requêtes à propos des résultats concernant la vérification de l'information dans le cache. Dans le premier cas, le contenu du message contient la description d'état de l'agent, et l'agent gestionnaire ajoute ce contenu dans la base de connaissances du groupe. Dans le deuxième cas, le contenu du message contient la réponse (positive ou négative) concernant l'exécution de l'action demandée auparavant.
- « `request` » : il s'agit d'une demande d'exécution d'une certaine action. Par exemple, l'agent de flux envoie le message « `request` » à l'agent d'analyse des requêtes, en demandant d'analyser la requête correspondante. Le contenu du message contient la description de la requête en question.

- « `refuse` » : il s'agit du refus d'un agent d'exécuter une action demandée. Par exemple, l'agent d'optimisation des données envoie le message « refuse » à l'agent d'analyse des requêtes-réponses, en refusant d'optimiser les données reçues. Le contenu du message contient la description d'une action demandée et l'explication d'un refus.
- « `not-understand` » : envoyant ce type de message, l'agent émetteur (par exemple, l'agent d'optimisation des données) informe l'agent récepteur (par exemple, l'agent d'analyse des requêtes) qu'il n'a pas compris le message précédent, envoyé par cet agent. Le contenu du message contient la description d'une action ou d'un acte communicatif incompris et l'explication des raisons d'incompréhension.
- « `failure` » : envoyant ce type de message, l'agent émetteur (par exemple, l'agent d'optimisation des données) informe l'agent récepteur (par exemple, l'agent d'analyse des requêtes) que l'accomplissement de l'action demandée par cet agent, a échoué pour certaines raisons. Le contenu du message contient la description d'une action échouée et l'explication des raisons d'échec.

3.5. *Exemple : les interactions entre les agents pendant la procédure du transfert des données avec BoWeMA*

L'exemple que nous prenons pour illustrer les interactions entre les différents agents est celui présenté dans la partie 3.3. : le navigateur envoie une requête ordinaire et il reçoit les données (la page Web) en retour.

Comme dans l'exemple précédent, nous analyserons les différentes étapes, indiquant les interactions entre les différents agents :

- **1) La réception d'une requête, provenant du navigateur** : le navigateur ouvre la connexion avec le système BoWeMA et lui envoie sa requête. Celle-ci est reçue par un des agents AF inoccupés. Ce dernier envoie le message « `inform` » à l'agent AG (cf. Figure 3 (1)) pour l'informer du changement de son état. Après l'analyse de l'information reçue, l'agent AF envoie le message « `request` » à un des agents AA^{req} (cf. Figure 3 (2)) en demandant d'analyser la requête envoyée par le navigateur, et puis informe de nouveau l'agent AG à propos du changement de son état. Tous les agents spécialisés informent leurs agents gestionnaires à propos de chaque changement de leurs états en envoyant les messages « `inform` ».
- **2) L'analyse d'une requête** : pour vérifier si l'information demandée est stockée dans le « cache », l'agent AA^{req} envoie le message « `request` » à l'agent AOmd (cf. Figure 3 (3)). Après la vérification, ce dernier envoie à l'agent AA^{req} le message « `inform` » dont le contenu correspond, dans ce cas précis, à la réponse négative (cf. Figure 3 (4)). L'agent AA^{req} envoie ensuite le message « `request` » à un des agents AF, en demandant de transmettre la requête reçue auparavant au serveur HTTP capable d'y répondre (cf. Figure 3 (5)).
- **3) La réception d'une réponse, provenant du serveur HTTP** : après avoir reçu la réponse (la page Web) provenant d'un serveur HTTP, l'agent AF envoie le message « `request` » à un des agents AArr, en demandant d'analyser l'information reçue (cf. Figure 3 (6)).
- **4) L'analyse d'une réponse** : dans ce cas précis, il s'agit d'une réponse contenant des données de type texte (la page HTML), et le navigateur est capable de décompresser les données de ce type. L'agent AA^{rr} envoie donc le message « `request` » à un des agents AO, en demandant d'optimiser les données (cf. Figure 3 (7)).

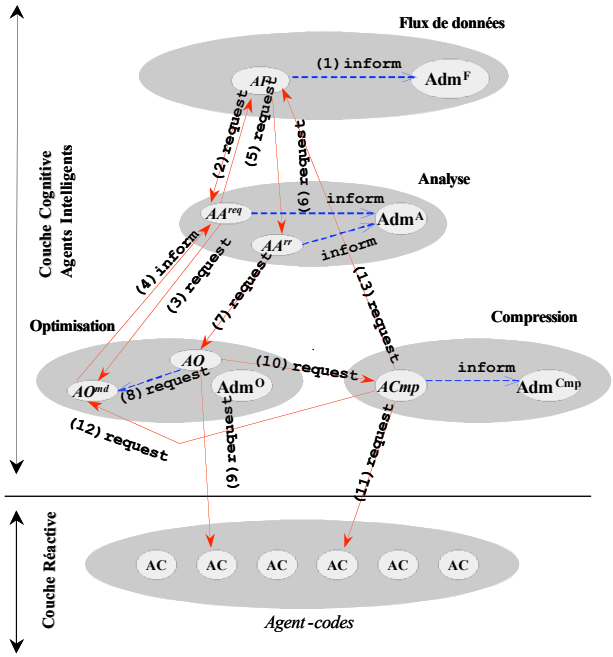

Figure 3. *Les interactions entre les différents agents dans BoWeMA (les flèches en traits correspondent aux interactions entre les agents de différents groupes, et les flèches en tirets montrent les interactions entre les agents d'un même groupe).*

