06/2010-E/INF

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université des Sciences et de la Technologie
''Houari Boumedienne''

Faculté d'Electronique & Informatique

THÈSE

Présentée pour l'obtention du grade de

DOCTEUR d'ETAT en INFORMATIQUE

Par

Youcef ZAFOUNE

Étude de l'Impact de la Technologie d'Agent Mobile dans la Gestion des Réseaux Mobiles Ad hoc

Soutenue publiquement le : 23/12/2010 devant le jury composé de :

Mr Badache Nadjib, Professeur à l'USTHB	**Président,**
Mme Mokhtari-Aissanni Aïcha, Professeur à l'USTHB	**Directrice de thèse,**
Mr Koudil Mouloud, Professeur à l'ESI	**Examinateur,**
Mr Naït-Abdesselam Farid, Professeur à l'université Paris V	**Examinateur,**
Mr Azzoune Hamid, Maître de conférences à l'USTHB	**Examinateur,**
Mme Moussaoui Samira, Maître de conférences à l'USTHB	**Examinatrice.**

Remerciements

Je tiens à remercier vivement Mme Aïcha Mokhtari-Aissani, Professeur à l'USTHB, et lui exprimer toute ma reconnaissance pour avoir encadré mon travail de thèse, pour ses conseils, ses encouragements et sa patience pendant ces longues années.

Je remercie particulièrement le Professeur Nadjib Badache pour m'avoir fait l'honneur de présider le jury de ma soutenance.

Je remercie les Professeurs Mouloud Koudil de l'ESI et Farid Naït-Abdesselam de l'université de Paris Descartes, pour l'intérêt qu'ils portent à mon travail en acceptant d'êtres examinateurs de cette thèse.

Je remercie Mr Hamid Azzoune et Mme Samira Moussaoui, Maîtres de conférences à l'USTHB, pour l'intérêt qu'ils portent à mon travail en acceptant d'être examinateurs de cette thèse.

Je remercie aussi Mr Rushed Kanawati, Maître de conférences à l'université de Paris Nord, avec qui j'ai le plaisir de travailler ainsi que l'équipe du LIPN de la même université pour m'avoir accueilli au sein de leur laboratoire.

Je remercie également tous mes collègues enseignants du département d'Informatique avec qui j'ai le plaisir de travailler.

Pour finir, je remercie tous ceux qui m'ont aidé de près ou de loin pour concrétiser ce travail.

Dédicaces

A toute ma famille,
à tous mes amis,
je dédie cette thèse.

Table des matières

Chapitre 3 : Technologie d'agent mobile

Chapitre 4 : Problème de localisation d'objets dans les réseaux

Liste de figures

Liste des tables

Résumé

La gestion des réseaux mobiles ad hoc est une tâche extrêmement difficile, comparé aux réseaux filaires, et ceci à cause de leurs propres caractéristiques. Ces réseaux doivent donc disposer de leurs propres systèmes de gestion, puisque les systèmes de gestion classiques pour les réseaux filaires ne leurs sont plus adaptés. Ainsi, le défi majeur des systèmes de gestion pour les réseaux ad hoc est de parvenir à proposer une qualité de service à ses utilisateurs, comparables à ceux des réseaux classiques et cela de façon transparente.

Les architectures à base d'agents mobiles représentent une alternative intéressante dans la gestion des réseaux ad hoc, puisqu'ils permettent un mode de fonctionnement déconnecté et asynchrone, une possibilité de gain en bande passante et une répartition des tâches de calculs. De plus, leur autonomie leur permet de s'adapter facilement aux changements dynamiques de contexte.

Dans cette thèse, nous avons proposé une nouvelle approche de gestion pour les réseaux mobiles ad hoc, de type semi-centralisée et basée sur la technologie d'agent mobile. Notre alternative permet de concevoir un système de gestion multi-serveurs, centralisé et mobile, dans un réseau où il n'est pas évident de penser à une gestion centralisée, dû à l'absence de toute administration ou infrastructure fixe dans ces réseaux. L'un des objectifs de l'approche semi-centralisée proposée, est de fournir une permanente disponibilité de services dans les réseaux ad hoc caractérisés par une gestion distribuée. Ceci en maintenant les serveurs du réseau dans une région centrale, rapidement accessible par les nœuds du réseau, tout en permettant aux différents serveurs un travail coopératif efficace. La mobilité des agents est exploitée afin de permettre une flexibilité et une auto-organisation de notre système. Les agents mobiles qui forment un système multi-agent constituent un serveur virtuel mobile, dans le sens il n'est pas lié en permanence à des nœuds du réseau. Les composants du serveur virtuel peuvent changer de support physique lorsque cela est nécessaire, et ce, pour offrir au mieux ses services aux utilisateurs. Une autre contribution de notre approche est une nouvelle structuration hiérarchique du réseau en zones, afin de limiter l'étendue des diffusions des informations, mais aussi pour assurer une coopération efficace et rapide entre les composants (serveurs de zones) du serveur virtuel du réseau.

Dans le but d'évaluer les performances de notre approche, nous l'avons appliqué au problème de localisation du code mobile dans les réseaux ad hoc. Il s'agit ici de localiser un

objet à double mobilité, celle de son support physique dans un environnement dynamique, la station, et celle de son code à travers les stations du réseau. La gestion de la localisation qui est une étape très importante dans la gestion de la mobilité dans les réseaux mobiles, permet de fournir au réseau des informations sur la position courante d'un objet mobile ainsi que la mise à jour de ces informations. Le mécanisme de localisation est utilisé généralement dans la première phase de routage (notamment pour les protocoles de routage dit géographiques), permettant d'assurer la communication entre les nœuds mobiles malgré le changement permanant de leurs positions. La localisation permet aussi de chercher auprès de ressources (logiciels ou matériels) fournies dans le réseau.

Abstract

The management of mobile ad hoc networks is an extremely difficult task, in contrast with wired networks, as a result of their own characteristics. These networks must therefore have their own management systems since the standard management systems for wired networks no longer fit them. Thus, the major challenge facing management systems for ad hoc networks is to manage to offer their users a quality of service, similar to standard networks and achieve this in a transparent way.

The architectures based on mobile agents represent an interesting alternative for the management of ad hoc networks, since they allow a disconnected and asynchronous functioning mode, a possibility that reduces bandwidth and offers a distribution of reckoning tasks. Moreover, their autonomy allows them to easily adapt to contextual dynamic changes.

In this thesis, we have proposed a new management approach for ad hoc networks, of the semi-centralized type and based on the mobile agent technology. Our alternative makes it possible to devise a management system with different centralized and mobile management servers, in a network where it is not obvious to think of a centralized management, owing to the absence of any administration or fixed infrastructure in the networks. One of the aims of the proposed semi-centralized approach is to supply the availability of permanent services in the ad hoc networks characterized by a distributed management. This is achieved by maintaining the servers of the network in a central region, thus, rapidly accessible by the network stations, while allowing the different servers to perform an efficient cooperative work. The mobility of the agents is taken advantage of in order to allow both flexibility and self-organization to our system. The mobile agents that constitute a multi-agent system make up a virtual mobile server, in the sense that it is not permanently tied to some stations of the network. The components of the virtual server can change their physical support whenever it is necessary, and this to offer for the best its services to the users. Another contribution of our approach is a new zone structuring hierarchy of the network, in order to reduce the broadcasting extent of information, but also to ensure an efficient and swift cooperation of the components (zone servers) of the virtual servers of the network.

In order to evaluate the performance of our approach, we have applied it to the problem of the mobile code localization in ad hoc networks. One has to locate in this context an object with a double mobility, that of its physical support in a dynamic environment, its station, and

that of its code through the network stations. The management of the localization which is a very important stage of the management of mobility in mobile networks allows the supplying of the network with information about the current position of a mobile object as well as the updating of these different information. The localization mechanism is generally used in the first routing phase (especially for the routing protocols called geographical), allowing the communication between the mobile stations in spite of the permanent changing of their positions. The localization mechanism allows as well to research among the resources (software or hardware) supplied in the network.

Chapitre 1

Introduction générale

1. Introduction

Les réseaux informatiques sont en train de devenir indispensables pour tous les domaines de la vie. Ils sont devenus des éléments stratégiques pour les entreprises et les particuliers. Alors que de nombreuses activités reposent sur les réseaux informatiques, il est donc nécessaire de les gérer pour veiller à leurs bon fonctionnement mais aussi afin d'exploiter au mieux les ressources disponibles, de rentabiliser au maximum les investissements réalisés, et d'offrir une qualité de services aux utilisateurs.

De nos jours, les réseaux informatiques sont de plus en plus grands et plus complexes. La gestion des réseaux est devenue un point critique par l'hétérogénéité même des éléments composant le réseau et par la nature caractérisant certains environnements de réseaux, tel que la mobilité ou les liens sans fils. Comparé avec les réseaux informatiques filaires, les réseaux mobiles doivent donc disposer de leurs propres systèmes de gestion. En effet, la gestion des réseaux mobiles est une tâche extrêmement difficile, notamment pour les réseaux mobiles ad hoc caractérisés par une absence de toute administration centralisée ou infrastructure fixe.

Le modèle client/serveur qui est le plus largement répandu dans la construction d'applications réparties pour les réseaux d'ordinateurs, se base sur des interactions distantes permettant à un client d'envoyer une requête de service à un serveur. Les différentes méthodes permettant de réaliser ces requêtes (envoi de message, appel à distance et invocation distante) ont montrées leur efficacité dans les environnements stables tels que les réseaux filaires. Cependant, l'introduction de la mobilité physique, due aux technologies sans fil, apporte un degré de dynamisme qui ne semble pas permettre à ces méthodes d'atteindre le même niveau d'efficacité. Toutefois, elles peuvent être améliorées en introduisant la mobilité du code [27].

Dans ce cadre, les agents mobiles représentent un paradigme intéressant pour les environnements mobiles. En effet, leur autonomie leur permet de s'adapter facilement aux changements dynamiques de contexte et à l'hétérogénéité des éléments présents [27]. De plus,

ce paradigme permet un mode de fonctionnement déconnecté et asynchrone (très avantageux pour les stations mobiles), une possibilité de gain en bande passante (très convoitée dans les réseaux sans fils), et une répartition des tâches de calculs (favorisant la conception d'architectures décentralisées).

Ainsi, dés les premières utilisations de la technologie d'agent mobile dans les réseaux mobiles au niveau applicatif (conception d'applications utilisateurs pour les environnements mobiles), la communauté scientifique s'est rapidement intéressée à leurs utilisations au niveau contrôle. Les architectures centralisées des systèmes de contrôle et de gestion de réseau ayant montré leurs limites pour les environnements mobiles, a favorisé l'émergence des architectures distribués et hiérarchiques, mais aussi celles basées sur les agents mobiles. Cependant, les agents mobiles se heurtent à de grandes difficultés de développement, se voient opposer leurs problèmes de sécurité toujours d'actualité et induisent un deuxième niveau de mobilité, i.e., la mobilité logicielle.

2. Evolution des réseaux mobiles ad hoc et applications

Ces dernières années, le besoin en mobilité des usagers n'est plus à démontrer. En effet, la chute des prix du matériel de transmission sans fil, l'augmentation de l'autonomie des batteries et l'augmentation des performances de la miniaturisation des processeurs de forte puissance, ont considérablement contribué à l'apparition d'unités mobiles de petite taille. Les réseaux mobiles ad hoc ou MANETs (*Mobile Ad hoc NETworks*) offrent une alternative très intéressante permettant aux usagers de communiquer tout en se déplaçant librement. Dans un réseau ad hoc sans fil, les terminaux (ou nœuds) mobiles peuvent communiquer sans requérir la présence d'une infrastructure particulière, lorsqu'ils sont à portée d'émission et de réception l'un de l'autre.

Le premier réseau ne reposant sur aucune liaison filaire (PRNet) a été construit par la DARPA en 1987. Naturellement, les protocoles radio de l'époque étant limités, la bande passante disponible était faible. Cependant, il constitue l'une des bases des réseaux mobiles ad hoc actuels. Les réseaux ad-hoc représentent l'extension des réseaux radio classiques : ils représentent littéralement des réseaux prêts à l'emploi [114]. Ils peuvent également être utilisés lorsque le déploiement d'infrastructures est coûteux ou sans objet, ou encore pour accroître la portée des réseaux à base d'infrastructure.

Diverses technologies sans fil (par exemple, Bluetooth, IEEE 802.11, HomeRF et HiperLAN) ont été proposées, sur lesquelles s'appuient les réseaux ad hoc. Plusieurs classifications des réseaux sans fil sont présentées dans la littérature, selon que l'on s'intéresse à un critère ou à un autre. La figure suivante [6] présente une classification possible selon l'étendue géographique et l'infrastructure utilisée.

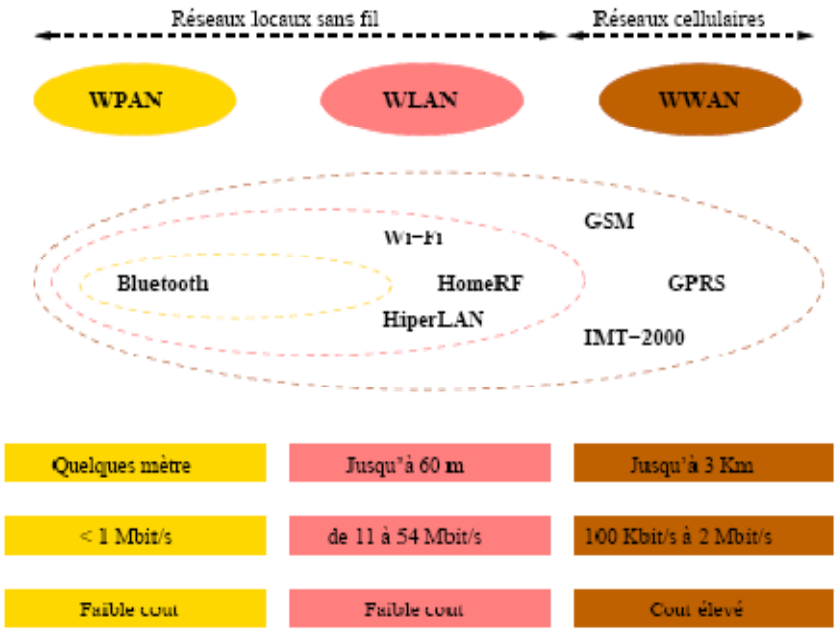


Figure 1 : *Les réseaux sans fil.*

Une autre classification, très utilisée dans la littérature est celle qui décompose les réseaux sans fils et mobiles en deux classes principales [5] :

- les réseaux avec infrastructure fixe
- et les réseaux sans infrastructure.

2.1. Réseaux avec infrastructure fixe ou réseaux cellulaires

Les réseaux mobiles avec infrastructure fixe sont des réseaux structurés, basés sur des équipements d'interconnexion faisant office de ponts entre un réseau radio et un réseau câblé permettant ainsi à de nombreux utilisateurs mobiles d'accéder à des ressources informatiques. Ils sont composés de deux ensembles d'entités distinctes :

- Les "sites fixes" du réseau filaire (*wired network*).
- Les "sites mobiles" (*wireless network*).

Certains sites fixes, appelés Stations de Base (SB) sont munis d'une interface de communication sans fil pour la communication directe avec les sites ou Unités Mobiles (UM), localisées dans une zone géographique limitée appelée cellule (figure 2).

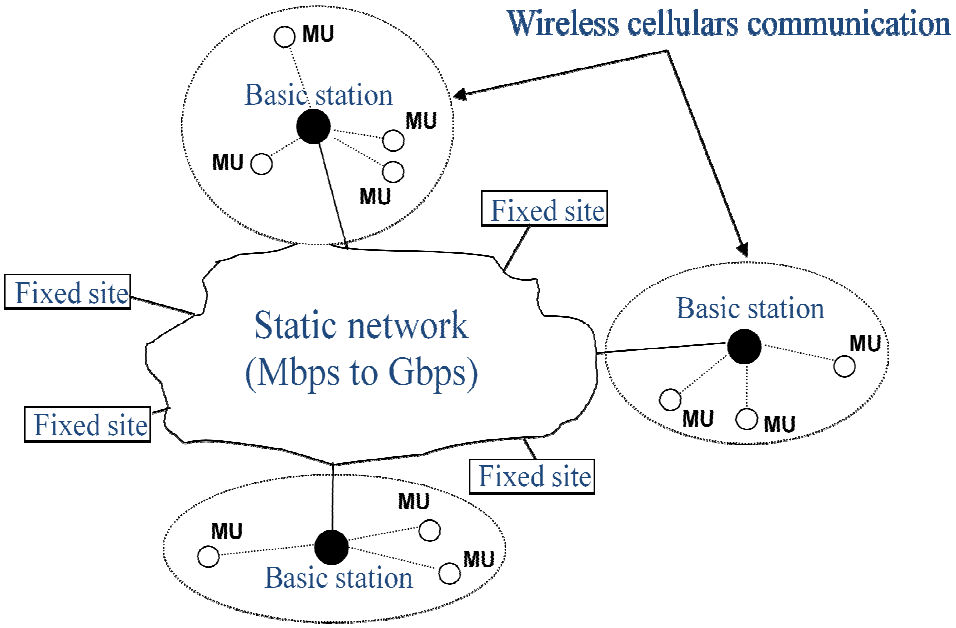


Figure 2 : *Réseaux cellulaires.*

2.2. Réseaux sans infrastructure ou réseaux ad-hoc

Dans le modèle des réseaux sans infrastructure préexistante, l'entité « site fixe » n'existe pas, tous les sites du réseau sont mobiles et communiquent entre eux d'une manière directe, en utilisant leurs interfaces de communication sans fil.

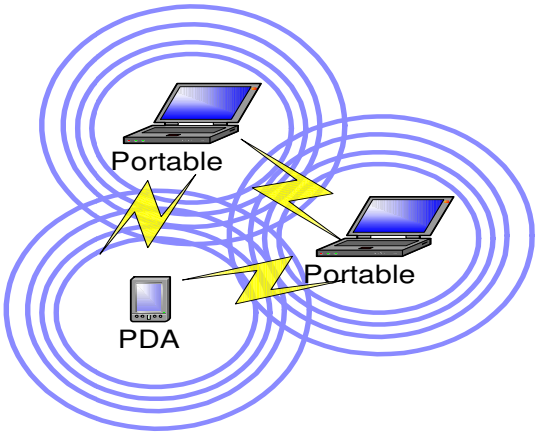


Figure 3 : *Exemple d'un réseau ad hoc.*

Les réseaux mobiles « ad hoc », ou réseaux « spontanés », sont des réseaux radio aptes à se construire et à s'organiser dynamiquement quand plusieurs équipements se trouvent à portée radio les uns des autres; n'importe quel outil mobile muni d'un microprocesseur peut potentiellement être un nœud d'un réseau ad hoc. Par conséquent, la topologie du réseau peut varier de manière rapide et surtout imprévisible du fait de la mobilité permanente de ses

composants. A un moment donné, un nœud mobile peut joindre ou quitter le réseau de façon dynamique. On parle alors de topologie dynamique pour les réseaux ad hoc.

Un nœud mobile peut communiquer directement avec un autre nœud s'il est dans sa portée de transmission. Au-delà de cette portée, les nœuds intermédiaires jouent le rôle de routeurs pour relayer les messages saut par saut. La route entre un nœud source et un nœud destination peut impliquer plusieurs sauts sans fil, d'où l'appellation de « réseaux sans fil multi-sauts ».

2.3. Domaines d'applications des réseaux ad hoc

Les réseaux ad hoc représentent donc une alternative intéressante aux réseaux à infrastructures, notamment pour les applications distribuées qui sont dynamiquement déployées, comme par exemple les applications dans le domaine militaire, pour lesquelles il n'est pas envisageable d'installer des infrastructures. Bien que les projets aient souvent débuté dans un cadre militaire pur, le domaine d'application des réseaux ad hoc s'étend bien au-delà [6].

Actuellement, ils sont exploités dans des contextes d'application divers et des environnements variés. En effet, ils ont l'avantage d'offrir une grande flexibilité ainsi qu'une rapidité et facilité de mise en place. Ils sont d'un grand apport dans de nombreuses situations dont les opérations d'urgence (catastrophes naturelles, incendies et tremblements de terre) impliquant la destruction totale ou partielle des infrastructures du réseau fixe, où il sera indispensable de disposer rapidement d'un réseau pour organiser les secours et les opérations de sauvetage. D'autre part, l'utilisation des réseaux ad hoc est préconisée dans le contexte quotidien (réunions de travail ou collaboration fortuites de personnes, maison communicante) ou lors de l'organisation d'évènements (des conférences, des salons), dès lors que les besoins ne justifient pas de déployer des infrastructures onéreuses.

3. Motivations de la thèse

Avec la récente évolution des technologies de communication sans fil, les réseaux locaux sans fil ont connu un développement rapide et leur utilisation est de plus en plus fréquente. Toutefois, de nombreux défis doivent être relevés pour que les réseaux mobiles ad hoc puissent être effectivement exploités par les utilisateurs, et pouvoir supporter les mêmes applications que les réseaux filaires et cela de façon transparente.

Le défi, lié aux caractéristiques intrinsèques des réseaux ad hoc, que nous allons aborder dans cette thèse est la problématique de gestion de ce type de réseaux. En effet, les systèmes de gestion classiques ne peuvent pas être appliqués directement dans les réseaux mobiles puisqu'ils s'adaptent mal à ce type d'environnement, et plus particulièrement aux réseaux ad hoc. De nombreux travaux de recherche ont été alors effectués qui présentent d'autres systèmes de gestion de type distribués et hiérarchiques, spécifiques pour les réseaux ad hoc.

Notre contribution dans cette thèse consiste en une proposition d'une nouvelle approche de gestion basée sur la technologie d'agent mobile. Notre alternative permet de concevoir des

serveurs centralisés et mobiles dans un réseau, où il n'est pas évident de penser à une gestion centralisée, dû à l'absence de toute administration ou infrastructure fixe dans ces réseaux. L'un des objectifs de l'approche centralisée proposée, est de fournir une permanente disponibilité de services dans les réseaux ad hoc caractérisés par une gestion distribuée. Ceci en maintenant les serveurs du réseau dans une région centrale, rapidement accessible par les nœuds du réseau, tout en permettant aux différents serveurs un travail coopératif efficace. La mobilité des agents, est exploitée afin de permettre une flexibilité et une auto-organisation de notre système. Les agents mobiles qui forment un système multi-agent constituent un serveur virtuel mobile, dans le sens il n'est pas lié en permanence à des nœuds du réseau. Le serveur peut changer de support physique lorsque cela est nécessaire, et ce, pour offrir au mieux ses services aux utilisateurs. Par exemple, lorsque des nœuds sont en manque d'énergie ou s'éloignent de la région centrale, supposée permettre une disponibilité permanente aux clients, le serveur réorganise ses agents en les invitant à migrer vers d'autres nœuds plus appropriés, assurant ainsi une continuité du service malgré la mobilité des nœuds supportant le serveur.

Dans le but d'évaluer les performances de notre approche, nous l'avons appliqué au problème de localisation du code mobile dans les réseaux ad hoc. Il s'agit ici de localiser un objet à double mobilité, celle de son support physique dans un environnement dynamique, la station, et celle de son code à travers les stations du réseau. La gestion de la localisation qui est une étape très importante dans la gestion de la mobilité dans les réseaux mobiles, permet de fournir au réseau des informations sur la position courante d'un objet mobile, tout en assurant la mise à jour de ces informations. Le mécanisme de localisation est utilisé généralement dans la première phase de routage d'information dans les réseaux ad hoc (notamment pour les protocoles de routage dit géographiques), permettant d'assurer la communication entre les nœuds mobiles malgré le changement permanant de leurs positions. La localisation permet aussi de chercher auprès de ressources (logiciels ou matériels) fournies dans le réseau.

Dans un réseau ad hoc, les approches préconisées pour localiser les ressources, se basent sur des techniques de diffusion [97]. Tous les nœuds ou la plupart d'entre eux sont mis à contribution dans le mécanisme de localisation, en conséquence, toutes les approches présentées sont de type distribuées. Cependant, une approche entièrement distribuée, basée sur la diffusion des requêtes et/ou des réponses dans l'ensemble du réseau, accroît l'encombrement dans un réseau caractérisé par une bande passante étroite. Il est nécessaire de proposer de nouvelles solutions afin de garantir une localisation efficace des ressources dans un réseau ad hoc, i.e. prenant en compte les contraintes caractéristiques des réseaux ad hoc, à savoir principalement, les changements de topologie du réseau, le manque de connectivité, la bande passante réduite, et les capacités limitées des terminaux. De plus, les solutions proposées doivent fournir à l'utilisateur un niveau de qualité de service satisfaisant, ce qui signifie en particulier, offrir un temps d'attente réduit pour l'utilisateur et une disponibilité permanente du service. C'est dans ce dernier contexte que se situe notre contribution à travers notre approche basée sur un serveur multi-agents mobile.

Une autre contribution de notre approche est une nouvelle structuration hiérarchique du réseau en zones, afin de limiter l'étendue des diffusions des informations, mais aussi pour assurer une coopération efficace et rapide entre les composants (serveurs de zones) du serveur virtuel du réseau. Effectivement, cette approche limite le trafic induit par la localisation, d'une part en ciblant les hôtes (serveurs de zones) vers lesquels sont dirigés les requêtes, et d'autre part en ne diffusant pas de requêtes de localisation dans l'ensemble du réseau (au niveau zone seulement).

4. Organisation du document

La suite de ce document est constituée de six chapitres. Dans le chapitre 2, nous dressons un état de l'art des travaux portant sur les systèmes de gestion de réseaux, en mettant l'accent sur l'utilisation des agents mobiles. Les caractéristiques de la technologie d'agents mobiles et un état de l'art de leurs plates-formes d'exécution seront présentés dans le chapitre 3. Le chapitre 4 décrit le mécanisme de localisation d'objets mobiles dans les réseaux informatiques, et présente les principaux protocoles et services de localisation proposés dans différents contextes de réseaux. Une approche de type distribuée, que nous avons proposé pour la localisation d'agents mobiles dans un réseau ad hoc, sera décrite dans le chapitre 5. Une autre approche proposée pour la localisation, de type semi-centralisée et basée sur la technologie d'agents mobiles, sera détaillée et étudiée dans le chapitre 6. La dernière partie de ce document, conclut et projette les perspectives de notre travail de thèse.

Chapitre 2

Gestion des réseaux informatiques

1. Introduction

Un réseau ne peut être complet sans une gestion du réseau et un contrôle de ses ressources [85]. La gestion des réseaux informatiques constitue un problème dont l'enjeu est de garantir au meilleur coût non seulement la qualité du service global rendu aux utilisateurs mais aussi la réactivité face aux besoins de changement et d'évolution. Elle se définit comme étant l'ensemble des moyens mis en œuvre (connaissances, techniques, méthodes, outils) pour superviser, exploiter des réseaux informatiques et planifier leur évolution en respectant les contraintes de coût et de qualité [77].

Une des ambitions du monde des réseaux est de parvenir à proposer une qualité de service à ses utilisateurs. Elle se décline sur plusieurs critères, du point de vue de l'utilisateur final, notamment la **disponibilité**, la performance (comme le temps de réponse ou le taux de perte de paquet), la fiabilité, la sécurité, etc.

Dans ce chapitre, après avoir rappelé les principales fonctionnalités de gestion des réseaux informatiques, nous présenterons un état de l'art des travaux sur les systèmes de gestion des réseaux, en mettant l'accent sur l'utilisation de la technologie d'agent mobile, que ce soit dans la fonction de gestion proprement dite, ou dans les protocoles de routage et le clustering des réseaux, en particulier pour les réseaux mobiles ad hoc.

2. Fonctionnalités de gestion

Les fonctionnalités d'administration et de gestion des réseaux informatiques sont communément classées en activités de [85, 101]:

- ***Supervision*** qui consiste à surveiller les systèmes et à récupérer les informations sur leur état et leur comportement, ce qui peut être fait par interrogation périodique ou par remontée non sollicitée d'informations de la part des équipements de réseaux eux-mêmes.
- ***Administration*** qui désigne plus spécifiquement les opérations de contrôle, de coordination et de surveillance des différentes ressources mises en œuvre, afin de fournir

des services opérationnels aux utilisateurs du réseau, avec la gestion des configurations et de la sécurité.

➢ ***Exploitation*** qui désigne l'ensemble des activités permettant de traiter les problèmes opérationnels sur le réseau : maintenance, assistance technique, etc.

L'OSI a regroupé ces activités en cinq groupes fonctionnels [77, 85, 101] (voir figure 4):

✓ ***La gestion de la configuration*** consiste à maintenir un inventaire précis des ressources matérielles et logicielles, nécessaires à l'initialisation et au lancement du réseau, et d'en préciser la **localisation** géographique. Il s'agit également d'associer, à chaque objet géré, un nom qui l'identifie de manière unique. Pour gérer une configuration, quatre procédures sont nécessaires :

- Collecte des informations sur l'état du système.
- Contrôle de l'état du système.
- Sauvegarde de cet état en établissant un historique.
- Présentation de l'état du système.

✓ ***La gestion des anomalies*** dont l'objectif est le diagnostic rapide de toute défaillance interne ou externe du système et de tout dysfonctionnement ; ces pannes pouvant être d'origine interne résultant d'un élément en panne ou d'origine externe dépendant de l'environnement du système (coupure d'un lien publique ou présence d'un goulet d'étranglement, par exemple). Cette gestion implique la surveillance des alarmes (signalisation du fonctionnement anormal), le traitement des anomalies (réparation et confirmation du retour au fonctionnement normal), la localisation et le diagnostic des incidents, la journalistique des problèmes, etc.

✓ ***La gestion des performances*** consiste à contrôler et à évaluer la performance et l'efficacité des ressources du système, à savoir le temps de réponse, le débit, le taux d'erreur par bit, le taux de perte de paquet, la **disponibilité** (aptitude à écouler du trafic et à répondre aux besoins de communication pour lequel la ressource a été mise en service). L'évaluation des performances nécessite la collecte d'information statistique (mesure du trafic, temps de réponse, taux d'erreurs, etc.) afin de déterminer, en permanence, si le réseau est apte à satisfaire les besoins de communication des utilisateurs. Elle comprend aussi le stockage et l'interprétation des mesures (archivage des informations statistiques, calculs de charge du système, tenue et examen des journaux chronologiques de l'état du système). La mesure de la dégradation des performances permet d'anticiper les défaillances et de programmer les évolutions du système. La gestion des performances fournit ainsi la fonction d'optimisation permettant de réduire les coûts et d'améliorer l'utilisation du matériel.

✓ ***La gestion de la sécurité*** concerne le contrôle d'accès au réseau, la confidentialité des données qui y transitent, leur intégrité, leur authentification et le non-désaveu.

✓ ***La gestion de la comptabilité*** consiste à gérer la consommation réseau par abonné, de relever les informations permettant d'évaluer les coûts d'utilisation des ressources du

réseau. Ces informations peuvent être très utiles dans la configuration du réseau et l'allocation de ses ressources. La comptabilité est établie en fonction du volume et de la durée des transmissions.

Suivant le type de réseau, les tâches de gestion varient et recouvre aussi d'autres opérations, telles que la gestion des ressources, l'analyse, la planification, etc.

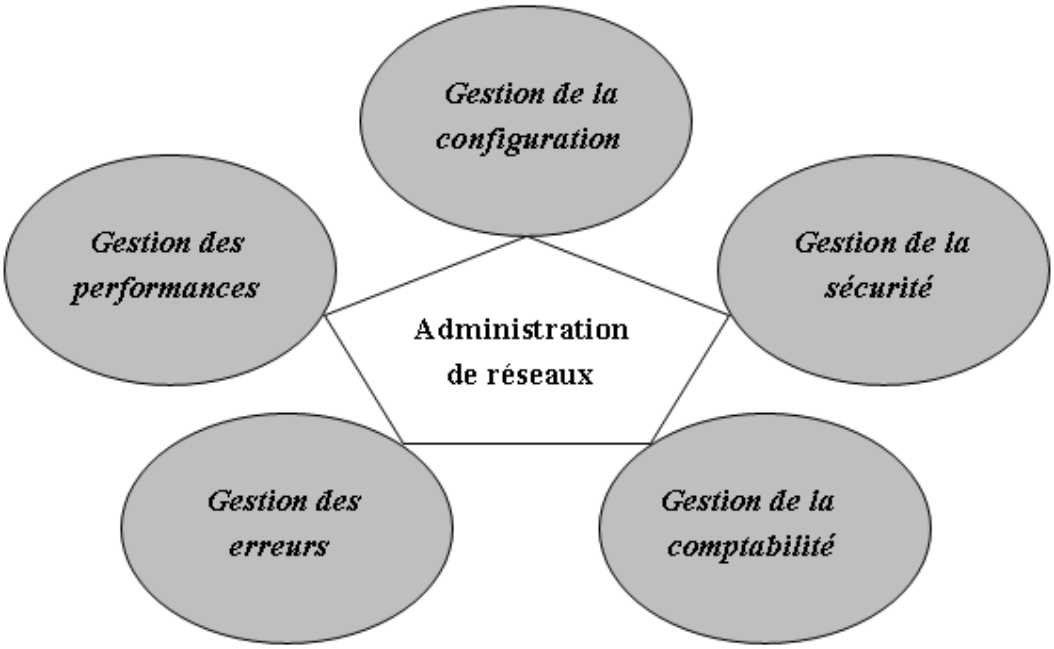


Figure 4 : *Activités de gestion d'un réseau informatique.*

3. Etat de l'art des architectures et plateformes de gestion de réseaux

Selon la stratégie de communication et la collecte des informations, il y a trois types de modèles de gestion de réseaux : le modèle centralisé, le modèle distribué et le modèle hiérarchique.

Dans un système de gestion centralisé, il y a une seule station gérante (*manager*) qui collecte les informations à partir de tous les nœuds et contrôle le réseau en entier. C'est une architecture très simple mais pose un certain nombre de problèmes classiques liés à la centralisation, tel que la défaillance de la station gérante et sa congestion dans le cas de larges réseaux ; aussi, lorsque un partitionnement du réseau survient à cause d'un certain type de coupure, la partie du réseau déconnectée de la station gérante se trouve sans fonctionnalités de gestion. C'est le cas des réseaux mobiles ad hoc qui souffre en plus d'une grande surcharge de messages dans la collecte de données.

Un système de gestion distribué est constitué de plusieurs stations gérantes; chacune gère un sous-réseau et communique directement avec les autres stations gérantes du système. Utilisant une approche distribuée, le système de gestion de réseau assure une grande fiabilité et efficacité aussi bien qu'une faible surcharge des ressources lors des communications et des

traitements. Ainsi, ce modèle est meilleur pour les réseaux très larges (réseaux ATM et de télécommunications) mais sa réalisation et son contrôle est complexe.

Un système de gestion hiérarchique combine les avantages des deux modèles précédents, et utilise des stations gérantes intermédiaires pour distribuer les tâches de gestion. Chaque station gérante intermédiaire, appelé chef de groupe (*cluster-head*), gère un domaine du réseau appelé groupe (ou cluster) ; elle collecte et traite les informations des nœuds de son domaine et transmet les informations à la station gérante du niveau supérieur si nécessaire. Elle transmet aussi les messages de la station gérante du niveau supérieur aux nœuds de son domaine. Il n'y a pas de communication directe entres les stations gérantes intermédiaires.

3.1. Architectures centralisées

Les deux architectures qui se sont développées en premier il y a une vingtaine d'années, basée sur la technologie classique client/serveur, proviennent du modèle Internet, avec le protocole SNMP (*Simple Network Management Protocol*), et du modèle de référence OSI normalisé par l'ISO, avec le protocole CMIP (*Common Management Information Protocol*). Le modèle ISO est aujourd'hui en chute libre, mais il reste un modèle au même titre que le modèle de référence. Au contraire, le protocole SNMP a pris totalement le devant de la scène en se présentant sous diverses formes dont nous allons présenter quelques uns dans ce chapitre.

Sur le point de l'administration, un système de réseau informatique se compose d'un ensemble d'objets qu'un système d'administration surveille et contrôle (figure 5) [77]. Chaque objet est géré localement par un processus appelé agent, qui transmet régulièrement ou sur sollicitation les informations de gestion relatives à son état et aux événements qui le concernent au système d'administration. Le système d'administration comprend un processus appelé gérant (*manager*), qui peut accéder aux informations de gestion d'une base d'informations locale appelée MIB (*Management Information Base*) via un protocole d'administration comme SNMP ou CMIP, ce qui le met en relation avec les divers agents. La MIB contient l'ensemble des informations décrivant les objets gérés, en définit le nommage, et en précise le type, le format et les actions. Ces données administratives se présentent sous la forme de compteurs, de seuils, de répertoires de noms et d'adresses, etc.

Le principe repose donc sur les échanges :
- D'une part, entre la base d'informations MIB et l'ensemble des éléments administrés (objets) ;
- D'autre part, entre les éléments administrés et le système d'administration.

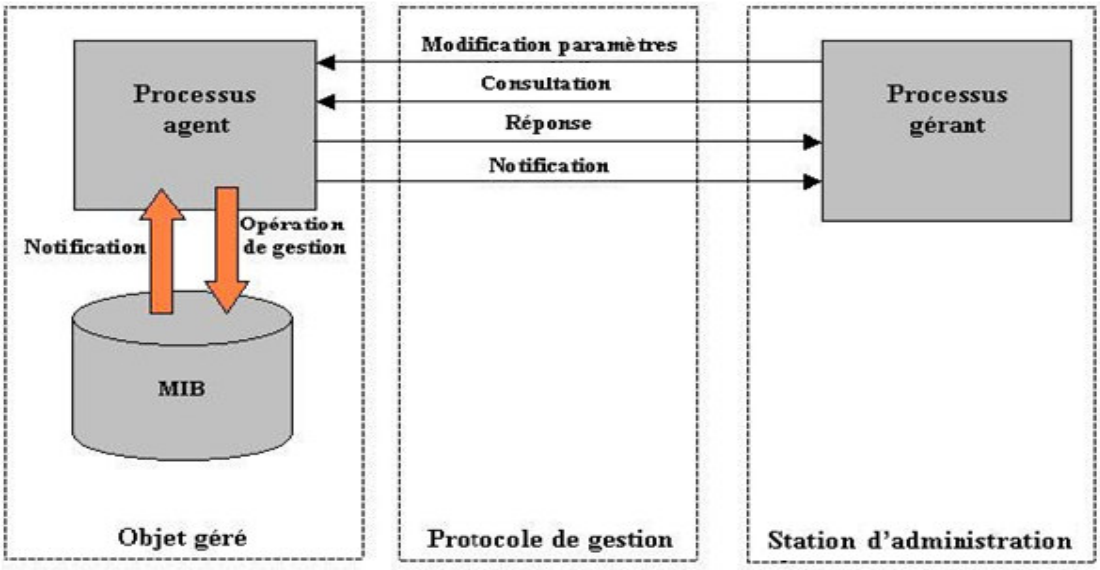


Figure 5 : *Structure fonctionnelle d'un système d'administration.*

3.2. Architectures distribuées et hiérarchiques à base d'agents mobiles

Depuis quelques années, la communauté de gestion et d'administration de systèmes et de réseaux a reconnu le fort potentiel qu'offrent les systèmes basés sur des agents et en particulier, sur des agents mobiles, pour effectuer des tâches d'administration [91] : décentralisation du contrôle, répartition des tâches de calculs liées à l'administration, autonomie, possibilité de gain en bande passante, etc. Les limites de l'architecture centralisée pour certaines catégories de réseaux, présentées ci-dessus, ont favorisé l'émergence des architectures distribués et hiérarchiques. Nous présentons ci-dessous celles qui sont basées le concept d'agent mobile pour montrer l'impact de l'utilisation de cette technologie dans la gestion des réseaux informatiques.

Les premiers travaux qui ouvraient la voie des plates-formes à base d'agents mobiles dans le cadre de l'administration système et réseau, sont les travaux d'administration par délégation. L'approche appelée MbD (*Management by Delegation*) [42] introduite au début des années 1990, est la première à présenter les concepts de la délégation des tâches d'administration de la station centralisée vers des éléments intermédiaires, qui a leur tour deviennent des stations d'administration agissant comme des proxies. L'architecture de MbD, repose sur des possibilités de déporter les fonctions d'administration de la station centralisée vers ces proxy, appelés *MAD agent*. Le composant central de l'architecture proposée est le MAD agent (voir figure 6). C'est une combinaison entre le modèle hiérarchique de supervision et d'un agent *proxy* (le MAD agent) qui fournit un ensemble de services nécessaire pour effectuer la supervision et le contrôle des éléments gérés. Cet ensemble de services peuvent être invoqués depuis la station d'administration ou depuis un autre MAD agent.

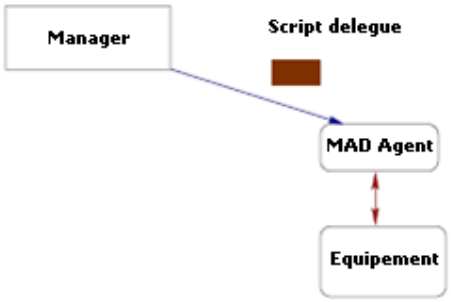


Figure 6 : *Architecture de MbD.*

Le processus de délégation est engendré par la station d'administration sur laquelle ont été décrites les fonctions d'administration qui seront transmises à un MAD Agent. Dans cette architecture distribuée, un MAD agent peut devenir à son tour le gestionnaire d'autres MAD agents, et la station maître d'administration devenir un agent proxy. L'approche MbD offre ainsi une gestion décentralisée et asynchrone des réseaux mais ignore l'intégration dans sa plateforme des systèmes basés sur le protocole SNMP, i.e. qu'elle n'est pas compatible avec SNMP.

Par la suite, un certain nombre de plates-formes à base d'agents mobiles pour l'administration systèmes et réseaux ont vu le jour. En effet, la technologie d'agent mobile apporte un degré de distribution supplémentaire en renforçant l'implication des éléments du réseau dans le système d'administration.

Un très grand nombre des travaux de recherches réalisés et utilisant le concept d'agent mobile sont bâties au dessus d'une architecture en Java. La popularité du langage Java et en particulier, le fait qu'il masque l'hétérogénéité des systèmes, qu'il permet aisément la mobilité du code et qu'il propose des outils permettant d'appréhender les problèmes relevant de la sécurité, a plus précisément focalisé l'intérêt de cette communauté sur des systèmes à agents mobiles écrits en Java. Mais il existe d'autres types de plates-formes à agents mobiles tournées vers l'administration système et réseau où les tâches à réaliser sont définies par d'autres langages, comme Tcl/Tk, qui permettent d'effectuer des opérations d'administration par l'utilisation de scripts, par exemple AgentTcl qui utilisent l'interpréteur de scripts pour distribuer le code dans les nœuds d'accueil et lancer des opérations d'administration à distance. Toutefois, un désavantage quant à l'utilisation des scripts repose sur la nécessité de l'interprétation du script, ce qui induit un temps d'exécution plus long, plutôt que d'utiliser un langage compilé comme l'est Java.

Les premiers travaux de recherche, utilisant les agents mobiles pour la gestion des réseaux, ont présentés des systèmes de gestion qui sont relativement simples. On peut citer les deux systèmes simples suivants:

– NMMA (*Network Management using Mobile Agent*) [124] est un système de gestion qui reprend l'architecture traditionnelle, ou le protocole SNMP est tout simplement remplacé par l'agent mobile pour la communication entre la station de gestion et les éléments gérés du réseau. L'objectif de l'approche NMMA est d'optimiser la consommation de la bande passante.

– IMA (*Intelligent Mobile Agent*) [52] est une infrastructure à base d'agent mobile pour une gestion distribuée de réseaux. Elle comprend une architecture d'agent mobile, un itinéraire de migration, un plan d'actions (figure 7), une gestion par délégation, intelligence et sécurité. Le fonctionnement général de l'agent mobile est montré dans l'organigramme de la figure 7.

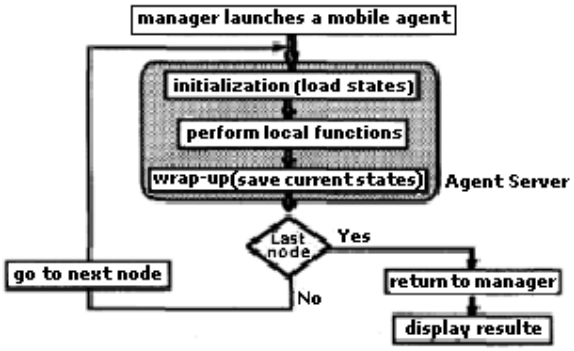


Figure 7 : *Fonctionnement général d'un agent mobile.*

Parmi les travaux les plus importants cités dans la littérature et qui sont assez complexes, on peut citer :

– MIAMI (*Mobile Intelligent Agents for Managing the information Infrastructure*) qui est un projet européen basé au dessus de Grasshopper, est la première plate-forme à agents mobiles compatible avec les standards industriels supportant des agents mobiles et leur gestion. MIAMI regroupe plusieurs entités différentes, comme des entreprises, sous la forme d'une place appelée P-OVPN (*Place-Oriented Virtual Private Network*), afin de pouvoir administrer des nœuds situés à différents endroits en appliquant, par exemple, la même politique de sécurité à tous ces nœuds. Cette approche permet de déléguer la gestion de cette P-OVPN à une entité extérieure à des fins d'administration [91].

– MAGENTA (*Mobile AGENT for Administration*) [96] est un environnement d'exécution d'agents mobiles mis en œuvre par l'IRISA au campus universitaire de Beaulieu en France, pour la réalisation d'un gestionnaire mobile de réseaux, appelé Astrolog, dans lequel les applications d'administration sont distribuées et permet à un administrateur de réseaux d'exécuter ses tâches à partir d'un poste de travail quelconque relié au réseau administré par un réseau quelconque. Le poste de travail peut être aussi bien un ordinateur de poche ou un ordinateur portable utilisant un réseau sans fil qu'une station de travail d'un réseau local. Astrolog est fondé sur la technologie d'agent mobile pour certaines fonctions d'administration, qui permet son fonctionnement en mode déconnecté et sur un réseau à faible débit ou coûteux. En effet, si l'administrateur utilise un ordinateur portable, il se peut qu'il soit déconnecté ou que le lien (SLIP/PPP) qui le relie au réseau soit très lent. Ainsi, lorsque les agents mobiles doivent revenir à la station d'administration, ils vérifient qu'ils ont la possibilité d'y revenir. Dans le cas contraire, l'agent mobile attend la reconnexion de la station avant d'y revenir. Pendant cette période d'attente, il stationnera sur un nœud de la plate-forme en attendant cet événement. L'environnement Magenta est portable et fonctionne dans un environnement

d'informatique nomade et intègre des mécanismes de tolérance aux fautes. Le système d'administration Astrolog et son environnement d'agents mobiles Magenta ont été intégralement mis en œuvre dans le langage Java.

– MobileSpaces [98,100], réalisée à l'institut national d'informatique au Japon, est une plate-forme qui fournit des agents mobiles transporteurs d'agents mobiles. Ces agents mobiles transporteurs assurent les opérations de migration des agents qu'ils transportent dans un sous-réseau donné. Un ensemble d'itinéraires adaptés pour le sous-réseau est défini et mis à la disposition des agents mobiles transporteurs. La politique de migration, dans un sous-réseau donné, est dépendante des fonctions d'administration que doivent exécuter les agents mobiles. Ainsi, l'agent transporteur aura l'itinéraire associé à la fonction d'administration. Une extension possible de ces agents transporteurs est de les utiliser pour faire traverser un pare-feu aux agents mobiles d'administration, diminuant ainsi les contraintes de sécurité qui sont mises en œuvre.

Pour permettre la compatibilité avec les applications de gestion classiques, d'autres travaux ont intégré la technologie d'agent mobile à l'architecture classique, sous le protocole SNMP/CMIP, et qui sont les suivants :

– MNMA (*Mobile Network Management Agent*) [67] est une architecture de gestion implémentée sous l'environnement CORBA. Elle utilise des agents mobiles pour l'exécution de fonction spécifiques de gestion de manière distribuée, et communique de façon transparente, via un serveur proxy, avec les objets des protocoles SNMP/CMIP (figure 8).

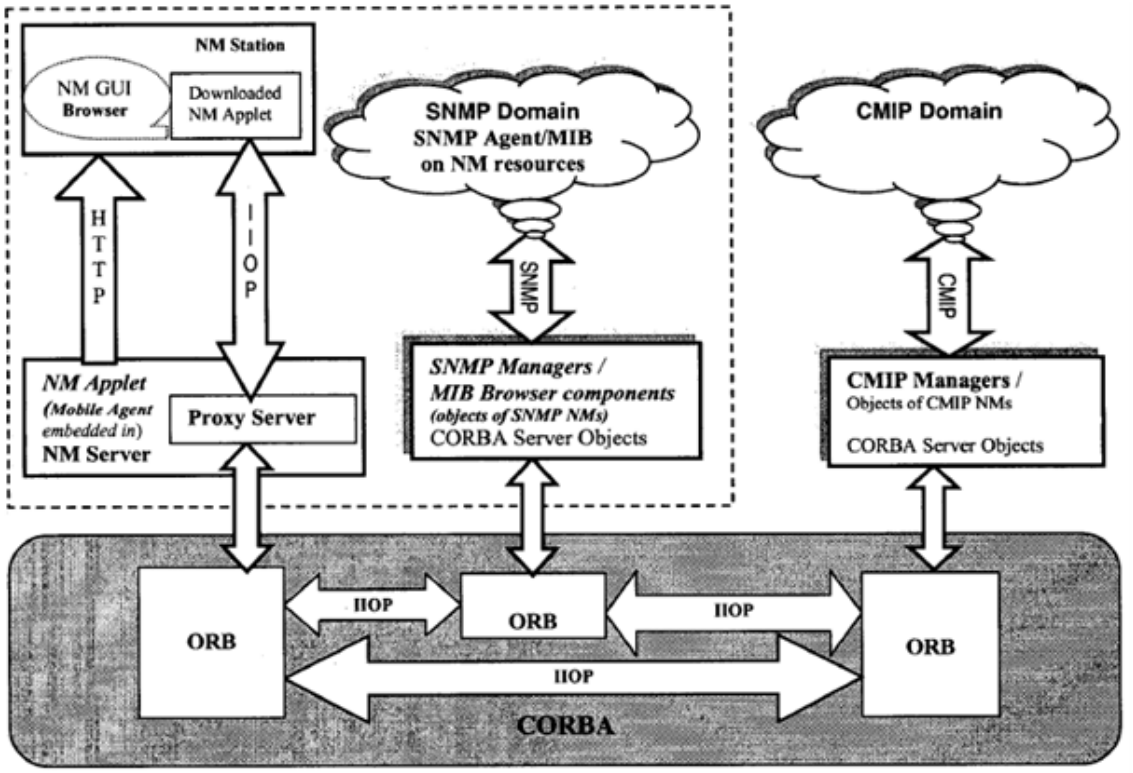


Figure 8 : *Architecture dynamique de MNMA.*

– JAMES [109] est une plate-forme à base d'agents mobiles, réalisée conjointement par l'université de Coimbra en Portugal et Siemens, pour la gestion des réseaux de télécommunications et des réseaux informatiques. Ce travail introduit la possibilité de superviser la plate-forme d'administration elle-même, c'est-à-dire James, en utilisant le protocole SNMP et l'instrumentation JMX (*Java Management Extensions*) pour tous les objets de la plate-forme. Pour cela la MIB-JAMES a été définie en intégrant le système de gestion classique sous SNMP dans la plate-forme JAMES (figure 9). Grâce à cette MIB, la plate-forme à agents mobiles peut aussi être supervisée par d'autres plateformes d'administration réseau classique (par exemple HP Network Node Manager).

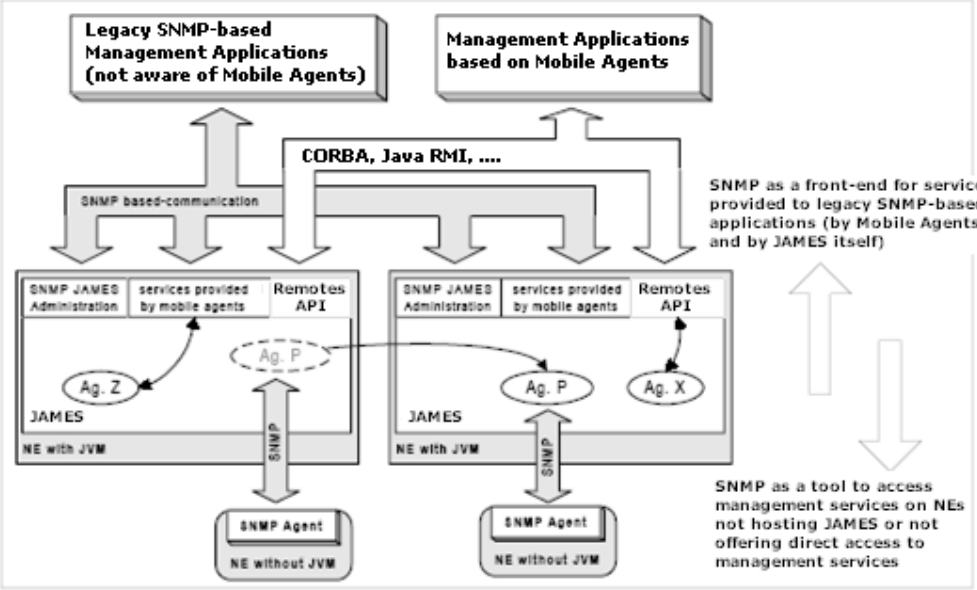


Figure 9 : *Architecture intégrée de JAMES.*

– Très récemment, Guo et al. [44] ont présenté un autre système hybride qui combine les agents mobiles avec SNMP et qui est implémenté sous la plateforme Grasshopper. Dans ce système (figure 10), un générateur d'agents mobiles crée des agents mobiles à la demande de l'application de gestion du réseau.

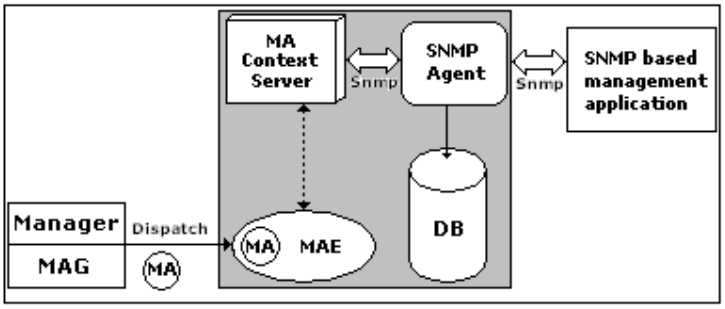


Figure 10 : *Système de gestion hybride combinant les agents mobiles et SNMP.*

– Hu et al. [53] ont présenté récemment aussi, un système de gestion de réseaux basé sur une combinaison de la technologie des agents mobiles avec la technologie Web. Pour gérer les bases de données distribuées, les agents mobiles collectent et mettent à jour les informations de la MIB en communiquant avec les agents statiques de l'architecture traditionnelle sous les protocoles SNMP/CMIP (figure 11).