- **5) Le traitement d'une réponse** : lorsque l'agent AO reçoit l'information, il envoie le message « `request` » à un des agents AO^{md}, en demandant de stocker la version originale de l'information reçue dans la mémoire cache (cf. Figure 3 (8)). Ensuite l'agent AO fait appel (le message « `request` ») à l'agent-code optimiseur, capable de réaliser l'optimisation des données de type texte (cf. Figure 3 (9)). L'agent AO envoie le message « `request` » à un des agents ACmp en demandant de compresser la version optimisée des données (cf. Figure 3 (10)). Ce dernier fera appel (le message « `request` ») à l'agent-code compresseur spécialisé, adapté aux données de type texte (cf. Figure 3 (11)).
  Une fois la compression effectuée, l'agent ACmp envoie deux messages « `request` » : le premier à l'agent AO^{md}, en demandant de stocker les données optimisées et compressées ans la mémoire cache (cf. Figure 3 (12)), et le deuxième à un des agents de AF, en demandant de transmettre la réponse au navigateur, qui a fait la demande (cf. Figure 3 (13)).

4. Conclusions et perspectives

Dans cet article nous avons présenté une architecture multi-agent distribuée, intitulée BoWeMA (BoostWeb Multi-Agent), pour l'accélération des flux applicatifs au travers de l'Internet. En comparaison des approches traditionnelles, notre approche présente les avantages suivantes :

- Le traitement distribué des problèmes simultanés et entièrement corrélés ;
- Une grande résistance à l'instabilité de l'environnement grâce à l'attribution dynamique des tâches ;
- L'évolution du nombre des agents dans un tel système au cours du temps.

Nous sommes actuellement en phase d'implémentation et de validation expérimentale de l'architecture BoWeMA. La modélisation complète des diagrammes de séquences en UML a permit d'identifier les classes répondant aux cas d'utilisation précis. Pour la validation de l'architecture BoWeMA, nous avons décidé d'utiliser une plate-forme agent existante, la plate-forme JADE [BEL 99] [JADE] (« Java Agent Development Framework »), conforme notamment aux spécifications FIPA.

Puisqu'il s'agit d'un travail en cours d'implémentation, il nous reste encore à valider l'utilisation de l'architecture multi-agent dans le contexte des flux applicatifs naturellement distribués via l'Internet. Cette validation devrait ouvrir la voie à de très nombreuses applications autour du développement d'Internet, et plus généralement pour tout ce qui concerne les échanges de données via des réseaux informatiques et de télécommunications.

Remerciements

Les auteurs tiennent à remercier Frédéric Thiré (BoostWorks) pour son constant support sur cette recherche.

5. Bibliographie

[BOO 00] BOOSTWORKS, BoostWeb Optimizer, v.3, Web page : http://www.boostworks.com, 2000, document visité le 20/05/01

[ALL 99] ALLIANE R., " Des composants intelligents pour l'accélération des réseaux : modélisation et spécification ", Mémoire de DEA, Université d'Evry Val d'Essonne, Sept. 1999

[BEL 99] BELLIFEMINE F., RIMASSA G., " JADE : a FIPA-compliant agent framework ", *The Forth International Conference on the Practical Application of Intelligent Agents and Multi-agents*, London, England, 1999, p.97-108.

[BER 00] BERNON C., GLEIZES M.-P., " L'agentification du code en perspective ", *4ème Ecole Informatique des Systèmes Parallèles et Réparties : ISPYR'2000*, 2000.

[FER 95] FERBER J., *Les systèmes multi-agents*, InterEditions, 1995

[FIPA 00] The Foundation for Intelligent Physical Agents. FIPA ACL Message Structure Specification, 2000

[FIPA 97] The Foundation for Intelligent Physical Agents. FIPA 97 specification: Agent Communication Language, Part 2, Version 2.0, 1997

[JADE] JADE, http://sharon.cselt.it/projects/jade, page Web visitée le 25/05/01

[LAB 00] LABROU Y., FININ T., " History, State of the Art and Challenges for Agent Communication Languages ", *Informatik - Informatique*, n°1, 2000, p.17-24.

[OBR 98] O'BRIEN P.D., NICOL R.C., " FIPA – towards a standard for software agents ", *BT Technology Journal*, vol.16, n°3, 1998, p.51-59.

[RAM 00] RAMONAITE A., " Contribution à la modélisation de systèmes multi-agents d'une architecture communicante pour l'accélération de flux applicatifs au travers de l'Internet ", Rapport d'activité, Université d'Evry Val d'Essonne, Nov. 2000.