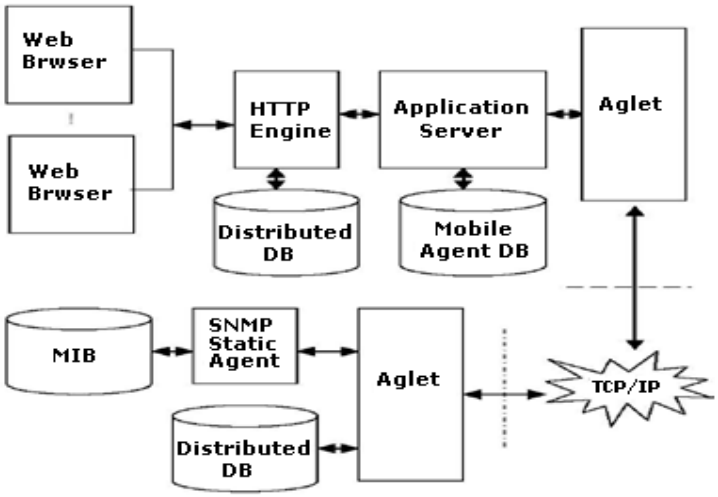


Figure 11 : *Système de gestion basé sur les agents mobiles et la technologie Web.*

Plusieurs autres travaux qui proposent des architectures hiérarchiques et basées sur les agents mobiles, divisent le réseau en plusieurs domaines de gestion (collection de nœuds regroupés sous la même autorité d'administration), permettant ainsi une gestion distribuée. On peut citer :

– Rubinstein et al. [93,94,95] ont proposé une architecture hiérarchique de gestion à deux niveaux. Basée sur l'agent mobile, elle utilise plusieurs stations gérantes (*Domain Manager*) et utilise aussi le concept de station gérante de stations gérantes (*Manager of Managers*).

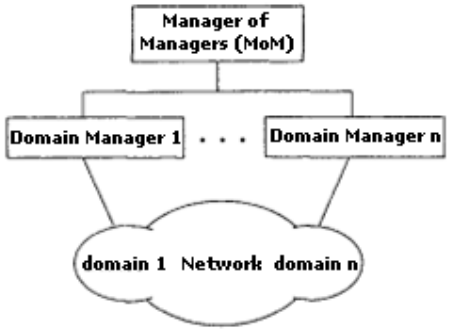


Figure 12 : *Architecture d'une gestion de réseau.*

Dans cette architecture, chaque station gérante gère un domaine du réseau (figure12). La station gérante principale est responsable de la coordination des stations gérantes des domaines et à qui elle leurs délègue des tâches de gestion. L'agent mobile est utilisé par la

station gérante pour gérer les éléments de son domaine. Afin d'offrir un mécanisme de tolérance aux pannes, une station gérante de secours est préconfigurée dans chaque domaine de gestion.

– Dans [33,125,127], les auteurs proposent des structures hiérarchiques, presque similaires, avec plusieurs domaines de gestion SNMP, chacun géré par un agent serveur (figure 13) et sur lequel est installé un environnement d'exécution d'agents mobiles qui reçoit les agents mobiles, envoyés par la station centrale de gestion du réseau, devant effectuer les opérations de gestion sur les éléments du réseau.

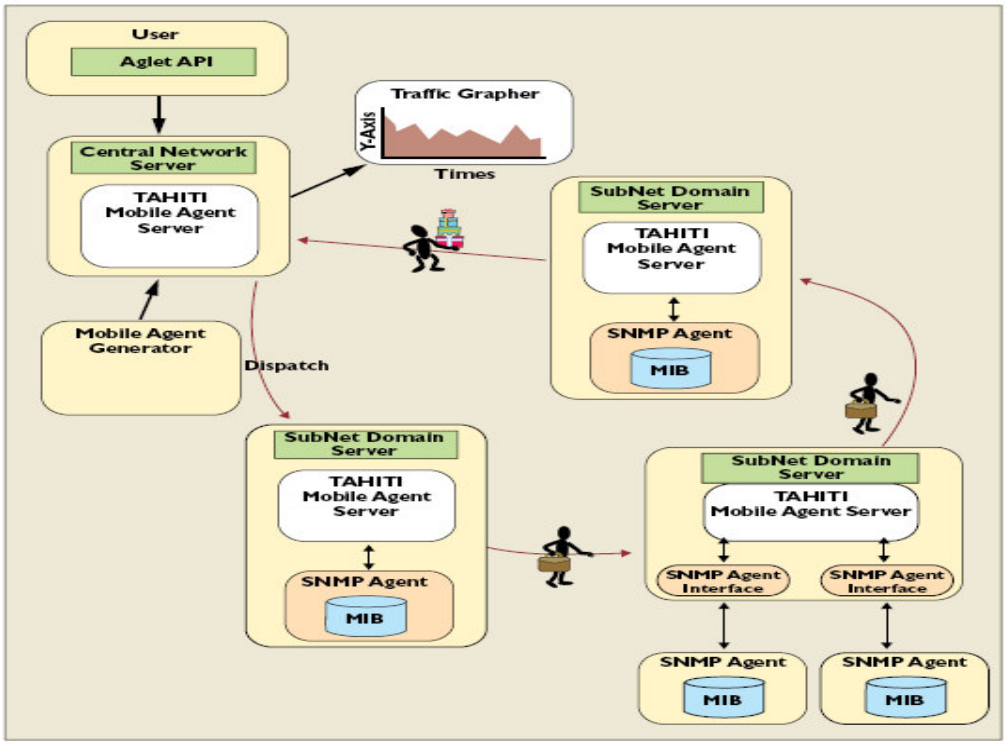


Figure 13: *Architecture distribuée de gestion de réseau à base d'agent mobile.*

D'autres travaux plus spécifiques, sur la gestion de panne dans les réseaux, ont été réalisés en se basant sur les agents mobiles. Chou et Kao [26] présentent un système qui utilise six types d'agents mobiles et qui ont pour tâches, entre autres, la découverte, la localisation et l'identification de pannes dans un réseau. El-Darieby et Bieszcza [35] ont réalisé un système de gestion de panne avec un modèle hiérarchique de réseau et à base d'agent mobile, s'exécutant sous l'environnement MCT (*Mobile Code Toolkit*) développé à l'université Carleton au Canada. L'agent mobile visite tout les nœuds possédant une quelconque information appropriée à la tâche assignée, traite les informations localement, établit des diagnostics, entreprend les actions appropriées et retourne les résultats obtenus à la station d'administration en quittant les nœuds du réseau.

Des travaux de recherche qui traitent du problème de la sécurité ont été aussi présentés dans la littérature, vu que l'inconvénient majeur de l'utilisation des agents mobiles est la

sécurité (voir paragraphe 3.9 du chapitre 3). En effet, un système de gestion doit réaliser les fonctions de gestion de manière sécurisée. Il doit garantir la confidentialité et l'intégrité des informations de gestion traversant le réseau, il doit aussi vérifier l'identité et les droits d'accès des agents (mobiles) impliqués dans la gestion du réseau à un niveau de sécurité acceptable. Ci-dessous quelques travaux sur la sécurité :

– MAP (*Mobile Agent Platform*) [86] est une plateforme à agents mobiles pour l'administration système et réseau, développée à l'université de Catania en Italie, qui sécurise en deux niveaux l'accès des agents mobiles sur les nœuds. Dans un premier temps, lorsque les agents mobiles entrent dans un nouveau domaine, ils obtiennent des droits d'accès sur les nœuds qui composent le domaine. De plus, une restriction peut être appliquée sur chaque nœud du domaine en fonction des exigences de sécurité mises en place par l'administrateur.

– Koliousis et Sventek [60] présentent une infrastructure pour la gestion de réseau qui est constituée de trois éléments, un système d'agent mobile (avec Ajanta comme plateforme), un agent SNMP et une interface pour une intégration sécurisée des agents mobiles avec SNMP (figure 14).

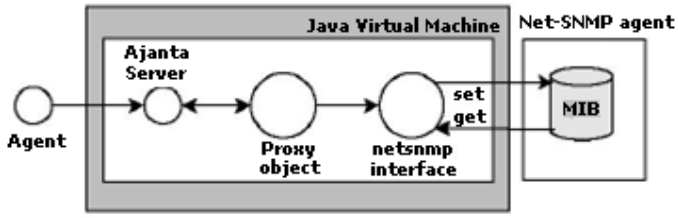


Figure 14 : *Infrastructure de gestion de réseau.*

– Ibrahim [54] propose une gestion hiérarchique du réseau organisé en domaine, fournissant un mécanisme de sécurité qui crypte le code et les données de l'agent mobile avant son lancement dans le réseau. Ceci permet aux seuls sites d'accueil possédant la clé de décryptage d'être capable d'exécuter l'agent.

D'autres travaux ont été réalisés dans ce sens, tels que SOMA (*Secure and Open Mobile Agents*) [80], TINMAN [3], etc. Cependant, le problème non encore complètement résolu dans ces travaux est la sécurisation des agents mobiles vis-à-vis des sites d'accueil mal intentionnés.

Pour terminer cette partie sur l'utilisation du concept d'agents mobiles dans la gestion des réseaux filaires, notons aussi leurs utilisations dans les protocoles de routage d'information. N. Minar et al. [74] présentent un système qui utilise deux types d'agents mobiles, l'agent « *routing* » et l'agent « *messenger* ». Les agents « *routing* » explorent le réseau en mettant à jour les tables de routage à partir de leur propre connaissance. Pour délivrer son message à une destination donnée, l'agent « *messenger* » consulte ces tables de routages, sélectionne le prochain nœud voisin par lequel est passé le dernier agent « *routing* » venant de cette destination, et ainsi de suite jusqu'à atteindre le nœud destinataire. Di Caro et Dorigo [30] se sont basés sur le système de colonies de fourmis (de très petits agents mobiles en grand nombre qui se propagent dans le réseau) pour proposer un algorithme de routage appelé

AntNet. Dans cet algorithme, chaque fourmi artificielle (agent mobile) construit un chemin depuis un nœud source vers une destination donnée, tout en collectant des informations explicites sur la durée de parcours du chemin et sur l'état de charge du réseau. Ces informations qui sont retournées par d'autres fourmis se propageant dans le sens opposé, permettent de mettre à jour les tables de routage des nœuds visités.

4. Gestion dans les réseaux mobiles Ad hoc

4.1. Problématique

Comparé avec l'environnement statique, l'environnement mobile (constitué des réseaux mobiles) permet aux unités de calcul, une libre mobilité et ne pose donc aucune restriction sur la localisation des usagers. Les environnements mobiles qui offrent une grande flexibilité d'emploi, permettent la mise en réseau des sites dont le câblage serait onéreux à réaliser dans leur totalité, voir même impossible (par exemple en présence d'une composante mobile).

Les réseaux mobiles Ad hoc permettent un environnement de communication plus flexible et favorisent davantage la mobilité. Cette classe de réseaux essaie d'étendre les notions de la mobilité à toutes les composantes de l'environnement ; toutes les unités des réseaux de cette classe se déplacent librement et aucune infrastructure fixe ou administration centralisée n'est disponible. De tels réseaux ont pour particularité de s'auto-construire, s'auto-organiser et s'auto-administrer, il s'agit de réseaux autonomes et très dynamiques dans lesquels les systèmes de gestion des réseaux filaires ne sont plus adaptés. Ces nouveaux réseaux doivent donc disposer de leurs propres protocoles de routage et **de gestion**. En effet, la gestion des réseaux mobiles ad hoc est une tâche extrêmement difficile, comparé aux réseaux filaires et aux réseaux avec infrastructure fixe et ceci pour plusieurs raisons:

- La topologie des réseaux ad hoc change rapidement, dû à la mobilité des nœuds.
- La bande passante disponible est limitée, vu la nature des médiums de communication utilisant des liens sans fils de type radio et l'étroitesse de la plage de fréquence allouée par les autorités de régulation des télécommunications pour ces réseaux.
- Une qualité variable de l'état de la liaison, dû à la variation de la distance entre deux nœuds sans fil qui constitue un vecteur non trivial d'atténuation et de distorsion du signal, et du fait que le canal de transmission est susceptible d'être victime d'interférences, bruits ou parasites.
- Des déconnexions fréquentes et imprévisibles (dû aussi à la mobilité des nœuds et à la nature des communications sans fils).
- En raison de leur alimentation par batterie, les nœuds mobiles ont une autonomie énergétique limitée.
- Des ressources modestes et plus particulièrement une capacité de stockage limitée.
- Des partitionnements fréquents du réseau, en raison de la nature dynamique des réseaux ad hoc (nomadisme des nœuds, batterie limitée, etc.).
- Une sécurité limitée.

Étant une étape importante dans la gestion des réseaux, la collecte d'informations est un vrai problème dans les réseaux ad hoc. En effet, la station de gestion du réseau a besoin de collecter périodiquement, les informations sur l'état du réseau (tels que l'état de la batterie, la qualité de la liaison, position des nœuds, vitesse et direction, etc.), à partir des nœuds dans un réseau ou la bande passante et la source d'énergie des nœuds sont limitées. Le système de gestion doit assurer de ne pas consommer une quantité significative de ces ressources. Il doit aussi s'adapter aux changements constants de la topologie et être assez robuste aux défaillances des liaisons et des nœuds, aux changements de la qualité de la liaison et aux partitionnements du réseau. Il doit avoir l'aptitude de réagir rapidement à tous ces changements. Il nécessite de fonctionner dans des environnements hétérogènes, allant de simples capteurs à des ordinateurs portables. Finalement, pour le besoin de l'interopérabilité, le protocole de gestion pour les réseaux ad hoc doit être compatible avec les protocoles de gestion actuellement utilisés, tel que le protocole SNMP, malgré leurs insuffisances pour les réseaux sans fils.

En plus des fonctionnalités de gestion pour un réseau filaire, le réseau mobile Ad hoc nécessite donc d'autres fonctionnalités supplémentaires de gestion, tels que :

- La gestion de la topologie, utilisée par beaucoup de protocoles de routage pour améliorer leurs performances. Elle consiste à organiser hiérarchiquement le réseau, le plus souvent en clusters, et à assurer la maintenance de la topologie. (paragraphe 4.3 et 4.4).
- La gestion de la bande passante, qui permet de garantir sa disponibilité en définissant des politiques d'accès aux canaux de communication (techniques de réservation, d'allocation et de partage).
- La gestion de l'énergie, qui a pour objectif la sauvegarde et l'économie de l'énergie.
- La gestion de la mobilité, qui a comme objectif principal le maintient des informations sur la position des nœuds mobiles et la gestion de leurs connexions lorsqu'ils se déplacent. Elle est censée permettre l'exécution des applications dans le réseau ad hoc de manière transparente à la mobilité des nœuds qui induit des déconnections, des partitionnements, etc. La gestion de la mobilité pour les réseaux mobiles (cellulaires et ad hoc), inclut deux procédures [6] :
  - **La gestion de la localisation des objets mobiles** (chapitre 4) qui permet de fournir au réseau des informations sur la position courante d'un nœud mobile ainsi que la mise à jour de ces informations.
  - La gestion des handovers, pour le compte des réseaux cellulaires. Le handover est le processus par lequel une communication établie est maintenue alors que le nœud mobile se déplace à travers le réseau cellulaire; elle implique que la communication puisse passer d'un canal physique à un autre avec une coupure sans conséquences.

Nous présentons ci-dessous quelques travaux sur la gestion de réseaux ad hoc, parmi lesquels, certains proposent l'utilisation de la technologie d'agents mobiles. Ensuite, des protocoles de routage d'informations plus spécifiques aux réseaux ad hoc, et qui sont basés sur le concept d'agents mobiles, seront présentés. Enfin, nous décrirons le principe de clustering utilisé dans la gestion de topologie des réseaux ad hoc.

4.2. Systèmes de gestion pour les réseaux ad hoc

Plusieurs travaux de recherches ont été effectués pour concevoir et réaliser des architectures de gestion pour les réseaux mobiles ad hoc. Les systèmes proposés cherchent à satisfaire un minimum de contraintes et de besoins nécessaires à la gestion des réseaux ad hoc, tels que l'adaptabilité, l'autonomie, l'économie, l'hétérogénéité, l'évolutivité et la survie. Les travaux les plus cités dans la littérature sont AMNP [24], PECAN [23], Guerilla [107], DNA (*Distributed Network Agents*) [78] et PBNM (*Policy-based Network Management)* [78]. La plupart des approches proposées se basent sur l'organisation du réseau en groupe de nœuds ou clusters (paragraphe 4.4) pour assurer de bonnes performances de gestion. C'est ainsi que la collecte de l'état du réseau englobe trois opérations : la création d'une gestion hiérarchique à travers le clustering, la propagation de l'information entre les clusters et la propagation de l'information à l'intérieur des clusters. Nous citons ci-dessous quelques uns de ces travaux :

– Le protocole ANMP (*Ad hoc Network Management Protocol*) est le plus référencé dans les travaux sur la gestion des réseaux ad hoc. Il est basé sur des agents non mobiles et utilise une structure hiérarchique de clusters à trois niveaux (figure 15). Le niveau inférieur consiste aux nœuds du réseau groupés en cluster ou groupe, et qui sont gérés par le nœud chef de groupe (*cluster head*). Les chefs de groupe, qui forment le deuxième niveau, sont gérés par le nœud gérant (*manager*) principal, qui à son tour représente le niveau supérieur.

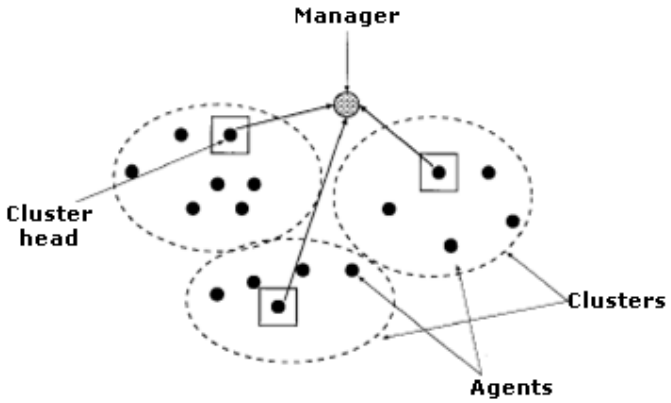


Figure 15 : *L'architecture hiérarchique d'ANMP.*

– PECAN (*Policy-Enabled Configuration Across Networks*) est un système de gestion de réseaux mobiles ad hoc développé dans le cadre du programme DRAMA (*Dynamic Re-Addressing and Management for the Army*) [25] de l'armée américaine. Le système est à base de règles (politiques) prédéterminées qui spécifient, à un haut niveau, les besoins et le comportement désiré d'un réseau. Ceci pour permettre des reconfigurations automatiques et une gestion flexible. Ces règles de gestion sont exprimées en termes d'actions qui peuvent être exécutés par des agents qui forment une structure hiérarchique. Ci-dessus deux exemples de règles pour améliorer les performances de gestion :

- Si le temps de réponse du serveur (un nœud gérant) dépasse les cinq secondes, alors prendre la décision s'il faut repositionner le serveur (le choix du nœud à agir comme un serveur sera

en fonction de ces capacités tels que la puissance de calcul, la capacité de la batterie, la puissance du signal).

- Selon la densité connue et la connectivité des nœuds dans le réseau, basculer entre le routage proactif et le routage réactif.

– SOLOMON [63] fournit une méthode efficace dans la collecte de données en utilisant une couche intermédiaire de gestion (figure 16), qui permet de réduire la charge dans le réseau en produisant un résumé lors de la collecte d'informations, avant de le transmettre vers le niveau supérieur dans la hiérarchie.

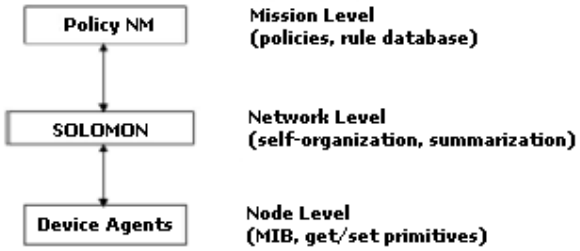


Figure 16 : *Architecture de gestion pour les réseaux ad hoc.*

En effet, SOLOMON gère ceci en organisant le réseau en clusters. La collecte des informations par un chef de groupe, à partir de chaque nœud d'un cluster, se fait périodiquement. Le résumé de ces informations collectées n'est diffusé par le chef de groupe qu'à un sous ensemble des autres chefs de groupe du réseau. Le choix de ce sous ensemble de voisins est fait avec le protocole Gossip [63].

– L'architecture Guerilla, basée sur le code mobile, est conçue selon un modèle gérant/agence (*manager/agency*) (figure 17) qui supporte un groupement dynamique de clusters (*agency*) et une gestion collaborative entre les chefs de clusters via le paradigme *Peer-to-Peer*. Une agence peut former récursivement d'autres sous-agences pour les grandes tailles de réseau.

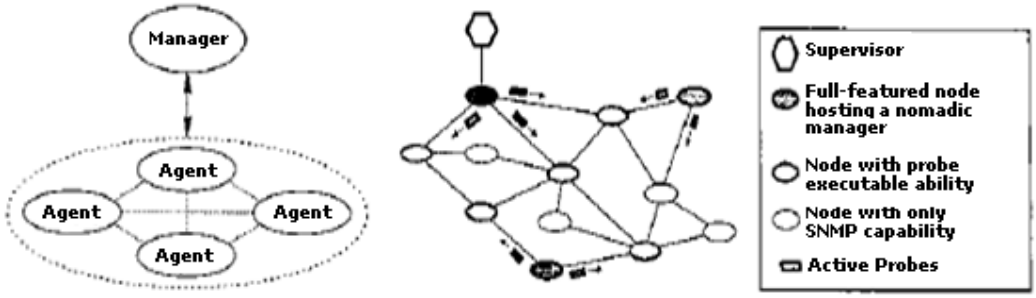


Figure 17: *Architecture de Guerilla.*

Les nœuds d'un réseau ad hoc, qui peuvent être hétérogènes (allant d'un simple capteur à un ordinateur portable), ne contribuent pas de la même façon à la gestion du réseau dans Guerilla. Ceux avec de grandes capacités exécuteront les tâches de gestion en tant que nœuds gérants (*nomadic managers*), les autres seront des nœuds gérés. Une entité logicielle

adaptative (code mobile) est alors installée sur chaque nœud gérant qui aura pour rôle la gestion des autres nœuds. Pour permettre l'extensibilité du réseau, les nœuds forment de manière adaptative des groupes avec au moins un nœud gérant. Les nœuds gérant collaborent de façon autonome pour gérer la totalité du réseau ad hoc avec un minimum d'aides par des entités externes et par le superviseur. Ils coopèrent aussi pour faciliter le processus du handoff pour les nœuds qui quittent un groupe et joignent un autre. Un nœud gérant peut changer de rôle suivant le changement de topologie, le niveau de son énergie, la densité des nœuds ou autres considérations. L'entité logicielle de gestion peut alors décider de migrer vers un autre nœud capable de prendre le relais. Des nœuds sont préconfigurés aussi pour assurer les fonctions de gestion dans le cas de partitionnement du réseau. Lorsque deux partitions se reconnectent, les entités logicielles des nœuds gérants correspondantes décident de fusionner. Les nœuds gérants peuvent utiliser aussi le code mobile pour la collecte de données à partir des nœuds gérés et l'échange des informations de gestion avec les autres nœuds gérants.

La technologie d'agent mobile a été utilisée aussi dans les systèmes de sécurité des réseaux mobiles ad hoc. Les travaux de recherche sur la sécurité peuvent être classés en trois catégories : la gestion de la clé, la sécurité du routage d'information et la détection d'intrusion.

– L. Zhu et al. [128] décrivent dans leur article une méthode de gestion distribuée de la clé qui introduit l'agent mobile. Des agents mobiles naviguent dans le réseau pour l'échange de clés privés et d'informations de topologie avec les nœuds visités et avec d'autres agents.

– ARMA (*secure distributed Anonymous Routing protocol based on Mobile Agent*) est un protocole proposés par Boukerche et Ren [17], basé sur le concept d'agent mobile, qui assure une communication sécurisée et anonyme dans un réseau ad hoc. Dans leur solution, l'agent mobile appelé AEA (*Autonomous Encryption Agent*) est utilisé pour protéger l'intimité de l'émetteur et du récepteur tout en assurant également la protection pour le contenu du message pendant qu'il traverse le réseau. L'AEA a principalement deux fonctions ; juger la destination prévue et produire différentes clefs pour chiffrer l'identité des nœuds intermédiaires et certaines informations. L'agent est généré par le nœud source, et seulement la source et les nœuds destinataires sont autorisés à contrôler cet agent après authentification. C'est-à-dire, les nœuds intermédiaires ne peuvent chiffrer et déchiffrer aucune information à travers cet agent.

– Kachirski et Guha [56] proposent un système distribué de détection d'intrusion (IDS modulaire) pour les réseaux ad hoc, basé sur le concept d'agent mobile. Un agent est un petit objet actif et intelligent qui traverse le réseau pour s'exécuter sur certains nœuds. Les agents sont légers, mis-à-jour dynamiquement et possèdent des fonctionnalités liées à la détection d'intrusion.

– Y. Ping et al. [83] ont proposé une architecture assez complexe pour la sécurité dans les réseaux ad hoc, basée sur un système multi-agent mobile, et qui s'inspire des systèmes humains immunisés ou le réseau informatique est assimilé au corps humain et les intrus aux microbes pathogènes. Les agents mobiles correspondent alors aux "lymphocytcs″ qui détectent et isolent les intrus. L'architecture qui est conçue pour être distribuée, autonome,

adaptable et extensible, est composée de trois types d'agents, l'agent "monitor", l'agent "*decision*" et l'agent " *killer*" (figure 18). L'agent "monitor" qui réside sur chaque nœud, gère les nœuds voisins en collectant tout les paquets dans sa portée de communication. Il code les informations sur le comportement de ses voisins et les remets à l'agent "*decision*". Celui-ci peut établir un jugement à l'aide des règles de sécurité et de sa mémoire immunisée. Si l'agent "*decision*" juge alors qu'un certain nœud est un intrus, il produit un agent "*killer*" qui est responsable pour éliminer l'intrus. L'agent "*killer*" migre vers le nœud voisin ou se trouve l'intrus pour le cerner et l'isoler. Il peut aussi l'isoler en interceptant les requêtes de routage de l'intrus.

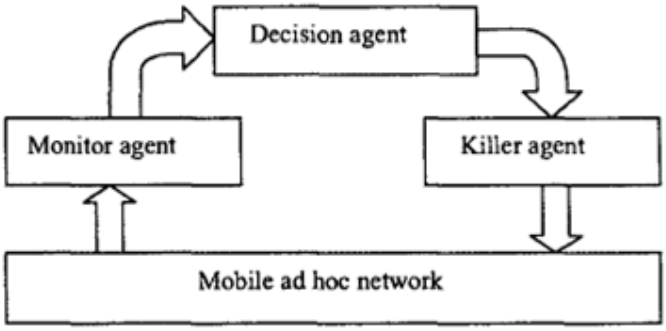


Figure 18: *Architecture d'un agent système.*

Comme pour la collecte de données dans la gestion de réseaux, le réseau ad hoc est structuré en clusters afin de mieux maitriser la sécurité dans ces réseaux. Cependant, le problème de ces systèmes de sécurité à base d'agents mobiles est comment protéger ces agents mobiles qui peuvent être les premières cibles d'une attaque (paragraphe 3.9.1.1, chapitre 3).

4.3. Routage dans les réseaux mobiles ad hoc

Dans un réseau ad hoc, si deux nœuds sont suffisamment proches pour que le signal envoyé par l'un soit perçu par l'autre, alors l'émetteur peut envoyer directement un message au destinataire. Ainsi, un nœud peut directement accéder aux ressources fournies par un autre nœud, sans qu'aucun intermédiaire ne s'interpose dans cette relation directe privilégiée. Cependant, deux nœuds hors de portée d'émission l'un de l'autre, ne peuvent établir une communication. C'est pour cette raison que des protocoles de routage pour réseaux ad hoc ont été introduits. Ces protocoles permettent de faire transiter un message par des nœuds intermédiaires, et offrent ainsi une meilleure connectivité aux nœuds du réseau. Précisément, les protocoles de routage pour les réseaux ad hoc, issus du groupe de travail MANET (*Mobile Ad hoc NETwork*) de l'IETF (*Internet Engineering Task Force*), localisent les nœuds mobiles présents dans le réseau en identifiant les chemins d'accès entre nœuds distants de plusieurs sauts, i.e., la liste des nœuds intermédiaires faisant transiter les paquets d'une source vers une destination.

Suivant la méthode de création et la maintenance des routes lors de l'acheminement des informations, les protocoles de routage se décomposent en cinq familles : protocoles proactifs, réactifs, hybrides, hiérarchiques ou géographiques (figure 19) [38]:

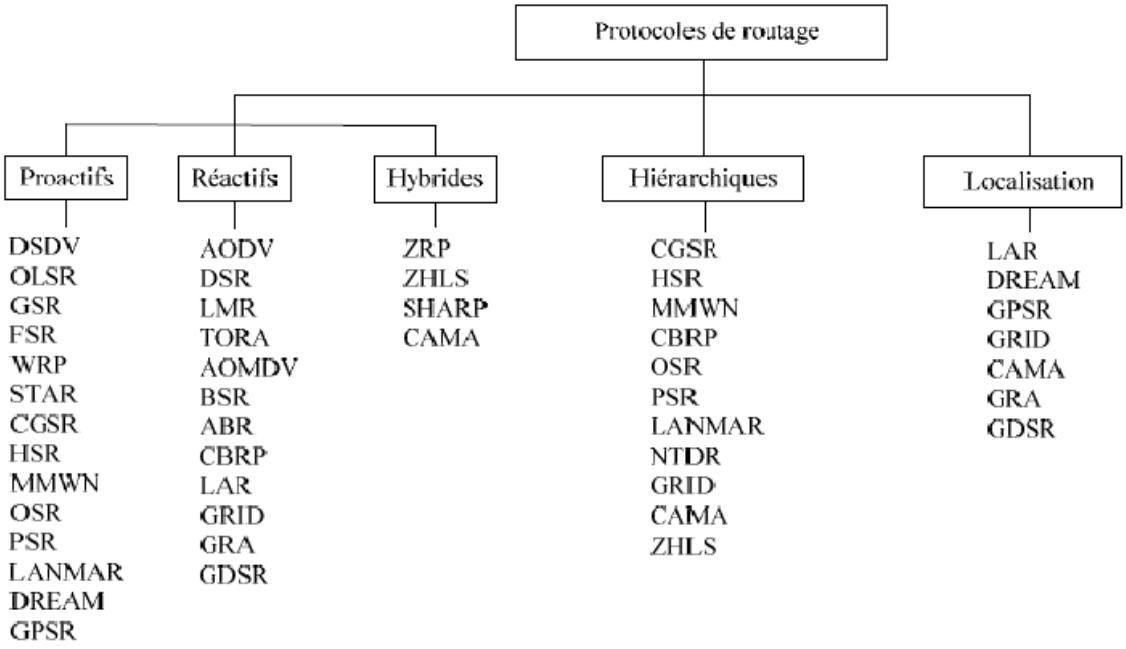


Figure 19 : *Différents type de protocoles de routage*.

- Les protocoles proactifs (protocole DSDV (*Destination-Sequenced Distance Vector*), par exemple), prennent en compte les changements de topologie du réseau ad hoc en maintenant à jour les tables de routage. Ils recherchent régulièrement des routes en échangeant, à intervalle de temps régulier, des messages de contrôle.
- A l'inverse, les protocoles réactifs, (protocole AODV (*On-Demand Distance Vector routing protocol),* par exemple), appelés aussi protocoles à la demande (*on demand*), sont caractérisés par le fait qu'une route n'est recherchée que lorsqu'un message doit être envoyé vers une destination. Ils font appel généralement à un processus de localisation de la destination qui se décompose en deux étapes: la première étape correspond à la diffusion d'une requête de **localisation** sur le réseau, et la seconde consiste à collecter des réponses reçues.
- Les protocoles hybrides, tel que ZRP (*Zone Routing Protocol*), réalisent un compromis entre les protocoles proactifs et réactifs, en combinant les deux modèles de routage. Le protocole proactif est appliqué dans un périmètre réduit autour de la source (nombre limité de voisins), tandis que le protocole réactif est appliqué au delà de ce périmètre (les voisins lointains). Cette combinaison est réalisée dans le but d'exploiter les avantages de chaque méthode et de contourner leurs limitations.
- Les protocoles de routage hiérarchiques s'adaptent mieux lorsque la taille du réseau devient de plus en plus importante. La technique de hiérarchisation sert à partitionner le réseau en sous ensembles de nœuds afin de faciliter le routage, qui se réalise à plusieurs

niveaux. Dans ce type de protocoles, la vue du réseau devient locale ; des nœuds spéciaux peuvent avoir des rôles supplémentaires. Il existe deux types de groupes de nœuds: la zone et le cluster (paragraphe 4.4). ZHLS (*Zone-based Hierarchical Link State Protocol*) et CBRP (*Cluster Based Routing Protocol*) sont deux exemples de protocoles de routage hiérarchique [14].

- Enfin, **les protocoles géographiques**, enrichissent les protocoles précédents en incluant des informations géographiques (de **localisation**) dans les tables de routage qu'ils maintiennent (paragraphe 4.3.1).

4.3.1. Protocoles de routage utilisant des informations de localisation

Echanger des informations de contrôle pour déterminer les routes consomme de la bande passante. Cette ressource se faisant rare dans les réseaux ad hoc, le protocole de routage doit tout mettre en œuvre pour minimiser ces échanges. Pour cela, les protocoles géographiques cherchent à réduire ces informations en précisant la direction vers laquelle se dirigent les informations de contrôle. En fait, en connaissant la position de la destination, les paquets qui vont à l'opposé de la destination ou au-delà de la position de la destination, sont généralement peu utiles. Dans ce cas, faire en sorte que ces paquets ne se propagent pas dans de mauvaise direction et soient plus centrés vers la position de la destination constitue le point clé des protocoles utilisant des informations de **localisation** [38].

Les protocoles de routage géographiques utilisent donc des coordonnées géographiques (par exemple, fournies par GPS) afin d'acheminer les messages de façon efficace vers la destination, en routant les messages dans la direction de la destination. Pour atteindre cet objectif, les coordonnées géographiques des nœuds sont incluses dans les tables de routage. Concrètement, un nœud inclut l'adresse IP et la position de la destination (fournie par le protocole de routage) dans le message de données à expédier, et envoie ce message dans la direction de la destination. Les nœuds intermédiaires répètent le même mécanisme jusqu'à ce que le message atteigne la destination. Les protocoles de routage géographique réduisent ainsi l'espace de recherche pour diminuer le nombre de nœuds qui transmettent des informations de routage. Cependant, réduire l'espace de recherche peut entrainer un échec à la découverte d'une route même si elle existe. Les protocoles de routage géographique combinent deux types de protocole pour pallier ce problème. Le premier est utilisé pour chercher une route en réduisant l'espace de recherche, le second diffuse généralement l'information de routage sur la totalité du réseau (comme le protocole AODV) en cas d'échec du premier.

Nous donnons ci-après des exemples représentatifs de protocoles géographiques en reprenant trois protocoles qui sont respectivement de type proactif, réactif et hybride :

– Le protocole DREAM (*Distance Routing Effect Algorithm for Mobility*) change la manière de concevoir les protocoles de routage. Les tables de routage, qui sont mise à jour de façon proactive, ne contiennent pas le prochain nœud ou le chemin pour joindre une destination mais les informations de localisation de chaque nœud. Lorsqu'un nœud veut transmettre un paquet de données, il utilise les informations de localisation concernant la destination dans sa table et envoie le paquet uniquement dans sa direction. Ce protocole repose donc

principalement sur l'échange des informations de localisation entre les nœuds. Pour cela, il réduit ces informations suivant deux constats [38]:

• ***La distance*** : Deux nœuds éloignés ressentent moins leur déplacement respectif que deux nœuds proches. Par conséquent, chaque nœud a besoin de mettre à jour sa localisation moins souvent vis-à-vis des nœuds éloignés.

• ***La mobilité*** : Tout nœud qui se déplace doit mettre à jour sa position. Cela doit être fait souvent d'autant plus qu'il se déplace rapidement.

Dans DREAM, chaque nœud émet périodiquement les informations de position le concernant (ses coordonnées, sa vitesse, etc.). Pour réguler la distance de propagation de ses messages de contrôle, chaque nœud marque le paquet par une certaine distance. Lorsqu'un nœud le reçoit, il calcul la distance que le paquet a parcouru. Si elle est plus grande que celle marquée dans le paquet, il est supprimé, sinon il est propagé. Ces paquets sont transmis d'autant plus souvent que la distance marquée est faible. La fréquence à laquelle les nœuds émettent les paquets de contrôle est fonction de la vitesse de déplacement du nœud lui-même. Plus il se déplace rapidement, plus il transmet souvent les informations sur sa position.

Lorsqu'un nœud a besoin d'envoyer un paquet à un autre nœud, il détermine l'ensemble des voisins permettant d'atteindre la destination avec une probabilité p. Il déclenche un temporisateur et émet le paquet. Lorsqu'un nœud reçoit un paquet, il vérifie s'il en est le destinataire. S'il ne l'est pas et qu'il est dans la direction de la destination, il propage ce paquet. Chaque nœud recevant un paquet et se trouvant dans la direction de la destination, détermine les voisins auxquels il transmet le paquet. Pour cela, il extrait de sa table de localisation, les paramètres de mobilité et de position concernant la destination. Ces paramètres sont le temps de mise à jour de l'information $t0$, la distance r le séparant de la destination, la vitesse de la destination v et l'angle γ entre l'axe des abscisses et la droite passant par lui et la destination. Le nœud peut facilement en déduire les voisins dont leurs positions se situent dans l'espace représenté par l'intervalle $[\gamma - \alpha, \gamma + \alpha]$. α est déterminé par la formule suivante : $\alpha = \arcsin v\,(t1 - t0) / r$. En connaissant la densité de probabilité de la vitesse de la destination, il est possible de calculer un α tel que la probabilité de trouver la destination dans la direction $[\gamma - \alpha, \gamma + \alpha]$ soit $\geq p$ avec $0 < p \leq 1$. Pour cela, il revient à déterminer la probabilité telle que : $P\,(x \leq (t1 - t0)\,v) \geq p$.

– Le protocole LAR (*Location-Aided Routing protocol*) est de type réactif. Il propose deux schémas pour limiter la surcharge du réseau induite par la localisation d'un nœud, en définissant soit une zone de recherche, soit la distance à la destination du nœud. Pour cela, chaque schéma va réduire l'espace de recherche en fonction de la connaissance de la position de la destination ainsi que d'autres paramètres comme sa vitesse et le temps de rafraîchissement de l'information. Dans les deux schémas du protocole LAR, lorsque la position de la destination n'est pas connue, le protocole diffuse une requête de création de route à la totalité du réseau. Une fois une route trouvée, la destination insère dans sa requête de confirmation de route sa position ainsi que des paramètres tels que la vitesse, le temps où la réponse est effectuée et le sens de déplacement. Le protocole de routage LAR est pleinement utilisé lorsqu'une liaison sur un chemin est rompue [38].

- ***Schéma 1*** *:* À partir de la dernière position de la destination (au temps *t0*) et de sa vitesse *v*, la source détermine une zone probable où doit se situer la destination. En effet, si la destination ne se déplace que dans un sens, elle peut se trouver à l'instant *t1* dans un cercle de rayon *v* (*t1* – *t0*). La source détermine à partir de sa position (*Xs*, *Ys*) et du cercle où est susceptible de se trouver la destination, une zone rectangulaire qui fera office de zone de recherche dans laquelle sont déterminées les routes. Cette zone rectangulaire est donc la plus petite zone pouvant contenir la source et le cercle susceptible de recevoir la destination. Une fois la zone rectangulaire calculée par la source, elle ajoute dans sa requête de création de route, les coins du rectangle. Lorsqu'un nœud reçoit la requête, il peut déterminer à partir de sa position s'il se trouve ou non dans cette zone. S'il s'y trouve, il traite la requête et la diffuse à ses voisins. Dans le cas où elle est extérieure à cette zone, la requête est supprimée réduisant ainsi le nombre d'informations de contrôle échangées.
- ***Schéma 2*** *:* La distance entre un nœud et la destination est l'élément déterminant dans la propagation d'une requête. Un nœud ne transmet la requête que si sa distance à la destination est plus faible que la distance à la destination du nœud qui lui a transmis la requête. Lorsqu'une la source *S* veut envoyer un paquet à un nœud destinataire et qu'elle ne connaît pas une route vers ce nœud, elle émet une requête de détermination de route en y insérant la position de la destination et sa distance *DISTS* vis-à-vis de cette dernière. Quand un nœud *I* reçoit cette requête, il détermine s'il doit la traiter et la retransmettre ou non. Pour cela, il détermine sa propre distance à la destination *DISTI* et vérifie si elle est inférieure à la distance du nœud *J* qui a transmis le paquet avec la formule : $DISTJ + \delta \geq DISTI$. Le paramètre δ permet d'ajuster l'expansion de la zone de recherche. En l'augmentant, il peut ainsi être employé pour accroître la chance de trouver une route.

Dans ces deux schémas, une route peut ne pas être trouvée. Cet échec peut être dû soit à une zone de recherche trop petite, soit parce que les paramètres sur lesquels la création de la zone est basée sont erronés ou tout simplement ne sont plus à jour. Un tel échec se révèle des plus dommageables, car le but premier d'un protocole de routage est de trouver une route si elle existe. Pour pallier ce problème, la source utilise un protocole réactif plus conventionnel tel que AODV ou DSR lors d'un tel échec.

– Le protocole ZHLS (*Zone-based Hierchical Link State routing*) est un protocole hybride. Il décompose, de façon statique, le réseau en zones disjointes à partir d'une carte. Chaque nœud connaît la topologie de sa zone (approche proactive) et connait aussi les nœuds jouant le rôle de passerelle entre les zones. Ces nœuds passerelles sont connus par les autres nœuds puisqu'ils diffusent sur l'ensemble du réseau les informations concernant les zones qu'ils connectent. Ainsi, chaque nœud du réseau connaît le découpage du réseau en zones et les nœuds donnant accès à ces zones. Lorsqu'un nœud désire connaître la localisation d'un autre nœud, il vérifie que le nœud ne se trouve pas dans sa zone. Si c'est le cas, il envoie une requête de localisation vers toutes les zones présentes dans le réseau en utilisant les informations correspondant aux nœuds passerelles pour atteindre ces zones [97].

4.3.2. Protocoles de routage utilisant le concept d'agent mobile

Plusieurs autres protocoles de routage ont été proposés dans la littérature, qui s'adaptent plus ou moins aux différents contextes caractérisant les réseaux mobiles ad hoc, tels que la taille du réseau, sa densité, la connectivité des nœuds, leurs degrés de mobilité, etc. Ce qui nous intéresse beaucoup plus dans ce chapitre, c'est les travaux de recherches sur le routage d'information qui cherchent à réagir efficacement à ces différentes situations en se basant sur le concept d'agents mobiles. Ci-dessous quelques uns de ces travaux :

– MARIAN (*Mobile Agents for Routing in Ad-hoc Networks*) est une architecture proposée par Migas et al. [72,73] qui utilise une combinaison d'agents statiques et d'agents mobiles pour générer la meilleure route à travers le réseau. L'objectif de MARIAN est de permettre aussi une migration des agents mobiles à travers de petits équipements mobiles tels que les PDA. En parcourant le réseau, les agents mobiles collectent des informations à partir des agents statiques, qu'ils utilisent pour faire les calculs nécessaires afin de mettre à jour les tables de routage et de découvrir de nouvelles routes. Ils doivent aussi informer les agents statiques et les autres agents mobiles des changements qui surviennent dans le réseau. Shekhar et Ramanatha [106] ont présenté un modèle similaire qui combine les deux types d'agents pour le routage et pour le contrôle de la congestion dans les réseaux ad hoc. Kakkasageri et al. [57] proposent aussi une architecture basée sur un ensemble d'agents statiques et mobiles pour la découverte à la demande, de routes stables entre les nœuds mobiles. Le système d'identification d'une nouvelle route stable est basé sur le temps d'expiration d'une route, calculé en fonction du degré de mobilité d'un nœud et de sa portée de communication. Le système est composé de trois types d'agents, un agent statique de gestion de nœuds, un agent statique de gestion de liaisons et un agent mobile de découverte de routes.

– Bandyopadhyay et Krishna [7] proposent d'utiliser les agents mobiles pour le transfert de messages en différé entre des nœuds mobiles, dans le cas d'un réseau mobile ad hoc très large et fortement dynamique. L'agent mobile dans ce contexte, agirait en tant que messager qui migrerait à partir d'une source et transporte avec lui le message vers une destination donnée. L'agent a une connaissance approximative sur la position de la destination et exploite la mobilité des nœuds comme un véhicule pour migrer d'une place à une autre, afin de trouver la destination et lui remettre le message. Chaque nœud dans le réseau est supposé connaître son identificateur, la portée de sa transmission et sa position courante (à travers le système GPS). Chaque nœud transmet périodiquement ces informations à ses voisins. Dans le cas ou la destination est déconnectée du réseau pour quelque temps, la livraison du message sera reportée et l'agent attendra dans un nœud intermédiaire approprié jusqu'à la reconnexion du destinataire. Cependant, générer un agent mobile pour chaque message à échanger dans le réseau semble très couteux dans les réseaux ad hoc, à moins que les agents mobiles sont de très petites taille et n'escortent pas n'importent quel message.

– D'autres travaux proposent des algorithmes de routage qui utilise les colonies de fourmis (*Ants*) [69,112], représentés par de très petits agents mobiles qui sont lancés dans le réseau à des intervalles réguliers et vers des destinations choisies aléatoirement. Ces agents mobiles se

déplacent dans le réseau d'un nœud à un autre, tout en mettant à jour les tables de routage des nœuds qu'ils visitent, à l'aide d'informations collectées lors de leurs parcours dans le réseau. Nous citons AntHocNet [31], ANSI (*Ad hoc Networking with Swarm Intelligence*) [88], ARA (*Ant-colony-based Routing Algorithm*) [45] et Termite [105], qui utilisent ce type d'agents mobiles, à la demande, pour la découverte de routes. Dhillon et al. proposent l'algorithme W_AntNet [29], qui est une adaptation de l'algorithme de routage AntNet (paragraphe 3.2), de Di Caro et Dorigo [30] aux réseaux mobiles ad hoc. Marwaha et al. [68] exploitent aussi ce concept en proposant une combinaison du protocole réactif AODV avec un mécanisme distribué de découverte de topologie utilisant des agents mobiles de type Ants.

– Un autre travail qui combine la technologie d'agent mobile avec les protocoles de routage connus pour les réseaux ad hoc, est celui de Rajabzadeh et al. [87] qui ont proposé une amélioration du protocole DSR (*Dynamic Source Routing*) à l'aide d'un système multi-agent mobile.

– Denko [28] présente une architecture hiérarchique (clustering) pour le routage d'informations, qui utilisent plusieurs types d'agents mobiles pour la collecte et la maintenance des informations de routage et de clustering sur chaque nœud mobile visité dans le réseau.

4.4. Le clustering dans les réseaux ad hoc

Le contrôle de topologie dans les réseaux ad hoc est d'un apport important dans l'amélioration des performances de gestion d'un réseau d'une manière générale, et des protocoles de routage en particulier. Il vise à maintenir une topologie adéquate en maîtrisant les liens à inclure dans le réseau.

Le principe du clustering, qui est une technique de contrôle de topologie, consiste à partitionner le réseau en regroupant des nœuds proches géographiquement en clusters (un cluster est donc un groupe de nœuds). Un chef de groupe (*cluster head*) est élu alors pour chaque groupe selon un processus distribué d'élection de chef (figure 20) [104]. Il maintient une liste des nœuds appartenant à son groupe. Il maintient aussi un chemin vers chacun de ces nœuds qui est mis à jour de façon proactive. De façon similaire, un chef de groupe maintient une liste de passerelles vers les groupes voisins. (Les passerelles sont des nœuds qui font le lien avec un autre cluster).

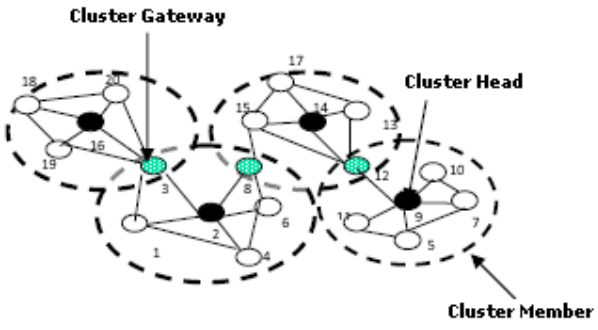


Figure 20 : *Exemple de clustering.*

Les chefs de groupe sont des nœuds dominant dans le réseau, ils peuvent être en charge d'allocation d'adresses, de slots de communication et du routage dans leurs clusters ou vers les autres chefs de groupe du réseau via les passerelles. Ainsi, un réseau ad hoc peut être représenté de manière logique comme étant un ensemble de clusters où chaque cluster est identifié par une tête, le chef de groupe, qui a le rôle d'une station de base et dont l'ensemble constitue la colonne vertébrale virtuelle du réseau (figure 21).

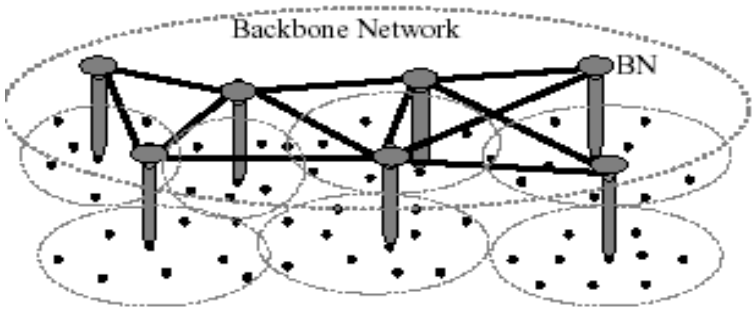


Figure 21 : *Illustration d'une colonne vertébrale dans un réseau ad hoc.*

La structuration d'un réseau Ad hoc en clusters offre plusieurs avantages pour les réseaux mobiles ad hoc. Elle permet de simplifier les tâches de gestion, notamment la décentralisation de la gestion qui permet la tolérance aux fautes. Elle améliore aussi la gestion du routage et de la mobilité. Elle réduit la charge du réseau en limitant l'étendue des diffusions aux seuls clusters, permettant ainsi l'optimisation de la bande passante. Le clustering qui assure la maintenance des informations de la topologie du réseau, permet aussi de supporter le passage à l'échelle (extension du réseau vers de grandes tailles). En fournissant une infrastructure virtuelle pour un réseau dynamique, le clustering aide à traiter plus efficacement l'allocation de ressources. Enfin, la structuration d'un réseau est un des outils principaux pour sauvegarder l'énergie dans chaque nœud du réseau, ce qui aboutit à prolonger la vie du système.

4.4.1. Techniques de clustering

Différentes techniques de partitionnement ont été développées et proposées dans la littérature. Le point de divergence principal entre elles est l'heuristique utilisée pour déterminer quel nœud sera le chef de groupe. En effet, certaines techniques se basent sur les identifiants des nœuds, leur degré de connectivité, leur vitesse de déplacement, leur capacité énergétique, leur position géographique ou la puissance de transmission, tandis que d'autres préfèrent utiliser un critère de poids pour chaque nœud qui représente une combinaison d'un certains nombre de ces paramètres. Ces techniques cherchent à éviter que deux clusterheads ne soient voisins directs, ce qui permet une bonne dispersion spatiale des clusterheads à travers le réseau, alors que d'autres cherchent à avoir au moins un voisin clusterhead pour chaque nœud du réseau, ceci permet des communications plus rapides entre toute paire de nœuds.

Cependant, pour exploiter au mieux le clustering, il est important que la technique de clustering garde la stabilité des clusters aussi longtemps que possible. Les changements fréquents dans l'architecture des clusters peuvent être couteux en bande passante et en énergie. En effet, la phase de maintenance est l'étape la plus importante dans la clustering, car les nœuds étant en mouvement, un chef de groupe peut rompre la connectivité avec les nœuds qui lui sont rattachés. De même, il peut cesser de fonctionner car la station peut s'éteindre ou ne plus avoir suffisamment d'énergie pour réaliser ce rôle. Lors de la perte de connectivité, cet ensemble de nœuds doit élire un nouveau chef de groupe. Pour cela, il suffit de réappliquer l'algorithme de plus faible identifiant ou de plus fort degré de connectivité ou autre algorithme sur l'ensemble des nœuds sans chef de groupe. Pour vérifier la connectivité entre un chef de groupe et les nœuds qui lui sont rattachés, des paquets de contrôle sont échangés périodiquement.

Ci-dessous quelques techniques de clustering :

– LCA (*Linked Cluster Algorithm*) [28] est un des premiers travaux sur le clustering dans les réseaux mobiles ad hoc. L'heuristique utilisé dans LCA assigne un identificateur unique à chaque nœud et choisie le nœud avec le plus petit identificateur comme chef de groupe. L'algorithme LCA est rapide et simple, en plus les clusters sont relativement stables. Son inconvénient est l'épuisement de la batterie vu que les nœuds avec le plus petit ID risquent de rester clusterhead pour une longue durée de vie. La figure 22 montre sur un exemple d'un réseau à 10 nœuds, l'algorithme LCA ou les nœuds 1, 2 et 4 sont des clusterheads et les nœuds 8, 9 sont des passerelles.

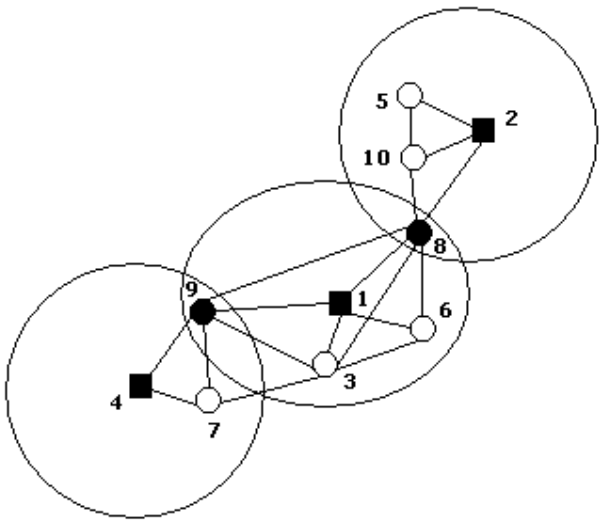


Figure 22: *Exemple de formation de clusters (LCA).*

– Gerla et Tsai [28] ont proposé d'utiliser le plus haut degré de connectivité des nœuds (leur voisinage) comme heuristique pour élire le chef de groupe. Ceci permet d'assurer une disponibilité des chemins pour le routage. Cependant, l'inconvénient de cet algorithme est que

la taille des clusters est non limitée, ce qui dégrade les performances du système. La figure 23 montre sur le même exemple de la figure 22, l'algorithme de Gerla et Tsai où les nœuds 5,7 et 8 sont des clusterheads et les nœuds 2, 3, 9, 10 sont des passerelles.

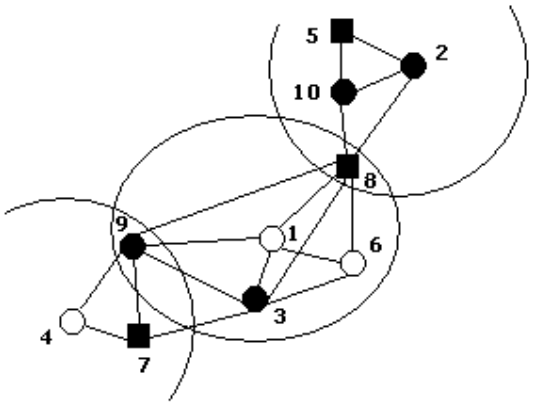


Figure 23: *Exemple de formation de clusters (Algorithme de Gerla et Tsai).*

– MOBIC [8] et MobDHop [37] sont deux techniques qui utilisent la mobilité des nœuds et les caractéristiques de la liaison comme heuristiques pour le clustering. Les nœuds avec le plus petit degré de mobilité (vitesse de déplacement) sont sélectionnés comme chef de groupe. Comme ces nœuds ne bougent pas ou très lentement en comparaison des autres nœuds, le partitionnement est garanti d'avoir une longue durée de vie et en conséquence le traitement associé à la maintenance des clusters en environnement mobile est minimisé.

– DCA *(Distributed Clustering Algorithm)* [110] est un algorithme qui utilisent une heuristique basée sur le poids des nœuds. L'idée proposée par Basagni en 1999 est assez différente car il s'est inspiré des deux phases de partitionnement, l'initialisation et la maintenance. La phase d'initialisation décrit l'ensemble des procédures nécessaires à la formation des clusters et la phase de maintenance introduit d'autres procédures pour le maintien des clusters face à la mobilité des nœuds. Le poids tel qu'il est défini par Basagni est un nombre entier strictement positif associé à chaque nœud; plus le poids d'un nœud est élevé, meilleur serait ce nœud pour le rôle de clusterhead. Le critère de poids peut représenter une combinaison d'un certains nombre des heuristiques précédentes, comme il peut être, par exemple, inversement proportionnel à la vitesse de déplacement d'un nœud, qui est une généralisation de l'algorithme de Denko [28]. D'autres algorithmes similaires basés sur le poids ont été proposés tels que WCA *(Weitghted Clustering Algorithme)* et PC (*Passive Clustering*).

4.4.2. Clustering multi niveau (hiérarchique)

Le clustering hiérarchique adopte le même principe que le clustering classique en structurant dynamiquement le réseau en plusieurs niveaux de clustering. En général, ces techniques s'exécutent sur des réseaux ad hoc non homogènes. Ils ont pour particularité d'avoir deux types différents de nœuds qui se différencient en fonction de leurs caractéristiques. Ainsi, les nœuds à grandes capacités servent à la formation du réseau dit à

backbone (figure 21). Ils sont appelés nœuds de backbone (*BN : Backbone Node*) s'ils forment la dorsale, ou nœuds de backbone capable (*BCN : Backbone Capable Node*) s'ils sont éligibles pour la formation de la dorsale. Ces nœuds ont pour particularité d'avoir une capacité importante de fonctionnement, en offrant des ressources processeur, des capacités mémoire et une autonomie importantes. De même, ils possèdent deux modules sans fil fonctionnant sur des fréquences différentes (l'un à faible puissance, et l'autre à puissance élevé). Le module à faible puissance est utilisé pour fonctionner avec les nœuds du réseau ad hoc alors que l'autre module est employé pour la communication entre les nœuds de la dorsale. Du fait que ces deux modules opèrent sur des fréquences différentes, ils n'interfèreront pas entre eux. Dans le clustering hiérarchique, les BNs peuvent être reconvertis en BCNs dans le but de réduire le nombre de BN actifs pour optimiser les ressources et l'énergie. La conversion de BCNs en BNs est utilisée aussi pour construire le cœur du réseau et fournir une connectivité désirée, tout en permettant de couvrir le plus large ensemble de nœuds ad hoc. Aussi, un cluster peut dynamiquement se fusionner avec un autre cluster ou s'éclater en fonction du nombre de nœuds [38].

Le clustering hiérarchique est très efficace dans la collecte d'informations pour les systèmes de gestion de réseau. Il est utile de collecter les informations à travers une arborescence. Ceci permet aux nœuds intermédiaires de composer les données à partir des données reçus des différents nœuds situés dans les niveaux inférieurs de l'arborescence, avant de les remettre au niveau supérieur. Cette composition de données réduit effectivement le flux d'informations collectées dans le réseau hiérarchique. Deux exemples d'algorithmes dans ce type de clustering [38] sont MBNP (*Mobile Backbone Protocol*) et MMWN (*Multihop Mobile Wireless Networks*).

4.4.3. Agent mobile pour le clustering

Comme les agents mobiles sont des entités autonomes et intelligentes, ils peuvent être utilisés aussi pour la construction de clusters dynamiques et adaptifs dans les réseaux mobiles ad hoc [30]. Ci-dessous quelques techniques de clustering basées sur la technologie d'agents mobiles.

– AHPCM (*Analytical Hierarchy Process Clustering Mobile agent protocol*) [104] est un protocole basé sur le calcul des poids relatifs à tous les nœuds du réseau, et utilisant des agents mobiles pour la collecte d'informations relatives à cette heuristique à travers les nœuds du réseau, tout en mettant à jour les bases de données sur la maintenance du clustering des nœuds visités.

– Sugar et Imre [113] proposent une technique de clustering où à chaque cluster est associé un agent mobile, qui est capable de prendre des décisions de modification d'adhésion des nœuds au cluster associé, et de gérer le partitionnement ou la fusion de clusters. La formation de clusters peut se faire selon différents critères. Pour réduire la surcharge due aux messages de contrôle liés au clustering, la communication se fait uniquement entre les agents voisins.

– Denko [28] propose une architecture qui utilise deux types d'agents mobiles pour réaliser respectivement l'opération de routage et de clustering. Ils collectent les informations de

routage et de clustering, et maintiennent périodiquement les tables correspondantes au niveau de chaque nœud du réseau visité. Les informations stockées dans les tables de routage sont utilisées pour le routage intra-cluster et inter-cluster. Le routage inter-clustering est réalisé à travers les chefs de groupe. Les agents mobiles participent aussi dans l'ajustement des tailles des clusters et leurs maintenances. Comme dans MOBIC et MobDHop, Denko utilise la mobilité des nœuds et les caractéristiques de la liaison comme heuristique pour le clustering.

5. Conclusion

Dans ce chapitre nous avons présentés un état de l'art des différents travaux réalisés sur la gestion des réseaux informatiques. Nous nous sommes intéressés plus particulièrement aux réseaux mobiles ad hoc. La gestion dans ces réseaux est une tâche extrêmement difficile, comparée aux réseaux filaires, et ceci à cause de leurs propres caractéristiques. Ces réseaux doivent donc disposer de leurs propres systèmes de gestion, puisque les systèmes de gestion classiques pour les réseaux filaires ne leurs sont plus adaptés. Ainsi, le défi majeur des systèmes de gestion pour les réseaux ad hoc est de parvenir à proposer une qualité de service à ses utilisateurs, comparables à ceux des réseaux classiques et cela de façon transparente. Comme présentée dans ce chapitre, la nouveauté dans les systèmes de gestion des réseaux ad hoc est l'introduction de la technologie d'agents mobiles, qui semble d'après les travaux présentés, une alternative intéressante au classique client/serveur dans certains types de configuration. Elle a été utilisée à différents niveaux de gestion, dans le contrôle des réseaux ad hoc, dans les systèmes de sécurité, dans les protocoles de routage d'information et même dans le contrôle de topologie des réseaux (clustering). Le concept d'agent mobile sera détaillé dans le chapitre suivant.

Le problème que nous avons traité dans cette thèse, dans le cadre de la gestion des réseaux, étant la localisation d'agents mobiles dans les réseaux ad hoc, nous avons mis l'accent dans ce chapitre sur le routage géographique basé sur le mécanisme de localisation, qui fera l'objet du quatrième chapitre.

Comme l'une de nos approches proposées à ce problème est basée sur une architecture hiérarchique du réseau, nous avons aussi présenté les techniques de structuration en clusters appliquées aux réseaux ad hoc, qui offrent plusieurs avantages pour ces réseaux, notamment la décentralisation qui permet de simplifier les tâches de gestion.

Chapitre 3

Technologie d'Agent Mobile

1. Introduction

Avec le développement des réseaux, il est devenu courant que le code exécutable et les données sur lesquelles il travaille ne soient pas au même endroit. Dans les architectures traditionnelles (client-serveur, par exemple) ce sont les données qui sont déplacées vers le code. Cependant, pour toutes sortes de raisons (volume de données, connexion, etc.) l'inverse apparaît parfois préférable: c'est la notion du code mobile.

La technologie du code mobile, issue du domaine de l'intelligence artificielle sous le vocable d'agent mobile, est en plein développement actuellement [96]. Elle est prometteuse dans de nombreux domaines d'applications, en particulier dans le domaine de la réalisation d'applications distribuées sur les réseaux à grande échelle (tels que la recherche d'informations et le commerce électronique sur le Web) et dans le domaine de l'informatique nomade (réseaux mobiles).

Le paradigme des agents mobiles propose d'utiliser la migration d'activité (du code) en supprimant la contrainte de connexion constante entre les deux parties en interaction (client/serveur), qui n'est pas évidente ni dans les réseaux de grande envergure ni pour les stations nomades. On demande aux deux parties d'être connectées seulement durant la phase de migration du code. Ainsi, un agent mobile peut se déplacer dans un réseau de machines offrant des services pour réaliser une tâche complexe.

La figure 24 illustre ce paradigme [96] : un client donne une mission à un agent, qui pour la réaliser se déplace dans le réseau de machines accédant localement aux services offerts par ces machines. On peut distinguer trois phases :

• L'activation de l'agent mobile avec la description de sa mission.
• L'exécution de la mission par l'agent qui se déplace pour accéder aux services.
• La récupération éventuelle des résultats de l'agent mobile.

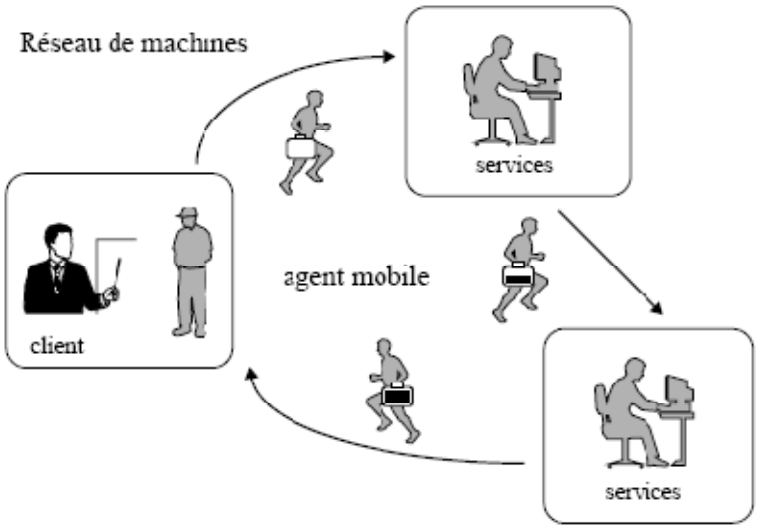


Figure 24 : *Le paradigme des agents mobiles.*

2. De l'architecture client/serveur à la technologie d'agent mobile

La technologie d'agent mobile constitue une alternative intéressante à l'architecture traditionnelle client/serveur, et ce dans beaucoup de situations notamment dans les environnements mobiles. Son apparition fait suite à une succession d'étapes dans lesquelles des mécanismes nouveaux ont été introduits au paradigme client/serveur que nous présentons ci-dessous, en commençant par la première méthode qui fût disponible lors de la construction des réseaux d'ordinateurs : l'échange de message.

2.1. L'échange de message

L'échange de message (*Message Passing*), qui repose directement sur le service de communication offert par un réseau de machines, est le mécanisme élémentaire fourni par le système pour l'interaction à distance entre des processus s'exécutant dans le réseau [27, 81]. Le principe du modèle des processus communicants est assez trivial. Le système met à disposition des processus un service de messagerie. Un client qui veut utiliser un service doit mettre en œuvre le protocole de communication imposé par le serveur. La figure 25 montre le mécanisme des processus communiquant via le service de messagerie [27].

L'échange de message entre processus est un mécanisme puissant offrant beaucoup de souplesse pour la programmation des échanges. Néanmoins le programmeur doit prendre en compte tous les problèmes liés aux techniques de communication comme la gestion des erreurs de transmission et des connexions, la construction et l'interprétation des messages, et la synchronisation entre les intervenants. En d'autres termes, l'échange de message nécessite de définir un protocole de communication stricte entre le serveur et ses clients. C'est le mécanisme de base utilisé dans les systèmes répartis. Il constitue aussi un modèle très utilisé pour les applications de calcul sur les machines parallèles fortement couplées [27, 81].

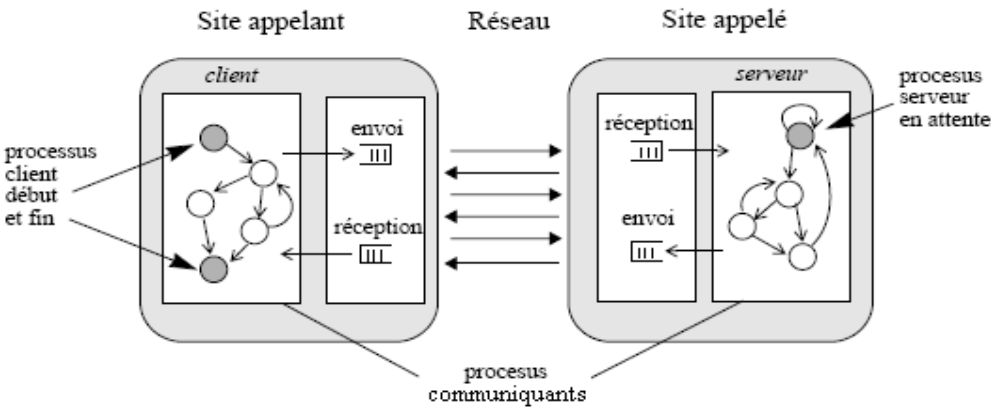


Figure 25: *Principe des processus communicants.*

2.2. L'appel de procédure à distance

L'appel de procédure est une méthode de structuration des programmes permettant le transfert du contrôle et des données entre les différentes fonctions d'une application. Cette méthode a été élargie pour les applications réparties, en offrant la possibilité de réaliser des appels à distance grâce au mécanisme d'appel de procédure à distance ou RPC (*Remote Procedure Call*). Ce mécanisme est une abstraction permettant aux développeurs de ne pas manipuler directement le service de communication réseau mais d'appeler une procédure distante comme si elle était locale. C'est un mécanisme de plus haut niveau que l'échange de messages, facilitant la construction des applications réparties. Notons cependant, qu'il faut connaître à l'avance la procédure que l'on souhaite utiliser.

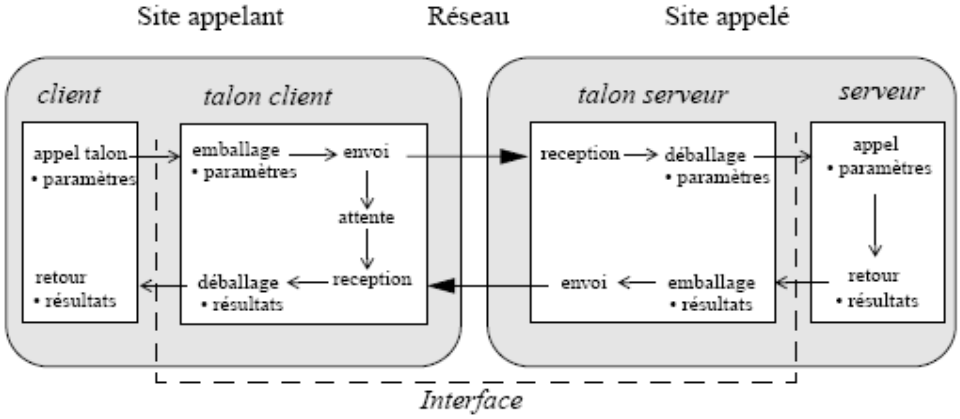


Figure 26: *Principe de l'appel de procédure à distance.*

Le principe d'interaction est synchrone. La machine cliente appelle une procédure qui est sur la machine serveur et attend la réponse. L'environnement appelant est suspendu durant la transmission des paramètres, l'exécution de la procédure et la transmission des résultats. La figure 26 illustre le principe de fonctionnement qui utilise le concept de *talon* [81]. Un talon est un programme chargé de représenter localement une activité distante afin de cacher la répartition. Le talon client représente le serveur sur le site appelant, il gère l'emballage et l'attente liés à l'exécution de la procédure ; symétriquement le talon serveur représente les clients sur le site des serveurs, il gère l'emballage et la prise en compte des demandes d'exécution de la procédure.

2.3. L'invocation d'objet à distance

Un objet est une unité de structuration qui regroupe des données et des méthodes permettant de manipuler ces données. Les systèmes à objet proposent directement cette abstraction aux applications. Le modèle d'exécution est fondé sur le mécanisme d'invocation de méthode sur un objet à partir de sa référence dans le système. Pour réaliser, ce modèle le système fourni un mécanisme de liaison dynamique de l'objet dans le contexte de l'application appelante. L'invocation d'objet à distance est une extension de ce modèle à un environnement distribué permettant de passer des paramètres par références. Le principe de l'invocation d'objet à distance est fondé sur trois éléments : un schéma de désignation commun pour tous les objets, un mécanisme de liaison dynamique des objets dans le contexte d'une activité et l'utilisation d'un représentant de l'activité sur le site cible.

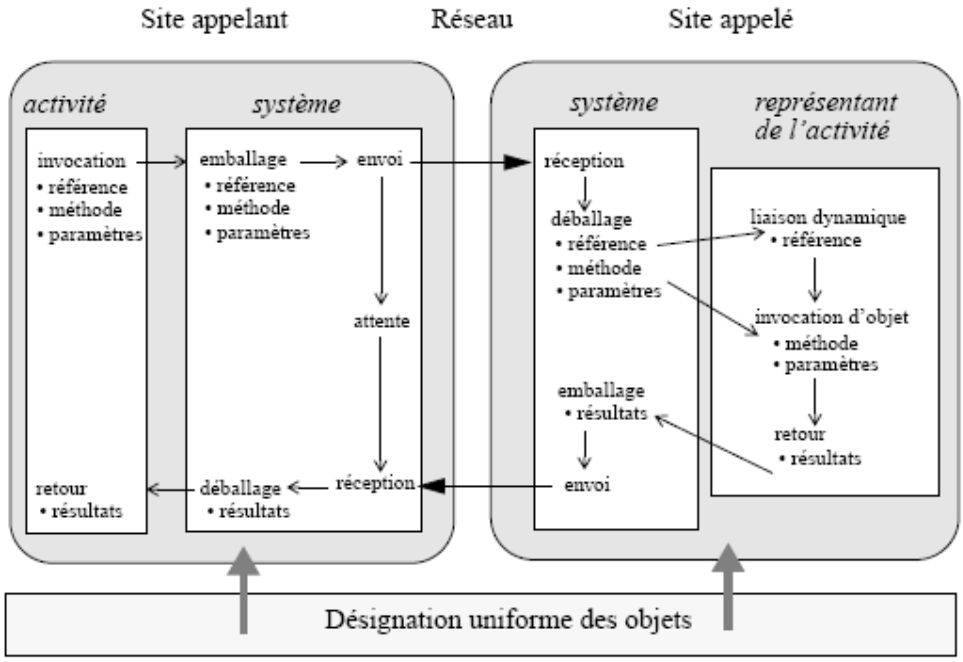


Figure 27: *Principe de l'invocation d'objet à distance.*

La figure 27 illustre ce principe [81]. Une activité réalise l'appel d'une méthode d'un objet à partir de sa référence avec des paramètres. Le système détecte que l'objet référencé est distant grâce au service de désignation des objets, et déclenche la procédure d'invocation à distance. La référence, le nom de la méthode et les paramètres sont emballés et transmis au site où se trouve l'objet. Le système sur le site cible reçoit cette demande et instancie un représentant de l'activité. Cette nouvelle activité effectue une liaison dynamique de l'objet désigné par la référence dans son contexte d'exécution. Elle réalise ensuite l'invocation locale de l'objet par l'appel de la méthode avec les paramètres. Cette exécution peut déclencher d'autres invocations d'objet locales ou distantes. Lorsque chaque invocation distante se termine, le représentant est détruit et l'activité initiatrice reçoit les résultats et peut continuer son exécution.

2.4. Introduction de la mobilité

L'interaction de type client/serveur que nous venons de présenter se base sur des interactions distantes entre le client et le serveur. Nous présentons ici une autre manière de concevoir les interactions en introduisant la mobilité qui peut être vue comme une variation du modèle clients/serveur classique. Nous en présentons les différentes formes existantes en nous basant sur le déplacement d'un ou plusieurs éléments composant le schéma client/serveur (figure 28) [27], i.e., le code (le savoir faire), les ressources (applicables au code) ou l'unité d'exécution (qui traite le code). Nous prenons comme référence le placement de chacun de ces éléments avant et après l'exécution du service.

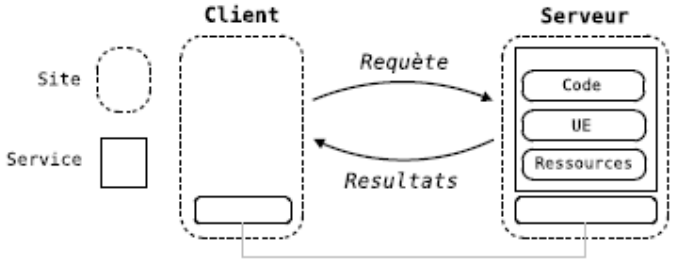


Figure 28: *Schéma d'organisation Client/Serveur.*

2.4.1. L'évaluation à distance

L'évaluation est un mécanisme disponible dans les langages interprétés permettant d'exécuter des expressions qui sont construites dynamiquement par un programme. L'évaluation à distance est une extension de ce mécanisme permettant l'envoi vers le serveur, du code (sous la forme d'une expression) par un client, propre au service à réaliser. Les ressources et l'unité d'exécution étant au démarrage sur le serveur, la mise en route du service est réalisée après que le serveur ait reçu le code à exécuter. Les résultats sont renvoyés une fois le service achevé, le code et l'unité d'exécution sont alors supprimés. La figure 29 illustre cette première utilisation de la mobilité [27].

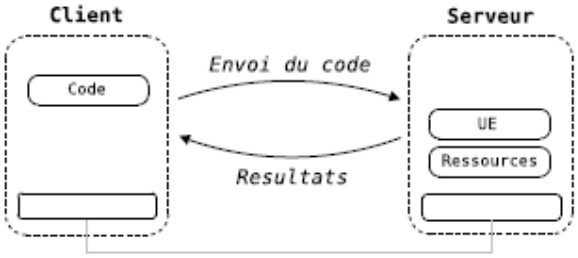


Figure 29: *Principe de l'évaluation à distance.*

Ce modèle de communication, par opposition aux modèles précédents, permet de supprimer les communications intermédiaires en déplaçant un programme sous la forme d'une expression. Néanmoins l'approche est synchrone, imposant au client et au serveur le maintien d'une connexion durant l'exécution complète du service. Comme exemples de ce modèle, nous pouvons citer les requêtes SQL, adressées à un serveur de base de données, et aussi les servlets Java déployés sur les serveurs http.

2.4.2. Code à la demande

Dans ce cas, le client dispose de l'unité d'exécution et des ressources mais pas du savoir faire (le code) qui va être récupéré auprès du serveur. Il s'agit donc de l'inverse du cas précédent. Ainsi, un client adresse une requête uniquement pour récupérer un code précis afin de l'exécuter localement avec les ressources présentes. Les trois éléments sont réunis sur le site client comme le montre la figure 30 [27]. Cette méthode permet d'étendre les fonctionnalités d'une application directement chez le client sans avoir besoin d'effectuer une nouvelle installation. L'exemple le plus répandu d'utilisation de cette méthode est le téléchargement d'applet Java à partir d'un serveur Web. Ces applets sont des classes Java présentes sur le site serveur qui sont téléchargées puis interprétées par la machine virtuelle du site client.

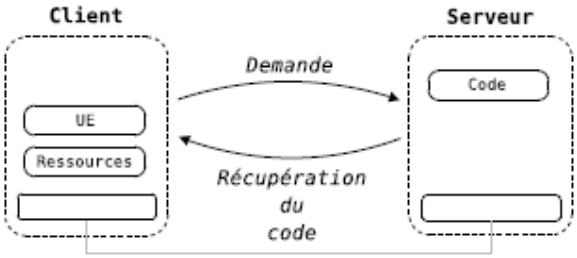


Figure 30: *Principe du code à la demande.*

2.4.3. La migration d'activité

La migration d'activité est définie comme le mouvement d'une entité en cours d'exécution vers une machine distante. Elle fournit un mécanisme d'exécution répartie puissant. Ce schéma est utilisé lorsque l'on souhaite déplacer un code avec son unité d'exécution comme le montre la figure 31 [27]. Ainsi, durant l'exécution du service, tous les éléments se retrouvent sur le site serveur. L'intérêt de cette méthode est qu'elle permet d'appliquer le même code à des ressources différentes et réparties sur le réseau sans avoir à le télécharger auprès du client à chaque fois.

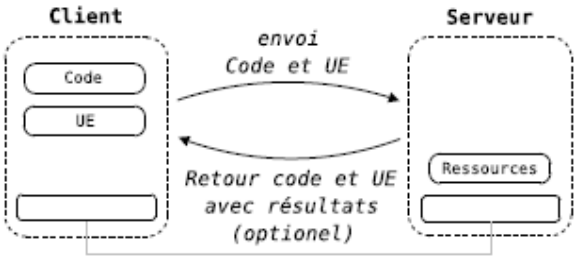


Figure 31: *Principe de la migration d'activité.*

2.5. Classification

Une action à distance résulte ainsi de l'exécution d'une portion de code sur un site cible. On identifie alors trois modèles d'exécution répartie à partir de la question [96]:
D'où vient le code de l'action et qui la réalise ?

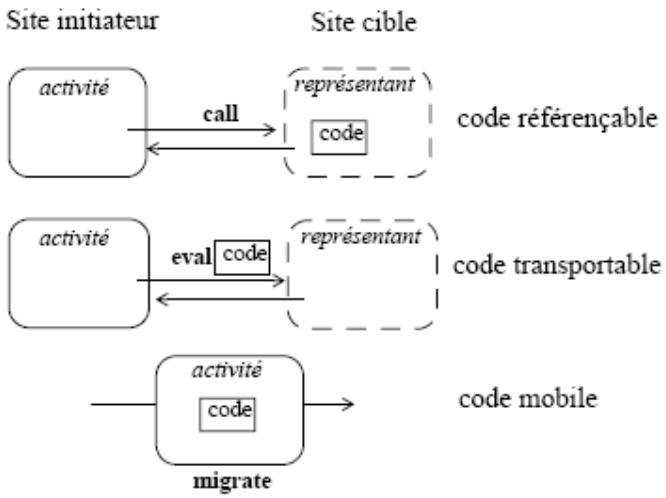


Figure 32: *Les modèles d'exécution répartie*

- ***le code référençable*** : l'activité envoie un message qui déclenche une action dont le code est déjà sur le site cible et qui est exécuté par un représentant distant;
- ***le code transportable*** : l'activité envoie un message qui déclenche une action dont le code est fourni par l'activité et qui est exécuté par un représentant distant;
- ***le code mobile*** : l'activité se déplace sur le site cible et réalise elle même l'action.

Les mécanismes présentés précédemment peuvent êtres classés selon ces trois groupes. L'échange de message, les appels de procédures à distance et l'invocation de méthode à distance sont des mécanismes qui demandent le placement préalable de programmes sur le site cible et se situent dans la classe du ***code référençable***. L'évaluation à distance et le code à la demande décrivent exactement le principe du ***code transportable***. La migration d'activité dans les systèmes répartis, utilisée dans les réseaux locaux, est un mécanisme de ***code mobile*** qui est très intéressant, parce qu'il permet un accès aux ressources du réseau et qu'il n'impose pas de gérer l'interaction avec un représentant distant durant l'exécution du code de l'action.

3. Notion d'agent mobile

3.1. Concept d'agent

L'utilisation du terme agent est très répandue en informatique. Le concept d'agent a donné lieu à de très nombreuses définitions notamment dans le domaine de l'intelligence artificielle. Parmi ces définitions nous pouvant citer les suivantes :

- ✓ « Un agent est une entité autonome, réelle ou abstraite, qui est capable d'agir sur elle-même et sur son environnement, qui, dans un univers multi-agent, peut communiquer avec d'autres agents, et dont le comportement est la conséquence de ses observations, de ses connaissances et des interactions avec les autres agents » [40].

- ✓ « Un agent est une entité logicielle ou physique à qui est attribuée une certaine mission qu'elle est capable d'accomplir de manière autonome et en coopération avec d'autres agents » [18].
- ✓ « Un agent est une entité active ***autonome*** agissant par ***délégation*** pour le compte d'un client » [81].

Cette dernière définition du terme agent qui est assez générale, insiste sur deux caractéristiques de la notion d'agent qui sont importantes [81]:

- ➢ L'agent est une entité qui agit par ***délégation*** ; l'agent doit respecter la stratégie du client vis-à-vis des choix qu'il est amené à faire, cela afin que le client soit responsable des tâches effectuées par l'agent.
- ➢ L'agent est une entité ***autonome*** qui contrôle ses actions et son état interne, et qui dispose de son propre environnement, c'est-à-dire que l'agent n'est pas lié constamment au client qui l'utilise, cela afin que le client ne dépend pas des contraintes vécues par l'agent.

Une autre propriété de l'agent est ce qu'on appelle la ***réactivité***, c'est-à-dire que l'agent perçoit l'environnement dans lequel il se trouve et peut répondre aux changements qui s'y produisent.

Il existe différent domaines de l'informatique utilisant le concept d'agent [81]:

- ✓ Les ***systèmes multi-agents*** *(SMA)* provenant des domaines de la robotique et de l'intelligence artificielle qui proposent de structurer une application en différentes fonctions représentées chacune par un agent. Les différents agents coopèrent pour la résolution des problèmes de l'application.
- ✓ Les ***agents d'interface*** provenant des domaines de l'interface homme-machine et de l'intelligence artificielle qui essayent de simplifier la vie de l'utilisateur, en automatisant des tâches réalisées couramment par l'observation des comportements répétitifs ou en surveillant des ressources comme le courrier électronique.
- ✓ Les ***agents mobiles*** provenant du domaine des réseaux dont l'objectif est la réalisation de tâches réparties sur un ensemble de machines interconnectées pour le compte d'un utilisateur ou d'une application.

Nous nous intéressons particulièrement dans cette thèse à la propriété de coopération (SMA) et de mobilité des agents, où nous essayerons de montrer plus loin l'apport de la mobilité des agents dans la gestion des réseaux mobiles ad hoc.

3.2. Système multi-agent

Un système multi-agent est un système informatique caractérisé par son environnement logiciel ou matériel, par ses éléments de base (ses agents) et par les mécanismes d'actions et d'interactions existant entre ces éléments et l'environnement en question.

La représentation d'un problème par un système multi-agent consiste à distribuer les tâches entre plusieurs entités autonomes, pouvant communiquer et collaborer afin d'exécuter la tâche globale à laquelle le système est dédié. Un système multi-agent peut être donc

considéré comme un ensemble d'agents qui coopèrent intelligemment pour réaliser une tâche commune. L'intelligence est ainsi répartie entre divers agents du système.

3.3. Définition d'un agent mobile

Un agent mobile est définit comme étant un agent répondant aux définitions ci-dessus (paragraphe 3.1) et capable de se déplacer d'une station à une autre dans un réseau pour accomplir sa tâche. Le schéma de mobilité des agents mobiles dérive ainsi de deux domaines différents, i.e. les agents venant de l'intelligence artificielle avec les systèmes multi-agents [18] et des systèmes distribués avec la migration de processus.

Dans [27] l'auteur reprend le concept d'agent mobile permettant de séparer l'aspect fonctionnel d'un agent de celui de la migration. Ainsi, il peut utiliser la description de la tâche générale d'un agent mobile comme un ensemble de sous-tâches à réaliser entre chaque migration. Il définit alors un agent comme un ensemble de comportements (fonctionnalité locale) et de transitions entre ses comportements. Une migration peut être vue comme une transition avec un changement de **localisation**.

3.4. Intérêts des agents mobiles

Les agents mobiles ont rapidement suscité un intérêt tout particulier dans les domaines de recherche portant sur les applications réparties. Très rapidement, cette nouvelle méthode de programmation a été évaluée afin de voir ce qu'elle pouvait apporter comme caractéristiques propres et ce qu'elle permettait d'améliorer par rapport aux méthodes de programmation plus classiques.

Le mécanisme d'agent mobile permet plusieurs gains non négligeables [13,27,81,91,96]:

- Une partie de l'application (représenté par un agent) s'exécute dans le réseau et peut bénéficier des ressources importantes qui le constituent, permettant ainsi une meilleure utilisation des ressources physiques mises à disposition ;
- L'activité (l'agent mobile) s'exécutant dans le réseau peut se rapprocher des données réparties permettant de réduire significativement, voir éviter, le transfert de données intermédiaires. En privilégiant les interactions locales, l'utilisation du réseau va se limiter principalement au transfert des agents, diminuant ainsi la consommation de la bande passante, une ressource très convoitée dans les réseaux sans fil.
- La machine cliente n'est plus constamment en attente de résultats intermédiaires et peut se déconnecter puis venir récupérer les résultats plus tard. En réduisant le plus possible les communications distantes aux seuls transferts d'agents mobiles, on diminue considérablement les périodes de connexion entre deux sites, où le maintien du lien de communication peut s'avérer difficile dans des réseaux à large échelle ou sans fil.
- Un autre avantage notable est la diminution des temps de latence. Dans le contexte des réseaux à large échelle, la mise en place d'applications reparties, nécessitant de fréquentes interactions entre client et serveur, se heurte aux temps de latence propres aux communications réseaux. Il arrive fréquemment que le temps d'attente de la réponse

d'une requête soit plus long que le temps de traitement nécessaire à la réalisation du service. En rapprochant client et serveur dans un même sous-réseau, voire sur un même site, on les place dans un environnement où les temps de réponse des interactions sont limités, ce qui permet de réduire d'autant les temps de latence.

- Une autre motivation des codes mobiles est qu'ils peuvent effectuer des opérations à distance, de manière autonome, sans dépendre de l'état du réseau. En effet, les agents mobiles peuvent permettre des interactions distantes plus robustes sur des réseaux non sûrs (par exemple, défaillances de canaux de communication ou de serveurs). Ils permettent d'envisager de nouveaux mécanismes de tolérance aux fautes, afin d'assurer une fiabilité accrue des exécutions.
- Le code mobile offre une alternative viable à la présence d'une copie de chaque programme sur tous les ordinateurs du réseau, réduisant ainsi de façon significative l'espace total de stockage requis.

Ainsi, ce nouveau paradigme de communication est intéressant dans les situations suivantes, (propres plus particulièrement à l'environnement nomade) :

- La machine cliente ne dispose pas d'assez de ressources pour effectuer une action dans le réseau (processeur, mémoire, batterie, connexion).
- L'action est longue et ne demande pas d'interactions de l'utilisateur, permettant ainsi un mode déconnecté et asynchrone.

3.5. Migration d'activité et mobilité d'agent

Il faut différencier la migration d'activité dans les systèmes répartis conçus pour les réseaux locaux et la mobilité d'un agent dans les réseaux de grande envergure et de stations nomades. Les systèmes d'agents mobiles pour les réseaux de grande envergure utilisent la mobilité pour réduire le coût des communications en amenant le calcul sur les serveurs et pour permettre la déconnexion du client. Le déplacement d'un agent correspond à une migration d'activité en une seule communication, où l'on déplace l'ensemble de l'image d'exécution de l'agent avant de le restaurer. Contrairement à la migration d'activité dans les systèmes répartis modernes, où le transfert du code et des données se fait après démarrage "à la demande" par les mécanismes de pagination.

Une différence majeure tient au fait que les systèmes répartis proposent des solutions de migration d'activité pour la construction d'un système d'exploitation uniforme sur un réseau local en cachant la répartition. Alors que la mobilité des agents constitue un modèle d'exécution répartie pour les réseaux de grande envergure et de stations nomades. Dans le premier cas c'est le système qui décide de la migration des activités, alors que dans le deuxième cas, c'est l'agent mobile lui-même qui décide, selon la stratégie applicative, quand, pourquoi et où se déplacer. Une fois migré sur un site distant, un agent mobile devient une entité autonome, qui peut suivant l'exécution du code, se déplacer sur un autre site, accumuler des résultats, communiquer avec d'autres agents, signaler un évènement, etc., sans qu'une connexion permanente soit maintenue avec le site initial. Alors que la migration d'activité

dans les systèmes répartis, fournit un mécanisme qui a été introduit pour répondre à trois types de problèmes posés dans les réseaux locaux [96] :

- La répartition de charge : il s'agit d'optimiser la ressource de calcul globale en jouant sur la charge des processeurs disponibles. On déplace une activité s'exécutant sur un processeur très chargé vers un processeur moins chargé.
- La tolérance aux pannes : il s'agit de gérer les pannes en déplaçant les activités vers des processeurs de secours pour assurer une disponibilité constante des services.
- Le partage d'information : le partage d'information entre activités réparties est mis en œuvre en déplaçant les activités vers la machine où les données sont en mémoire plutôt que de gérer la réplication des données dans plusieurs mémoires.

Nous estimons que la caractéristique principale d'un agent mobile est de disposer d'une certaine autonomie qui le différencie de la migration de processus.

3.6. Applications

La principale motivation de l'utilisation d'agents mobiles, comme on vient de le voir précédemment, est généralement la minimisation des communications distantes. Il est en général moins coûteux de migrer le code de traitement que les données à traiter, qui peuvent être beaucoup plus volumineuses. C'est le cas de l'application de recherche d'informations.

La recherche d'information sur des bases de données disséminées dans le réseau, mettant en jeu un grand nombre de serveurs et touchant un volume important d'information brute, est l'application la plus simple envisagée par les développeurs. Les modèles de communication actuellement utilisés sont fondés sur le paradigme client-serveur qui est peu adapté à la réalisation d'actions complexes dans ces réseaux d'information. Une application peut instancier un agent dans le réseau d'information qui se déplace vers les serveurs pour obtenir la liste des documents pertinents à une requête, et réaliser une sélection plus fine basée sur une heuristique propre à l'application. L'agent porteur de la requête peut également à distance procéder à des traitements en fonction du profil de l'utilisateur, comme par exemple la traduction des données multimédias vers des formats adaptés au terminal de présentation, et négocier la qualité de service pour une transmission synchrone ultérieure avec l'utilisateur. Dans le cadre des applications de commerce électronique, les agents mobiles peuvent rechercher des informations, faire des achats, négocier, ou effectuer n'importe quelle action pour le compte de leur propriétaire.

Une autre application des agents mobiles concerne le cas de l'informatique nomade, où les ordinateurs portables sont sujets durant leurs déplacements à de fréquentes déconnexions volontaires (à l'initiative de leur utilisateur) ou involontaires (suite à une interruption temporaire de communication dans un réseau sans fil, en arrivant dans une zone non-couverte par exemple). En outre, les réseaux sans fil sont caractérisés par un faible débit, un coût de communication élevé et une faible qualité de service. Ainsi, les agents mobiles peuvent être utilisés pour la programmation d'applications destinées à un environnement d'informatique nomade, en particulier pour surmonter les problèmes posés par la déconnexion des sites. En

effet, un agent mobile créé à l'initiative d'un site mobile peut s'exécuter dans le système même si le site mobile créateur fonctionne en mode déconnecté après la migration de l'agent vers d'autres sites.

Ils peuvent aussi être employés comme agents de surveillance ou d'écoute sur un réseau, pour le compte par exemple, de l'administration de réseaux qui sont traditionnellement fondés sur un modèle client/serveur. Le système d'administration par délégation (*Management by Delegation* (MbD)) [42] est le premier système d'administration fondé sur du code mobile ayant été proposé.

C'est dans ce contexte que s'inscrit notre thèse, puisqu'on s'intéresse à étudier la possibilité de l'utilisation de la technologie des agents mobiles dans les applications de gestion des réseaux mobiles ad hoc, plus particulièrement la gestion de la localisation des objets mobiles qui constitue le problème fondamental dans la gestion de la mobilité dans les environnements mobiles.

3.7. Plate forme d'agent mobile

Depuis l'apparition de la notion d'agent mobile, de nombreuses plates-formes ont été développées afin de faciliter la programmation d'applications structurées en termes d'agents mobiles. Une plate forme d'agents mobiles fournit une interface de programmation et un environnement d'exécution offrant des primitives pour créer, lancer et obtenir les résultats des calculs effectués par les agents.

Un environnement d'exécution d'agents mobiles est constitué d'un ensemble de programmes statiques, appelés places ou agences, s'exécutant sur les sites du système susceptibles d'accueillir des agents. Les places sont des programmes qui fournissent aux agents l'infrastructure de base pour leur exécution et leur permettent de se déplacer (voir figure 33). Les environnements d'exécution d'agents mobiles offrent plusieurs services de base permettant l'exécution d'un agent mobile sur un site. Ces services sont les suivants [96]:

-Création et migration d'un agent mobile,
-Réception de l'agent, identification et activation du code,
-Accès aux ressources locales par les agents mobiles,
-Communication et coopération entre les agents mobiles,
-Traçage (**localisation**) des agents mobiles en cours d'exécution.

Le premier système à proposer ces services fut Téléscript de Général Magic qui est à l'origine du premier système d'agents mobiles, né dans la première moitié des années 90. Il a déclenché l'intérêt des grands opérateurs de télécommunication pour cette nouvelle forme de structuration des communications, et il a depuis donné lieu à de nombreux descendants, extensions du langage Java, les *Aglets* d'IBM, *D'agents* ou *Agents-Tcl* de l'université de Dartmouth, *Voyager* de la société ObjectSpace de Dallas*, Tacoma, Mole, Ara, Java-To-Go, Concordia, Grasshopper, Odyssey,* etc. Ces différents systèmes seront présentés au paragraphe 4.

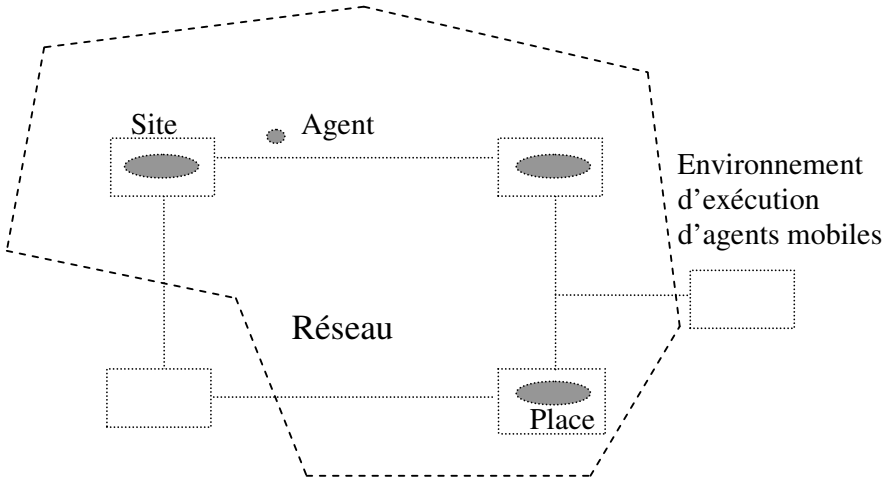


Figure 33 : *Environnement d'exécution d'agents mobiles dans un réseau.*

3.7.1. Création d'un agent

La création d'un agent consiste à créer une instance d'agent sur une place donnée. Elle s'effectue à partir du modèle d'agent. Les données d'un agent sont initialisées lors de sa création. Une fois créé, l'agent peut être actif, c'est-à-dire que l'exécution du code de la tâche associée à l'agent est déclenchée. La création/activation d'un agent peut prendre plusieurs formes :

- Création locale,
- Création et exécution à distance (*remote execution*).
- Création et exécution à la demande (*code on demand*),

On parle de création locale lorsque l'agent est créé et activé sur la place située sur le site ayant émis la requête de création ; l'agent peut alors migrer vers le réseau à partir de ce site.

Lorsque la création est locale et que l'activation a lieu sur un autre site à distance (l'agent est transféré après sa création et avant son activation), on parle de création avec exécution à distance. Dans ce cas, la place sur laquelle l'agent doit être activé est déterminée par l'entité ayant émis la requête de création de l'agent. Un exemple de cette forme est le cas de répartition de charge dans les systèmes répartis.

Dans le cas de la création avec exécution à la demande, une place initialise le transfert du code de l'agent. L'exécution à la demande peut être utile lorsqu'une place désire à un moment donné que l'on lui amène un agent mobile depuis une autre place. Des technologies qui ne peuvent pas être définies comme des agents mobiles, mais qui utilisent cette idée sont ActiveX et Applets Java.

Le tableau suivant résume les trois mécanismes cités :

	Création locale		Création et exécution à distance		Création et exécution à la demande	
	Site A	Site B	Site A	Site B	Site A	Site B
Requête de création	X		X		X	
Création effective	X		X			X
Activation	X			X	X	

Tableau 1 : *Les différents mécanismes de création/activation des agents.*

3.7.2. Exécution d'un agent

Une fois initialisé (crée), l'agent peut commencer son exécution. L'exécution de l'agent consiste en l'exécution du code de la tâche qui lui est associée. Elle a pour effet de modifier les données de l'agent. Plusieurs agents peuvent s'exécuter en parallèle sur une place. L'exécution d'un agent peut conduire à des accès aux ressources du site local et à des interactions avec d'autres agents.

L'accès aux ressources locales d'un site par un agent mobile consiste à effectuer des entrées/sorties, accéder à l'interface utilisateur, au système de gestion de fichiers, aux applications externes et à l'interface du réseau. Les agents ont donc besoin d'accéder au système hôte. Toutefois, les mécanismes d'accès varient d'un système d'exploitation à un autre. Par conséquent, maintenir l'indépendance vis à vis du système d'un environnement d'agents mobiles, tout en autorisant en même temps l'accès aux ressources, est un problème important à surmonter dans les environnements d'exécution d'agents mobiles. Une solution consiste à fournir un accès indirect aux ressources à travers l'utilisation d'agents statiques, fournisseur de services, dont la fonction est alors de réaliser la médiation entre l'agent client et les services disponibles

3.7.3. Structure d'un agent mobile

Un agent mobile est défini par un modèle d'agent. Le modèle d'agent définit l'état d'un agent par un ensemble de paramètres d'une part, et des primitives d'accès à cet état permettant d'initialiser et modifier les paramètres de l'agent, d'autre part. Une instance d'agent mobile comporte trois parties : des données, du code et un contexte d'exécution.

-Les données sont les valeurs des paramètres définis par le modèle d'agent. Parmi les données d'un agent mobile, on trouve son nom, son itinéraire, le programme de la tâche à exécuter, un dossier destiné à recevoir les résultats de l'exécution de ce dernier.

-Le code met en œuvre les primitives d'accès aux valeurs des paramètres.

-Le contexte d'exécution reflète l'état d'exécution courant de l'agent mobile (valeurs des registres, pile d'exécution).

3.7.4. Communications

On identifie trois types de communication dans un système d'agent mobile :

- Communication d'un agent mobile avec un agent statique,
- Communication entre agents mobiles,
- Communication entre un agent et l'environnement client.

L'agent statique représente un service sur la machine serveur (site d'accueil) et constitue un moyen d'interface supplémentaire. La communication de l'agent mobile avec un agent de service offre un moyen pour l'accès aux ressources du site d'accueil.

La communication entre agents mobiles pose essentiellement les problèmes suivants:

- Du fait de la mobilité des agents, les agents impliqués dans une communication ne connaissent pas a priori la **localisation** de leur partenaire.
- Un des agents impliqués dans une opération de communication est susceptible de migrer au cours de cette dernière.

Pour pallier l'absence de localisation fixe des agents mobiles, certains environnements d'exécution d'agents mobiles, proposent une primitive de rendez-vous (*meet*). Dans l'environnement Tacoma, les agents impliqués dans un rendez-vous doivent être situés sur la même place pour que la communication puisse intervenir. Le premier agent exécutant la primitive de rendez-vous attend le second. L'environnement AgentTCL fournit un mécanisme similaire. Un agent X envoie à un autre agent Y une demande de connexion par le biais de la commande *meet*. L'autre agent Y se sert de l'acceptation de l'agent pour obtenir la demande de connexion, soit pour une acceptation ou un rejet. Une acceptation comprend un numéro de port TCP/IP. Les deux agents peuvent alors échanger des messages sur cette connexion.

La communication entre l'environnement client (le propriétaire de l'agent) et l'agent qui a été délégué pour une action est peu abordée. Les systèmes utilisent souvent la communication classique par échange de messages où proposent d'attendre que l'agent revienne dans l'environnement client pour communiquer localement.

Les mécanismes de localisation des agents mobiles, étudiés dans cette thèse, s'intéressent plus particulièrement à ces deux derniers types de communication.

3.8. Caractéristiques d'un agent mobile

3.8.1. Types de mobilité d'un agent

La mobilité ou migration est un mécanisme qui vise à transférer un agent d'un site sur un autre pour qu'il y poursuive son exécution. Il existe différents types de migration d'un agent qui peuvent être classifiés selon deux grandes caractéristiques : la première est la prise de décision de la migration (proactive / réactive) et la seconde est la capacité de déplacement du contexte d'exécution (faible / forte).

- ***Migration proactive / réactive***

Il existe deux grandes classes de migration, la migration proactive ou la migration réactive :

- Avec la migration proactive, c'est l'agent qui initie le déplacement, ceci permettant de garder intégralement leur caractère autonome.
- Avec la migration réactive, c'est le système qui initie les déplacements sans avoir besoin d'une demande explicite de l'agent. La migration réactive intervient généralement lorsque l'on souhaite mettre en place des migrations totalement transparentes aux unités déplacées. Dans ce contexte, chaque unité retrouve, avant et après la migration, le même environnement et n'a absolument pas conscience de son déplacement. (voir paragraphe 3.5 pour le cas de la migration d'activité dans les systèmes répartis).

- ***Migration faible / forte***

D'un autre côté, deux types de mécanismes de migration ont été proposés dans les environnements d'agents mobiles : la migration faible et la migration forte.

- La migration faible d'un agent, consiste à faire migrer uniquement son code et ses données; un nouveau contexte est bâti éventuellement à l'aide de données initiales.
- Pour une migration forte, le contexte d'exécution d'un agent est déplacé en même temps que son code et ses données, vers la place de destination ; l'exécution de l'agent reprendra sur le site distant à l'instruction suivant immédiatement la primitive de migration dans le code.
  Les environnements d'agents mobiles fournissant la migration forte nécessitent l'accès au contexte d'exécution de l'agent ainsi qu'un format pour le transfert de l'agent, ceci en définissant un modèle global d'agent et des méthodes pour la capture de son état interne et le transfert d'une image de cet état. Seulement quelques langages permettent d'externaliser l'état d'un agent à un niveau d'abstraction aussi élevé. Ces difficultés ont poussé la plupart des environnements d'agents mobiles à fournir un mécanisme de migration faible.

3.8.2. Types d'agents mobiles

Ils existent deux types d'agents mobiles. Les agents légers et les agents lourds :

- Les agents légers sont des agents qui exécutent des tâches simples, qui effectuent de courtes phases de calcul local et ils vont donc migrer fréquemment et rapidement. Il s'agit d'agents de petite taille dans le sens où leur code exécutable est réduit le plus possible et les informations qu'ils véhiculent sont peu volumineuses. Cette petite taille va leur donner la faculté d'un déplacement très bref dû à un temps de transmission très court, grâce à leur faible coût en bande passante. Cette caractéristique est fort appréciable dans les environnements dynamiques visés où le lien entre deux sites a une

durée de vie limitée et souvent réalisé par une connexion sans fil peu généreuse en bande passante.

L'intérêt principal des agents légers est qu'ils vont se déplacer bien plus vite que ne peuvent le faire les sites supports (nœuds mobiles). En étant plus rapides à bouger que les sites à renouveler leurs voisinages, ces agents légers pourront migrer sur n'importe quel élément accessible avant que celui-ci ne disparaisse, permettant ainsi aux agents légers de ne pas réellement se soucier du problème liés aux fréquents déplacements des sites. Notons que cette rapidité de mouvement n'empêche absolument pas les phases de dialogue mais que la politique générale de migration va privilégier les déplacements plus que les longs échanges. Pour pallier à ces brèves phases d'échange, on peut augmenter le nombre d'agents légers provenant d'une même application.

Le domaine d'application des agents légers s'inscrit principalement dans la réalisation des services de base du système, qui demande une forte indépendance par rapport aux déplacements des sites et une grande réactivité aux changements de l'environnement. Pour ce faire, ces agents seront nombreux et vont généralement utiliser la migration libre en multipliant de brèves phases de coopération, directe ou indirecte, avec des partenaires pris au hasard. Leur tâche principale est généralement inspirée des algorithmes de types fourmis, car ils correspondent parfaitement à la philosophie d'indépendance vis à vis de l'environnement et des autres composants de l'application.

- A l'opposé des agents légers, les agents lourds réalisent des tâches relativement complexes, imposant de longues phases de traitements locaux et en conséquence, ils effectuent de rares déplacements qui sont relativement lents. Ces agents sont dits lourds car la taille du code exécutable ainsi que celle des données transportées seront beaucoup plus volumineuses que celles des agents légers. Cette grande taille va imposer de lents déplacements à cause de leur grande consommation en bande passante nécessitant un long temps de transmission. Cette propriété va poser un problème dans les environnements dynamiques qui ne peuvent pas garantir la continuité des liens de communication et ne peuvent proposer qu'une faible bande passante.

  Ces agents lourds sont généralement utilisés lors de la conception d'applications qui nécessitent un savoir faire particulier, bien plus complexe que celui des agents légers. Dans ce type d'applications, le développeur connaît généralement les autres services qu'il souhaite utiliser et va donc choisir de mettre en place la migration ciblée pour établir les rencontres nécessaires à la mise en œuvre de la coopération.

  C'est là où ce situe le principal problème d'un agent lourd, car les longues phases de traitements peuvent le rendre beaucoup moins mobile que les sites supports et vont nécessiter des adaptations à chaque fin de calcul, pour appréhender les modifications de l'environnement et pouvoir effectuer la prochaine rencontre. De plus, vu le coût important d'une migration pour un agent lourd, il faudra minimiser le nombre de déplacements qui lui permettront d'atteindre son partenaire et par conséquent, il ne

pourra pas naviguer au hasard de sites en sites mais il devra orienter ses mouvements afin d'optimiser le coût général de cette phase de recherche.

Cubat présente dans ses travaux [27] une architecture basée sur une combinaison des deux types d'agents. Comme montré sur la figure 34, les agents légers qui interviennent au niveau de la couche système, constituent une couche intermédiaire plus stable pour les agents lourds utilisés dans la couche application.

En effet comme déjà vu, les agents lourds, de par leur coût élevé de migration, vont devoir minimiser le plus possible le nombre de déplacements, et pour pouvoir faire ces choix optimisant leurs migrations, ils ont besoin d'avoir une représentation, logicielle et matérielle, qui leur garantissent une probabilité élevée de trouver à l'endroit indiqué les composants recherchés. Cette représentation peut être offerte par la couche d'agents légers à laquelle vont se référencer les agents lourds pour être aiguillés sur leurs choix de migration, aussi bien sur la localisation de leurs partenaires que sur les itinéraires à suivre.

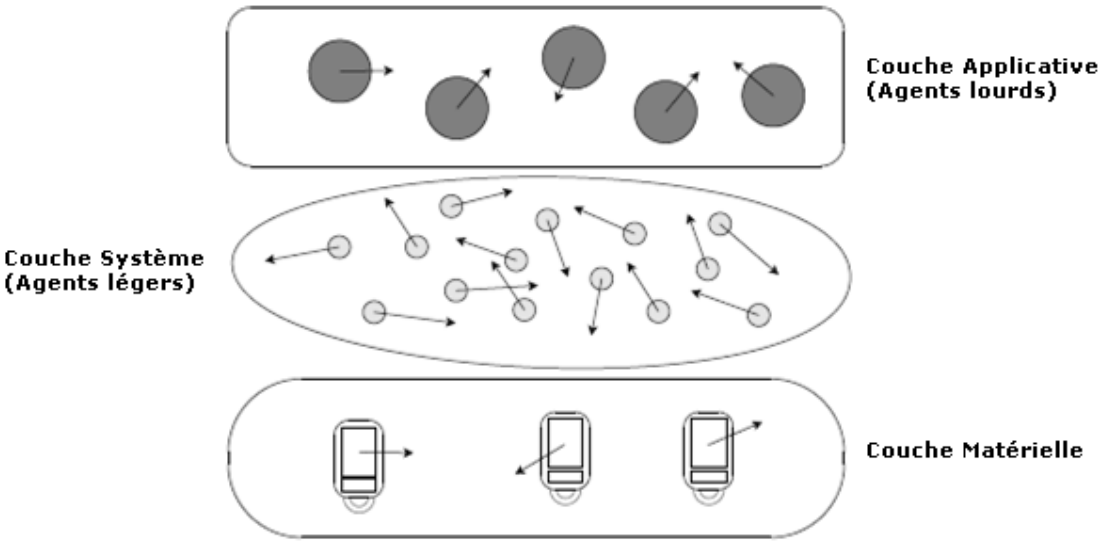


Figure 34 : *Les couches d'agents.*

3.9. Problématique des agents mobiles

Le concept d'agent mobile présente toutefois certains inconvénients liés principalement à la qualité de services offerte par les environnements d'exécution d'agents mobiles et au problème de l'interopérabilité des agents sur les différentes plates formes utilisées.

3.9.1. Qualités de service dans les environnements d'exécution d'agents mobiles

Comme dans tout environnement distribué, il est important de garantir des propriétés de qualité de service dans les environnements d'agents mobiles, comme la sécurité ou la tolérance aux fautes, du fait de la nature même du modèle d'exécution des agents mobiles et du fait des interactions des agents mobiles avec plusieurs systèmes et ressources.

Nous présentons successivement dans la suite de ce paragraphe les mécanismes mis en œuvre par les environnements d'exécution d'agents mobiles pour garantir les propriétés de sécurité et de tolérance aux fautes.

3.9.1.1. Sécurité et protection

Pour ce qui est des agents mobiles, d'un point de vue de la sécurité, nous sommes face à un problème qui n'est pas encore intégralement résolu. C'est d'ailleurs le principal argument avancé pour expliquer la faible utilisation de ce paradigme. En effet, les agents mobiles représentent un nouveau champ d'investigation pour le domaine de recherche en sécurité, d'une part dans la protection des sites vis-à-vis des agents malveillants et d'autre part dans la protection des agents vis-à-vis des sites malveillants.

- **Protection des sites**

Cela consiste à protéger les sites contre la sur-utilisation des ressources et les actions malveillantes (intentionnelles ou non) des agents accueillis. Au cours de son exécution, un agent mobile peut avoir accès aux ressources du site sur lequel il est situé. L'agent mobile peut donc profiter des accès aux ressources locales pour propager des virus, des Worms ou des chevaux de Troie. Il peut masquer sa véritable identité en usurpant l'identité d'un autre et mettre en place des attaques de dénis de service.

Le problème de la protection des sites est aujourd'hui bien maîtrisé. En effet, plusieurs solutions permettent maintenant de se prémunir d'éventuelles attaques et voici les méthodes les plus connues [96]:

- ***Bac à Sable :*** La technique du bac à sable consiste à exécuter un agent à l'intérieur d'un environnement restreint, en interdisant l'accès au système de fichiers par exemple. Ainsi, un site peut exécuter un agent douteux dans le bac à sable sans trop se soucier des problèmes de sécurité. Cette approche peut facilement se mettre en place en utilisant des interpréteurs de code dont leurs possibilités sont limitées. Pour illustrer cette technique nous pouvons citer les applets Java exécutés à l'intérieur d'un navigateur Web.
- ***Signature de code :*** La signature de code intervient lors de la création d'un agent, son créateur le signant numériquement afin qu'il puisse s'identifier durant ses déplacements. Cette technique permet d'obtenir une authentification de haut niveau pour les sites. Les applets Java adoptent désormais ce mode de fonctionnement. Ainsi si une applet est signée, elle est considérée comme un code de confiance et peut accéder à toutes les fonctionnalités de Java. Elle sera placée dans un bac à sable dans le cas contraire.
- ***Contrôle d'accès :*** Pour améliorer les deux précédentes techniques, une politique de contrôle d'accès plus complexe est mise en place. Elle consiste en un raffinement d'une politique de bac à sable générale vers une politique spécifique à chaque application ou classe d'agents. En fonction des agents, le site pourra autoriser l'accès à un ensemble précis de fonctionnalités. Le contrôle d'accès permet de mixer les deux

premières techniques en offrant aux agents signés plus de fonctionnalité qu'un simple bac à sable sans pour autant accéder à toutes les fonctionnalités.

- ***Vérification du code :*** La vérification de code permet d'obtenir une garantie sur la sémantique d'un code à travers l'analyse de sa structure, ou de son comportement pour un agent, en fonction d'une politique de sécurité donnée. Les bacs à sable font déjà des vérifications rudimentaires en cours d'exécution, principalement pour garantir le typage des opérandes des instructions, mais cela est fortement coûteux. Une autre approche est de vérifier automatiquement le code, avant son lancement, en s'appuyant sur une preuve de conformité. Lors de la mise en route de l'agent, son créateur fournit un ensemble de preuves intégrées qui est transporté par l'agent. Ces preuves garantissant le comportement de l'agent en fonction de critères de sécurité des sites à visiter. Lorsque l'agent commence une nouvelle visite, le site récupère la preuve lui correspondant et vérifie si elle correspond à sa politique de sécurité. Le site choisit alors d'exécuter ou non l'agent. Cette technique se base pour l'instant sur des propriétés de typage et la preuve est fournie par le compilateur.

- **Protection des agents**

Cela devrait permettre d'avoir l'assurance que ni le code de l'agent ni ses données ne seront modifiés (mutation d'agent), qui peut se traduire par un changement de comportement de l'agent et qui peut devenir malveillant. Le problème du transport d'un agent se résout par des techniques de signature digitale assurant qu'un message transmis n'a pas été modifié. Par contre, le problème de la protection de l'agent vis-à-vis du système où il s'exécute est un problème plus complexe que le précédent, puisque tout ce qui est accessible à l'agent l'est aussi au système d'exploitation dans lequel il s'exécute, et n'a pas reçu à ce jour de solutions satisfaisantes et reste encore aujourd'hui un champ de recherche ouvert.

Pour comprendre ce que risque un agent lors de son exécution sur un site malveillant, nous pouvons rappeler les éléments transportés pouvant être cible d'attaque :

- Le code : ensemble des instructions composant la tâche de l'agent.
- Les données statiques : données ne changeant pas durant les déplacements (la signature par exemple)
- Les données collectées : ensemble des résultats obtenus au cours des déplacements réalisés par l'agent depuis son lancement.
- L'état courant : ensemble de données servant à l'exécution courante de l'agent.

La sécurité des agents mobiles consiste alors à garantir les critères de confidentialité et d'intégrité de l'ensemble de ces éléments. Du point de vue des données, il est évident qu'un agent ne souhaite pas divulguer des informations critiques à n'importe quel site. Par exemple, un site malveillant pourrait récupérer la signature d'un code et l'utiliser pour créer un nouvel agent afin de s'introduire dans des environnements auxquels il n'a normalement pas accès. Pour le code, un agent transporte un savoir-faire propre à son concepteur qui pourrait tomber aux mains de ses concurrents.

Nous pouvons classifier trois grandes catégories d'attaques que les sites sont capables de réaliser : l'inspection, la modification et le rejeu [96].

- L'inspection consiste à examiner le contenu de l'agent, ou le flot d'exécution afin de récupérer des informations critiques transportées par l'agent.
- La modification se réalise en remplaçant certains éléments de l'agent dans le but de conduire une attaque. En remplaçant le code, l'agent effectuera des opérations malveillantes sur les futurs sites à visiter.
- Le rejeu s'obtient en clonant l'agent puis en exécutant le clone dans plusieurs configurations pour retrouver le savoir de l'agent.

Différentes techniques sont en cours d'étude pour garantir aux agents l'accès à des sites dans lesquels ils peuvent avoir toute confiance ou pour détecter les agents corrompus. Cependant, aucune d'elles n'apporte un niveau de sécurité suffisant comme celui proposé pour protéger les sites.

Ce que nous pouvons dire pour conclure sur les problèmes de sécurité liés à l'utilisation des agents mobiles, c'est qu'il s'agit encore d'un champ d'investigation complet. Bien que la protection des sites soit à présent maîtrisée, la protection des agents n'est, pour l'instant, toujours pas satisfaisante.

3.9.1.2. Tolérance aux fautes

L'agent de par son modèle d'exécution qui implique une interaction avec plusieurs sites est enclin à disparaître à cause de la défaillance d'un site ou de la déconnexion soudaine et imprévue d'un site sur lequel il s'exécute. Pour certains types d'applications, il est essentiel que les environnements d'exécution d'agents mobiles offrent des mécanismes de tolérance aux fautes. En effet, un agent peut disparaître après avoir visité plusieurs sites. Si aucune précaution n'est prise, les résultats de son exécution sur ces sites peuvent être perdus. Il est alors important de détecter la disparition d'un agent pour en informer la place qui l'a lancé.

Peu d'environnements d'exécution d'agents mobiles existants fournissent des mécanismes de tolérance aux fautes. Les environnements Tacoma et Ara (paragraphe 4) offrent des mécanismes visant à faire face à la défaillance d'un site (et donc de la place qui s'y exécute). Plus précisément, il s'agit d'assurer la reprise des agents mobiles qui s'exécutaient sur la place défaillante. Pour ce faire, un mécanisme de point de reprise est mis en place permettant la sauvegarde de l'état interne d'un agent mobile. Les points de reprise sont conservés sur disque, ce support étant supposé fiable. Ils peuvent ainsi être utilisés ultérieurement pour restaurer l'agent en cas de défaillance. Cependant, les points de reprise et la restauration doivent être utilisés avec soin, par exemple, restaurer un agent qui est toujours actif sur un système distant pourrait produire plusieurs copies du même agent, ce qui requiert un traitement explicite.

3.9.2. Interopérabilité

L'interopérabilité a pour objectif de permettre à plusieurs systèmes logiciels, de même nature ou hétérogènes, de communiquer et de coopérer afin de minimiser les coûts

d'intégration de ces systèmes [36]. En tant que système logiciel, la technologie d'agent mobile est également confrontée au problème de l'interopérabilité (ou portabilité du code) sur les différentes plates-formes d'accueil, d'autant plus que leur vocation à interagir et à coopérer de façon autonome est l'essence même de son existence [74].

L'interopérabilité repose sur des normes techniques (à la base des standards), qui définissent des exigences accompagnées de recommandations permettant à deux systèmes qui satisfont aux exigences de dialoguer et d'évoluer librement, tant qu'ils respectent la norme définissant leurs interfaces. Cette normalisation devrait permettre à terme de rendre compatibles les différents systèmes. On peut trouver à l'heure actuelle deux normes principales [13]: il s'agit de la norme FIPA (*Foundation for Intelligent Physical Agents*) et de la norme MASIF (*Mobile Agent System Interoperability Facility*).

- **MASIF :** la norme MASIF a été spécifiée par l'OMG (*Object Management Group*) qui se préoccupe généralement de l'hétérogénéité entre les systèmes, comme dans le cas de CORBA. Dans cette optique, le but dans la norme MASIF, est de décrire les notions élémentaires permettant l'échange des agents entre différentes plates-formes. Pour ce faire, elle standardise la manière de gérer le code des agents, leur identification, la migration et l'adressage local.
- **FIPA :** en revanche, la communauté à l'origine de FIPA étant celle des systèmes multi-agents, plus proche de l'intelligence artificielle, elle va se situer à un niveau plus élevé, c'est-à-dire le niveau applicatif en décrivant les éléments nécessaires à la réalisation d'une application et principalement en détaillant la communication entre les agents. Le but est de décrire un ACL (*Agents Communication Language*), les ontologies et les protocoles de négociation permettant ainsi de définir parfaitement les interactions entre les agents.

La différence entre MASIF et FIPA réside dans le fait que MASIF vise à permettre aux agents mobiles de migrer entre les systèmes d'agents du même profil par l'intermédiaire des interfaces normalisées de CORBA, contrairement à FIPA qui permet l'interopérabilité d'agents intelligents par l'intermédiaire de la communication normalisée des agents et des langages. Ces deux normes tendent plus vers la complémentarité que vers la divergence. C'est déjà le cas de FIPA qui a inscrit dans son planning l'intégration des règles de MASIF sur la gestion de la migration [13].

3.9.3. Formalisme de description

Les agents, et en particulier les agents mobiles, constituent un concept utilisé dans différents domaines et dont les contours sémantiques sont assez larges. Il est donc difficile d'en trouver une formalisation unique.

Quelques travaux sur les formalismes de description des agents mobiles ont été proposés [27], tels que *Mobile Unity* basé sur les systèmes de transitions. Des extensions du λ-calcul ont été proposées, dans le cadre du projet SARDES, pour traduire la mobilité des agents. De même, une architecture de communication entre agents mobiles est formalisée en utilisant une

extension du λ-calcul. Dans le cas de la logique Klaim, les propriétés des agents sont exprimées dynamiquement en fonction de l'évolution de leur localisation.

Les techniques de conception pour les applications orientées objet, telles que UML, ont été adaptées au domaine des agents afin de formaliser les interactions, comme dans AUML. Cependant, ces techniques viennent du domaine des systèmes multi-agents qui se concentrent plus sur l'expression des relations inter-agents. L'expression de la mobilité du point de vue des systèmes distribués n'est pas décrite [27].

4. Etat de l'art des travaux sur le concept d'agent mobile

Beaucoup de travaux de recherche ont été effectués sur la technologie des agents mobiles, pour à la fois approfondir cette nouvelle forme d'architecture des réseaux et étudier son impact sur l'existant en matière de communication et de services.

Deux travaux sont à l'origine de l'utilisation du code mobile pour la conception des applications réparties [96] :

– Premièrement, le travail sur les programmes *Worm* en 82 de Shoch et Hupp qui se posent le problème du placement dynamique de code sur un réseau de station de travail. L'objectif du système mis en œuvre pour supporter ce type de comportement est d'exploiter les machines oisives d'un réseau local pour réaliser du calcul réparti. Shoch et Hupp ont réalisé une expérience originale sur les programmes se répliquant d'une manière automatique dans un réseau de machines. Un programme *Worm* est composé d'un ensemble de segments s'exécutant chacun sur une machine distincte. Chaque segment est une réplique du programme original mais dispose d'une identité unique durant l'exécution. Le programme prend en charge le placement des segments dans le réseau en fonction de la charge des machines. Les communications entre les segments sont réalisées classiquement par échange de messages. Le système utilise en plus la diffusion de groupe pour maintenir dans chaque segment la liste des autres segments du programme. La procédure de création de segment se déroule en deux phases : le choix de la machine oisive et le clonage du programme. Le programme peut diffuser une requête demandant aux machines oisives de s'identifier et faire son choix ou bien tester une machine particulière. Puis, il se répand sur la machine choisie en créant un segment distant.

– Deuxièmement, le travail sur les messages actifs de Vittal en 81 qui a donné lieu au modèle *Enabled Mail* de Borenstein et au modèle des *Messengers* de Thsudin :

– Borenstein et Rose proposent un modèle d'interaction appelé *Enabled Mail* qui utilise le courrier électronique comme infrastructure de communication. Ces travaux sont basés sur le principe des messages actifs de Vittal. Les messages peuvent contenir du code qui peut être exécuté à n'importe quelle phase du processus de traitement d'un courrier électronique. Le code peut être exécuté soit par le système de transport lorsque le courrier est envoyé par l'application de rédaction au système de messagerie local ou reçu par le système de messagerie distant, soit lorsque le courrier est lu par l'application de consultation. Le problème de la sécurité est solutionné par l'utilisation de l'interpréteur *Safe-Tcl*. Les opérations dangereuses d'accès au système ont été supprimées ou remplacées par des

opérations contrôlées en limitant l'accès à certaines ressources ou en demandant une authentification. Le transport du code dans un courrier est possible grâce à l'extension du standard MIME (*Multipurpose Internet Mail Extension*) via les types *application/Safe-Tcl* et *multipart/enabled-mail* permettant d'encapsuler le code dans des messages multimédia. *Enabled Mail* propose donc un mécanisme de code mobile par l'intermédiaire de *SafeTcl* et MIME, mais n'impose aucune synchronisation avec l'initiateur et permet lors de l'exécution du code d'envoyer de nouveaux courriers contenant eux même du code. Un tel mécanisme permet de mettre en œuvre une exécution répartie dans un réseau de grande envergure qui est proche du paradigme des agents mobiles.

– L'objectif de Tshudin est de proposer une nouvelle forme de structuration des communications pour la configuration dynamique des protocoles. Ces travaux sont basés sur le principe des messages actifs de Vittal et des programmes *Worm*. Généralement, les processus coordonnent leur exécution en s'échangeant des messages qui sont interprétés selon un ensemble préétabli de protocoles installés sur chaque machine intervenant dans la communication. L'idée de Tshudin est d'échanger des programmes qui contiennent l'ensemble de la logique du protocole. Le code complet du protocole est confiné dans le premier message de l'initiateur. Le site qui initie la communication envoie à une cible un message contenant un programme. Cet élément appelé *Messenger* est automatiquement exécuté sur le site cible et peut communiquer avec les autres sites en émettant lui même des *Messengers*. Il devient ainsi possible pour une application d'explorer un sous ensemble des sites d'un réseau selon sa logique de communication sans avoir à installer préalablement du code spécifique sur chaque site.

– La notion même d'agent mobile dans les réseaux est apparue en 94 avec le système *Telescript* [96]. C'est un système d'agents mobiles développé par la société américaine *General Magic* proposant un système fermé pour le commerce électronique. *Telescript* est un langage interprété proche de LISP intégrant la notion d'objets. Il propose un environnement utilisant différent concepts. Les places sont constituées par les zones de services et les stations clientes du réseau. Les agents sont des entités capables de migrer leur exécution de place en place par l'instruction *go*, de rencontrer d'autres agents sur la même place par l'instruction *meet*, et de communiquer avec d'autres agents sur d'autres places par des connections permettant l'envoi de messages. Les connections n'intègrent aucun mécanisme de localisation global.

– Telescript a été suivi par de nombreux autres systèmes tels Obliq [96] qui est un langage interprété, basé sur les objets et non typé. Les objets Obliq sont locaux aux sites mais il est possible de déplacer des traitements d'un site vers un autre, autrement dit, il prend en charge la mobilité forte mais ne permet pas de gérer les aspects de sécurité de façon satisfaisante.

D'autres travaux proposent des systèmes d'agents mobiles originaux. Nous en présentons quelques uns :

–TACOMA (*Tromso And COrnell Moving Agents*) [27] est un projet des universités de Tromso et Cornell. Il a comme objectif de fournir un support dans les systèmes d'exploitation pour la programmation d'agents mobiles. Le modèle ne fait pas de distinction entre un client,

un serveur et un agent. Il propose simplement la notion d'agent et des mécanismes de migration et de communication. Potentiellement tous les agents peuvent se déplacer dans le réseau de machines. C'est au niveau applicatif que l'on peut instancier un agent avec un comportement de client mobile, de serveur statique ou réaliser un monde d'agents intelligents. Un agent est un processus exécutant un script *Tcl* qui est capable de se déplacer volontairement sur une autre machine. TACOMA propose un mécanisme pour la communication fondé sur le concept du rendez-vous par la commande *meet*. La migration dans TACOMA est réalisée par un rendez-vous avec un agent distant spécial nommé *ag_tcl* dont la seule fonction est d'instancier un nouvel agent à partir d'un *Folder CODE* qui contient le code *Tcl* de réactivation de l'agent.

– Les AGLETS (AGile appLETS) [27] sont des composants développés par une équipe de chercheurs du laboratoire de recherche d'IBM à Tokyo au début de 1995, dans le but de fournir une plateforme uniforme pour les agents mobiles dans un environnement hétérogène, tel que celui de l'Internet. Ce sont des objets Java mobiles qui réagissent comme des agents mobiles. Les principaux éléments de cette plateforme sont :

• un *Aglet* est un objet *Java* comportant un flot d'exécution et disposant de méthodes pour le déplacement, le clonage, le rapatriement, la désactivation sur support persistant, l'activation depuis un support persistant et la destruction.

• un *contexte* correspond à l'environnement d'exécution des Aglets et permet de mettre en œuvre une plate-forme proposant un ensemble de services. Il fournit des mécanismes de sécurité par l'intermédiaire d'un gérant de la sécurité qui est contacté à chaque opération dangereuse, par exemple lors de l'arrivé d'un aglet dans le contexte.

• un *proxy* qui sert d'intermédiaire pour la manipulation des Aglets, les protégeant ainsi de tout accès direct.

• un *message* est un objet échangé entre les Aglets via un proxy.

La sécurité repose sur trois niveaux : la sécurité de *Java*, la sécurité des *Aglets* avec la notion de contexte et la sécurité applicative grâce aux mécanismes d'événement inclus dans les *Aglets*. La migration d'un Aglet est déclenchée en invoquant la méthode *dispatch* de la classe Aglet qui réalise la sérialisation de l'objet, assure le transport et déclenche la réactivation de l'objet dans le contexte cible. La sérialisation de l'Aglet est une procédure qui réalise l'agrégation de tous les objets utilisés par l'Aglet dans un message qui, lorsqu'il est reçu, permet de reconstruire le graphe des objets de l'Aglet. La communication avec un *Aglet* se fait par échange de message classique. Le système des *Aglets* ne dispose en outre d'aucune fonctionnalité de localisation, tous les appels sont explicites au moyen d'un proxy fourni par le contexte pour les *Aglets* locaux et par l'intermédiaire de la désignation par URL pour les *Aglets* distantes.

Le système des Aglets offre ainsi une plate-forme distribuée d'objets actifs mobiles pouvant s'échanger des messages dans un environnement sécurisé. Elle n'offre pas de mécanisme de localisation et la migration est réalisée sans transfert implicite de l'état

d'exécution (code et données seuls). Notons enfin, que les Aglets utilisent depuis 2003, le langage KQML et respectent la norme MASIF émulée sans CORBA.

– *Agent-Tcl* [96] est un système d'agents transportable développé à *Dartmouth College*. Il définit un agent transportable comme un programme identifié qui peut se déplacer de machine en machine dans un réseau hétérogène en choisissant le lieu et le moment de ses déplacements. Le système *Agent-Tcl* est composé de deux programmes : un serveur qui s'exécute sur chaque site du réseau et une version modifiée de l'interpréteur *Tcl* incluant l'extension des agents. Le serveur prend en charge la gestion des agents. Il reçoit puis authentifie et démarre les agents se présentant sur le site avec l'interpréteur, il stocke les messages, fournit un service de désignation local et maintient une table d'information sur les agents présents sur le site. L'interpréteur fournit un mécanisme de sauvegarde et de restauration de l'état d'une exécution d'un script *Tcl*. Il propose des commandes pour la migration, la communication et la création d'agent. Un agent est un script *Tcl* qui s'exécute dans cet interpréteur.

L'agent exécute la commande *agent_begin* pour s'enregistrer auprès du serveur et obtient un identificateur unique. La commande *agent_jump* permet de migrer l'agent sur une autre machine du réseau. Le système capture l'état d'exécution puis l'envoie au serveur de la machine cible qui restaure l'agent avec l'interpréteur. Chaque fois qu'un agent est envoyé vers un serveur (création ou migration) il est crypté et signé digitalement. Le serveur utilise cette signature pour authentifier l'agent afin d'éviter les modifications durant les transferts. Le système fournit un service de messages classique proposant la communication asynchrone avec les commandes *agent_send* et *agent_receive* . Il propose également l'établissement de rendez-vous synchrone entre deux agents avec les commandes *agent_meet* et *agent_accept* permettant de réaliser ensuite des communications directes sur une connexion identifiée (TCP), plutôt que d'attendre des messages dans un buffer maintenu par le serveur. La localisation des agents est réalisée par le service de désignation qui implante une table des agents mise à jour automatiquement en fonction des migrations. Aucun détail n'est fourni sur le protocole utilisé pour mettre à jour cette table lors des migrations.

Le service de désignation n'est plus global, il fournit seulement la liste des agents s'exécutant sur la machine courante. Un agent qui veut être localisé informe une machine connue à l'avance des migrations qu'il effectue durant son exécution. Ainsi, tout agent qui désire communiquer avec lui s'adresse d'abord à cette machine particulière qui lui retourne la machine où l'agent s'exécute.

– JADE (*Java Agent DEvelopment Framework*) [27] est un projet développé par le groupe *CSELT Telecom Italia* avec la coopération de l'université de Parme (Italie). Son but est de simplifier le développement des systèmes multi agents (SMA) tout en fournissant un ensemble complet de services et d'agents conformes aux spécifications FIPA. La plate-forme d'agents peut être répartie sur plusieurs serveurs. Une seule application Java, et donc une seule machine virtuelle de Java (JVM), est exécutée sur chaque serveur.

Plusieurs autres travaux sont présentés dans la littérature [27,96]. Nous citons le système *Tabriz* de *General Magic* qui est une ouverture du système *Telescript* sur le *World-Wide Web*

avec le langage *Java*. Le système *Mole* de l'Université de Stuttgart, JAE de l'Université de Aachen, Concordia de Mitsubishi Horizon System Lab qui sont des travaux proches des *Aglets* d'IBM, toujours avec le langage Java, et le système ARA de l'Université de Kaiserslautern qui propose une architecture pour supporter plusieurs langages de programmation d'agents mobiles.

5. Conclusion

Le critère de performance des agents mobiles sera très utile dès que les applications vont comporter des communications intensives. Grâce à leurs interactions locales, les agents permettent de ne pas subir les ralentissements dus aux bandes passantes limitées et aux temps de latence des réseaux, surtout sur Internet. De plus, en s'exécutant sur les machines serveurs, généralement plus performantes que les clients, les phases de traitement seront plus rapides. D'un point de vue conception, les agents mobiles permettent de décrire des comportements plus difficiles à caractériser avec le client/serveur. Ainsi, l'exploration d'un réseau, la représentation d'un utilisateur, le support des ruptures de communication ou encore l'adaptation à l'environnement sont naturellement exprimables grâce aux agents mobiles.

A travers les différentes caractéristiques des agents mobiles que nous venons de présenter, leurs utilisation semble très prometteuse dans la conception d'applications de gestion pour les réseaux mobiles ad hoc. Plus particulièrement, les agents mobiles de types légers semblent plus adaptés aux caractéristiques des réseaux ad hoc.

Cependant, nous pouvons dire que les agents mobiles ne représentent pas le paradigme idéal pour tous les types d'applications réparties, mais simplement une possibilité nouvelle pour la construction des applications. Selon les besoins de performance, de conception, de développement et de sécurité, les agents mobiles pourront apporter une alternative intéressante au classique client/serveur dans certains types de configuration. De plus, nous pouvons dire que le problème le plus important empêchant l'adoption actuelle des agents mobiles est sans nul doute la sécurité. En effet, même si la protection des sites est quasiment assurée, celle des agents reste un réel problème qui n'a pas de solution définitive.

Chapitre 4

Problème de localisation dans les réseaux informatiques

1. Introduction

Avec la multiplication des objets communicants et le développement des communications et des réseaux sans fil, la fonction de localisation est devenue une composante majeure des futurs services informatiques [19]. Elle consiste à trouver la position d'un objet (logiciel ou matériel), généralement mobile, situé dans un environnement mobile ou statique. L'utilisation de mécanismes de localisation dans les réseaux informatiques est essentielle à la fois pour les protocoles de communication (généralement, pendant la première étape du routage d'information lors de la procédure de découverte de route, et plus particulièrement pour le routage géographique), que pour certaines applications tel que le suivi de véhicules. En effet, la fonction de localisation permet d'assurer la communication entre différents objets dans un environnement mobile, malgré le changement de leurs positions.

La gestion de la localisation met en œuvre deux opérations de base: la recherche et la mise à jour. L'opération de recherche permet de localiser un objet lorsqu'il y a un besoin de le contacter. L'opération de mise à jour permet d'informer les nœuds du réseau de la nouvelle position d'un objet mobile lorsqu'il se déplace dans le réseau. Ainsi, il est important d'établir [20]:

- Quels sont les nœuds du réseau qui auront la responsabilité d'héberger les informations de localisation ? Cela pourrait être quelques nœuds spécifiques seulement lorsqu'il s'agit d'une architecture centralisée, ou bien tous les nœuds du réseau dans le cas d'une architecture distribuée.
- Doit-on conserver les informations de localisation de tous les objets mobiles du réseau (structure non hiérarchique), ou de certains objets spécifiques (structure hiérarchique) qui serviront comme repères pour le calcul de l'information de localisation des autres objets mobiles.

- Quand doit-on mettre à jour les informations de localisation ? De manière réactive ou proactive ? et à qui seront envoyées les mises à jour ?

Différents contextes ont été présentés dans la littérature où est posé le problème de la localisation d'objets mobiles dans les réseaux informatiques, et de nombreuses techniques de localisation ont été développées et proposées pour les différents types de réseaux et selon la nature de l'objet à localiser. Dans ce qui suit, nous allons présenter une étude de l'état de l'art des mécanismes de localisation des objets mobiles (code ou station) dans les réseaux mobiles ad hoc, après avoir présenté quelques techniques de localisation dans les réseaux filaires et les réseaux mobiles cellulaires.

2. Localisation dans les réseaux filaires

Une approche couramment utilisée dans les réseaux filaires, afin de localiser différents types d'objets (ressources logicielles ou matérielles) est de se baser sur un annuaire [97]. Cet annuaire centralise les informations relatives aux ressources, tels que des fichiers, des services, des agents, etc., et ce dans le but de répondre aux requêtes des clients. Cette approche est aussi bien utilisée dans des réseaux à large échelle comme, par exemple, par UDDI (*Universal Description, Discovery and Integration*) pour la découverte de services dans l'Internet, que dans des réseaux locaux comme, par exemple, par Jini et SLP (*Service Location Protocol*) [34].

Dès l'apparition de la technologie d'agent mobile et ses premières utilisations dans les réseaux filaires, le problème de la localisation de l'agent ou code mobile a été posé. Deux mécanismes de localisation sont principalement utilisés dans ce cas [2] :

2.1. Mécanisme distribué pour la localisation d'agent mobile

Le premier mécanisme crée dynamiquement une chaine de répéteurs ou lien de poursuite permettant de localiser l'agent mobile. Les techniques basées sur les chaines de répéteurs ont été d'abord introduites dans les systèmes distribués tel que DEMOS/MP [84] pour trouver les processus mobiles. Le mécanisme est très simple : à chaque fois que l'agent quitte une station, il laisse une référence (une marque), appelée répéteur, qui pointe sur la prochaine destination (i.e. l'agent mobile sur la station de réception). Ceci crée dynamiquement une chaine de répéteurs, constituée de l'ensemble de stations marquées et qui permet de localiser l'agent mobile ; quand un répéteur (station marquée) reçoit une requête de localisation, il la transmet à la prochaine destination de la chaine, et ainsi de suite jusqu'à atteindre la station qui héberge l'agent mobile.

Un mécanisme de raccourcissement de la chaine, permet la mise à jour de la chaîne dès qu'une communication aura lieu. Quand un message transmis par une station marquée atteint un agent mobile, ce dernier communique sa nouvelle position à l'appelant. En conséquence, toutes les requêtes ultérieures émises par cet appelant ne passeront pas par les stations marquées intermédiaires, permettant ainsi de raccourcir la chaîne. Une illustration du schéma de raccourcissement est montrée dans la figure 35 : un message est envoyé par la source à la dernière position connue de l'agent (hôte B). Comme l'agent n'est plus à cette position, le

message est alors transmis à la station qui a été visitée par la suite par l'agent (Hôte C). Aussi, l'agent s'est déjà déplacé quand le message arrive à C et le message est transmis à la prochaine station visitée (Hôte D) ou l'agent est finalement localisé. Un message de localisation est alors envoyé par l'agent (localisé sur l'hôte D) à la source et le message suivant sera envoyé par la source directement à l'hôte D.

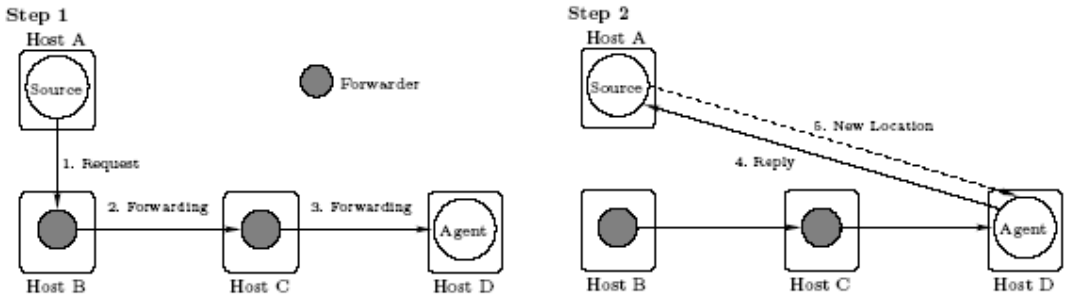


Figure 35: *Un exemple du mécanisme de raccourcissement de la chaîne.*

La description du protocole est comme suit:

1) Après une migration, l'agent mobile laisse un répéteur sur la station courante.
2) Ce répéteur est (re)lié à l'agent mobile sur la prochaine station.
3) Aucune communication ne peut avoir lieu lorsque l'agent est en cours de migration.
4) Lorsqu'une requête de localisation est reçue, le répéteur la transmet vers la prochaine destination de la chaîne (possible que ça soit l'agent mobile).

2.2. Mécanisme centralisé pour la localisation d'agent mobile

Une alternative à l'approche distribuée des répéteurs pour la localisation d'agent mobile est l'utilisation d'un serveur centralisé pour accomplir cette tâche. Le principe de fonctionnement du serveur est simple : le serveur garde une trace de la position de l'agent mobile dans une base de données. A chaque fois que l'agent migre, il informe son serveur de sa nouvelle position. Lorsque la station source veut atteindre son code mobile, il envoi son message à la dernière position connue de l'agent mobile ; si la communication échoue, alors la source envoie une requête de localisation au serveur.

La description du protocole utilisé par la source et l'agent mobile pour communiquer avec le serveur est comme suit :

L'agent mobile :

1) Effectue une migration ;
2) Envoi sa nouvelle positon au serveur.

La source :

1) Emettre un message vers l'agent mobile avec la position connue.
   En cas d'échec aller à l'étape 2 ;
2) Demander au serveur la nouvelle position de l'agent mobile ;
3) Transmettre le message à l'objet mobile avec la position fournie par le serveur.
   En cas d'échec, retourner à l'étape 2.

Le seul problème qui se pose est quand l'agent mobile migre au moment où la source reçoit une réponse du serveur. Cette situation est illustrée dans la figure 36 : le serveur ne donne pas la position correcte de l'agent à la source puisque l'agent a initié une migration en même temps. Comme conséquence, la source devra envoyer une seconde requête de localisation au serveur (événement n°9) avant de pouvoir enfin atteindre l'agent mobile (événement n°11).

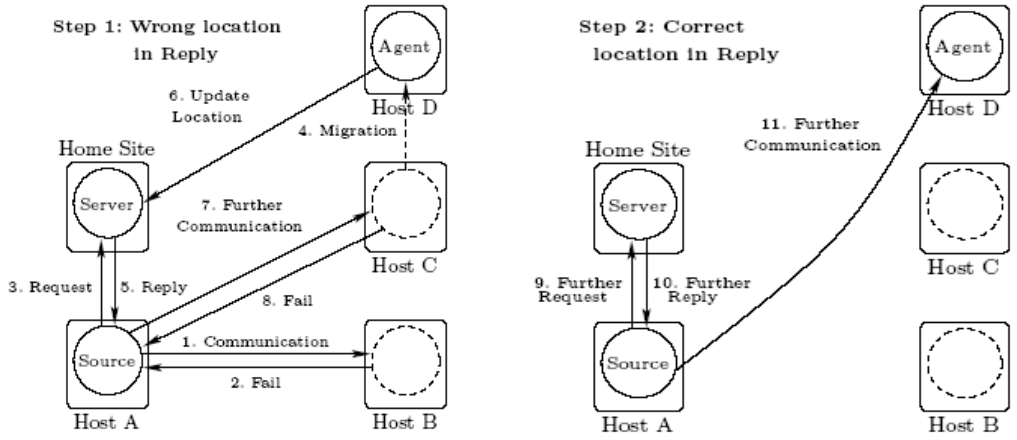


Figure 36: *Un scénario de l'approche centralisée pour la localisation d'agent mobile.*

3. Localisation dans les réseaux mobiles avec infrastructure fixe

Les mécanismes de localisation de ressources dans les réseaux filaires s'adaptent assez bien dans les réseaux cellulaires en raison des infrastructures fixes qui y sont disponibles, notamment, ceux basés sur des annuaires ou serveurs centralisés. La localisation dans les réseaux cellulaires concerne beaucoup plus les stations mobiles ainsi que les stations de bases du réseau.

Castaneda [21] présente une étude sur la localisation dans les réseaux cellulaires, aussi bien de stations de base que de stations mobiles. Les techniques de localisation sont basées sur les mesures radios suivantes: le temps d'arrivée (TOA), la différence de temps d'arrivée (TDOA), l'angle d'arrivée (AOA) ou le niveau de puissance reçue (RSS). Il propose l'algorithme TABLA (Algorithme de Localisation Basé sur le Temps d'Arrivée) pour la localisation de stations de base par une station mobile à partir des temps d'arrivées. Il propose aussi l'algorithme ADABLA (Algorithme de Localisation Basé sur l'Angle et le Retard

d'Arrivée) afin d'estimer la position d'une station mobile par les stations de base à partir des deux mesures, l'angle d'arrivée et le retard d'arrivée.

4. Localisation dans les réseaux ad hoc

Les mécanismes de localisation centralisés ne sont pas directement exploitables dans un réseau ad hoc. En effet, comparé à un réseau filaire, les déconnexions dans un réseau ad hoc sont extrêmement fréquentes. Or, si un annuaire est déconnecté, la localisation de ressources ne peut avoir lieu. Ainsi, un annuaire correspond à un point de cassure unique. Par ailleurs, cet annuaire constitue un goulet d'étranglement. La concentration du trafic provenant de l'ensemble du réseau au niveau de l'annuaire est en outre à l'origine d'une consommation énergétique élevée pour les terminaux proches de ce dernier. Finalement, un réseau ad hoc, par essence même, ne dispose pas d'administrateur réseau gérant le déploiement d'annuaire. Pour ces raisons, les solutions à la localisation de ressources dans un réseau ad hoc se basent sur une approche décentralisée [97].

Dans un réseau ad hoc, les approches préconisées pour localiser les ressources, se basent généralement sur des techniques de diffusion [97]. Cette localisation est soit effectuée par les terminaux mettant à disposition les ressources, qui assurent typiquement la diffusion des informations caractérisant les ressources (approche dite *push*), soit par le client qui interroge les nœuds constituant le réseau (approche dite *pull*) en se basant également sur des techniques de diffusion. Dans le premier cas, garantir la disponibilité des ressources nécessite une forte fréquence de notification. De plus, l'augmentation du nombre de clients et/ou de terminaux mettant à disposition des ressources accroît l'encombrement dans un réseau caractérisé par une bande passante étroite. L'approche *push* est par exemple adoptée par le protocole DEAPspace [97] pour la localisation de ressources dans les réseaux ad hoc à un saut.

La localisation de services ou de ressources disponibles dans un réseau ad hoc implique à la localisation physique des nœuds les hébergeant. Le problème de localisation des nœuds et les différents services de localisation proposés dans la littérature pour les réseaux mobiles ad hoc seront présentés dans le paragraphe suivant.

4.1. Localisation de nœuds mobiles

La possibilité de connaître la localisation physique d'un nœud dans un réseau mobile ad hoc en fournissant seulement un identifiant du nœud, permet une classe toute entière d'applications, tel que le routage géographique de paquets. En effet, la gestion de la localisation des nœuds dans les réseaux ad hoc mobiles est fortement motivée par l'adoption du routage géographique de paquets. Celui-ci s'adresse principalement au problème de passage à l'échelle. En utilisant le routage géographique, une source a besoin d'avoir un moyen d'apprendre la position géographique de n'importe quelle destination pour être capable d'étiqueter les paquets avec la position de cette destination. Les protocoles de routage géographique pour les réseaux ad hoc mobiles sont alors dépendants de l'existence de services de localisation capables de fournir aux sources les positions des destinataires. La disponibilité d'un service de localisation efficace dans le contexte de réseaux ad hoc mobiles constitue un

problème difficile vue l'absence d'infrastructure. Lorsqu'un nœud source demande la position d'un nœud destinataire, il a uniquement connaissance de l'identifiant du nœud destinataire. Le problème consiste à établir un rapport dynamique entre l'identifiant d'un nœud destinataire et sa localisation pour permettre le routage géographique de paquets vers ce nœud destinataire [21].

4.1.1. Services de localisation

Les services de localisation de nœuds mobiles dans un réseau ad hoc peuvent être classés selon les critères déjà cités plus haut, à savoir l'architecture supportant le service de localisation qui est distribuée ou centralisée, la stratégie de mise à jour qui peut être réactive ou proactive, et la structure de gestion de l'information de localisation, selon quelle soit hiérarchique ou non hiérarchique. Une classification des différents services faite dans [20] est montrée sur la figure 37. Elle les classe d'abord selon deux principaux types selon qu'ils soient réactifs ou proactifs:

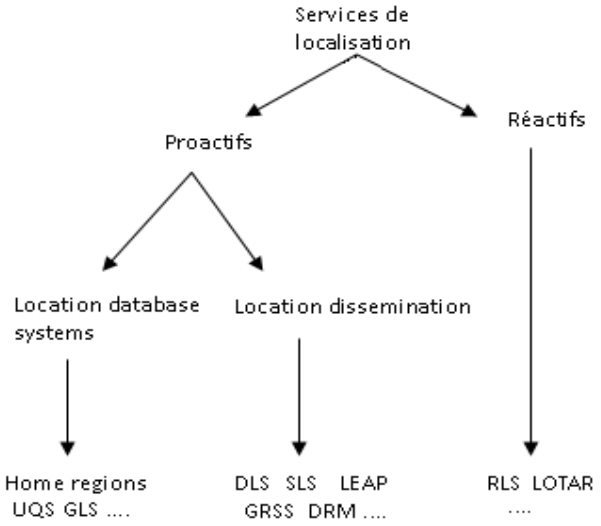


Figure 37 : *Classification des services de localisation.*

4.1.1.1. Les services de localisation réactifs

Les services de localisation réactifs cherchent la position d'un nœud seulement quand ils reçoivent une requête de localisation. Cela réduit les coûts de mise à jour mais ça encourt un délai avant chaque réponse. Comme exemple de ce type de services, nous présentons les services réactifs suivants :

- ***RLS*** (*Reactive Localization Service*) [19]. Lorsqu'un nœud veut connaître la position d'un certain nœud destinataire (i.e., avant de lancer une communication assurée par un routage géographique), il lance une requête de localisation contenant sa position, son identificateur, ainsi que l'identificateur de la destination. Dans le service de localisation réactif, RLS, la requête est diffusée d'abord aux voisins. Si la source ne reçoit pas de réponse positive de ses voisins après un certain délai, la requête est diffusée à tout le réseau. Quand le destinataire reçoit la requête, il envoi une réponse au nœud source

contenant sa position, qui sera insérer dans tous les paquets en direction du destinataire pendant la communication.

- ***LOTAR*** (*LOcation Trace Aided Routing protocol*) [19] est basé sur la réduction de la zone de recherché en prévoyant la durée de vie d'une route. Pour cela, chaque nœud sauvegarde dans sa table de localisation les informations les plus récentes sur d'autres nœuds, tels que la position, la vitesse et la direction d'un nœud à un temps donnée.

4.1.1.2. Les services de localisation proactifs

Les services de localisation proactifs diffusent continuellement les informations de mise à jour. Le temps de réponse aux requêtes de localisation est plus petit, cependant les mises à jour proactives sont consommatrices en bande passante, ce qui n'est pas intéressant quand les requêtes de localisation sont peu fréquentes.

Les services proactifs sont à leur tour classés en deux types [20]:

a) ***Les services de dissémination***, où tous les nœuds du réseau reçoivent périodiquement les mises à jour sur les informations de localisation d'un nœud donné. Ainsi, quand un nœud donné a besoin d'une information de localisation d'un autre nœud, il l'a trouve disponible dans sa table de localisation. Quelques services de dissémination sont présentés ci-dessous :

- ***DLS*** (*Dream Location Service*) est un service de localisation qui met à jour les tables de routage de localisation des nœuds, à l'aide d'une information de localisation, appelée paquet de localisation, échangée périodiquement entre les nœuds du réseau et qui contient la position du nœud source, sa vitesse et l'instant de l'envoi du paquet. Le paquet de localisation est envoyé vers les nœuds les plus proches de la source avec un rythme donné, par contre il est envoyé avec un rythme plus petit vers les nœuds lointains. En effet, dans DLS, il n'est pas nécessaire de maintenir l'information de localisation des nœuds éloignés, puisqu'ils ont l'air d'avoir une vitesse de déplacement plus lente que les nœuds les plus proches par rapport à la source. Ceci est expliqué sur la figure 38, qui montre l'effet du déplacement des nœuds B et C par rapport au nœud A, lorsque la distance le séparant des deux nœuds est différente.

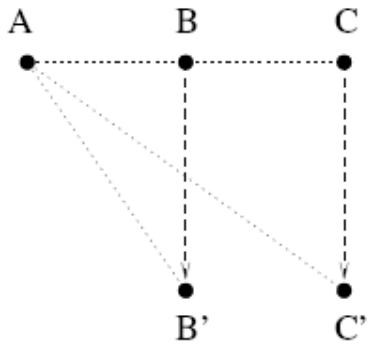


Figure 38 : *L'effet de la distance relative.*

- ***SLS*** (*Simple Location Service*) est un service qui transmet les informations de localisation uniquement aux nœuds voisins. En plus, chaque paquet de localisation qui met à jour les tables de localisation, contient la position de plusieurs nœuds, la vitesse de chacun d'eux ainsi que leur temps d'émission.
- ***LEAP*** (*Legend Exchange and Augmentation Protocol*) est un autre service de localisation de type dissémination. Il manipule deux types de tables de localisation. Une table locale est stockée au niveau de chaque nœud et contient les informations de localisation des autres nœuds du réseau. Une autre table de localisation dite globale, contient les informations de localisation de tous les nœuds ainsi que des informations pour décider de l'itinéraire à suivre par cette table. En effet, la table de localisation globale migre (*leaps*) d'un nœud à un autre nœud dans le réseau, tout en collectant les informations de localisation des différents nœuds traversés. Le contenu de cette table est distribué à tous les autres nœuds du réseau, mettant à jour leurs tables locales si les informations de localisation sont plus récentes.
- ***DRM*** (*Dead Reckoning Method*) [61] est un service de localisation basé sur une technique de prédiction de localisation. Chaque nœud construit un modèle de ses mouvements qui est alors diffusé avec sa position courante. Le modèle est construit en calculant différentes valeurs de vitesse, pris en plusieurs échantillons de positions et à des instants successifs. Les autres nœuds peuvent alors prévoir le mouvement de chacun des autres nœuds dans le réseau.

b) ***Les services de bases de données***, où des nœuds spécifiques dans le réseau servent comme une base de données de localisation pour d'autres nœuds spécifiques du réseau, i.e., ce sont des serveurs de base de données de localisation qui sont sollicités pour traiter les requêtes de localisation et pour mettre à jour leurs bases de données en cas de changement des positions des nœuds dans le réseau. Trois types de localisation existent pour les services proactifs de bases de données:

- ***Service de localisation basé sur les Grilles*** qui est une approche de localisation qui partage la région de déploiement en une hiérarchie quelconque. Les nœuds qui se trouvent dans les différentes parties de cette hiérarchie jouent le rôle de serveur de localisation distribués en différents niveaux de densité. Ci-dessous les services de localisation de grille les plus importants :
  - **GLS** (*Grid Location Service*) [62] structure l'espace couvert par le réseau ad hoc en grilles hiérarchiques avec des carrés de taille croissante (figure 39). Il utilise un ensemble de nœuds pour être des serveurs de localisation pour un nœud donné, et qui sont déterminés dans des grilles géographiques prédéfinies. Comme le groupe de serveurs de localisation pour chaque nœud est différent du groupe correspondant pour les autres nœuds, l'information de localisation est distribuée à travers le réseau. La sélection de l'ensemble des serveurs de localisation est telle qu'un nœud choisit dans chaque grille, le nœud ayant le plus petit identificateur qui est supérieur à son identificateur, comme l'un de ses serveurs de localisation. L'exemple de la figure 39

montre les serveurs de localisation sélectionnés (entourés par un cercle) pour le nœud B (identificateur 17).

Quand un nœud se déplace, il envoie un paquet de mise à jour à tous ses serveurs de localisation dans le réseau. Pour réduire le trafic induit par la mise à jour de l'information de localisation, l'étendue de la mise à jour vers les serveurs de localisation est proportionnelle à la distance parcourue par le nœud depuis la dernière mise à jour. Ainsi, les tables de localisation des serveurs éloignés sont mises à jour moins fréquemment que celles des serveurs voisins.

Lorsqu'un nœud veut envoyer un paquet de données, et qu'il ne trouve pas la position de la destination dans sa table de localisation, il lance une requête de localisation vers le serveur de localisation du destinataire qui se trouve dans sa grille. Dans l'exemple, le nœud A qui veut communiquer avec B, contacte le nœud serveur de B dans sa grille (identificateur 20) pour apprendre la position de celui-ci.

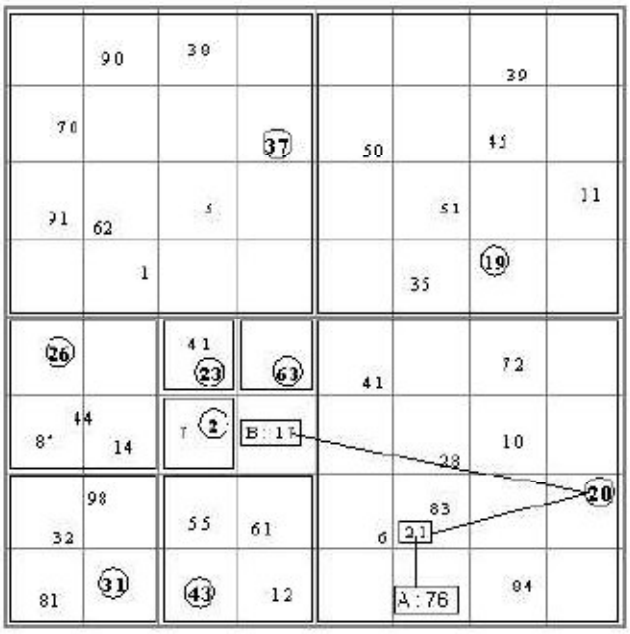


Figure 39: *Exemple de grille dans GLS.*

- **DC** (*Doubling Circles*) est un service similaire à GLS, cependant, il structure le réseau en cercles. Chaque cercle de rayon *r* mètres, a comme centre la position d'un nœud donné. Lorsqu'un nœud se déplace et quitte son cercle de rayon *r*, il transmet une mise à jour de sa position à tous les nœuds à l'intérieur d'un cercle de rayon *(r+1)* mètres, centré par la nouvelle position du nœud. La figure 40 montre un exemple de déplacement du nœud A qui quitte sa zone d'appartenance (cercle en continu de rayon *r*). La mise à jour de sa position concerne tous les nœuds dans le cercle (en discontinu) de rayon (*r+1*).

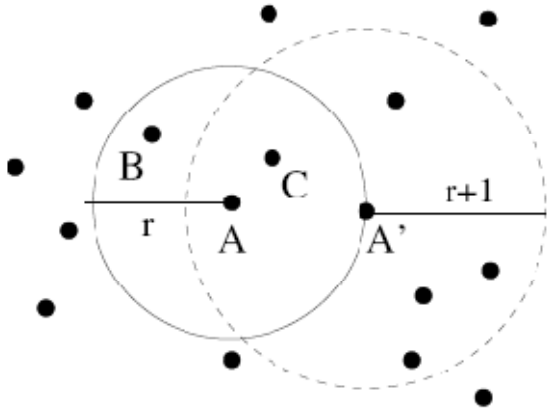


Figure 40: *Exemple de mise à jour d'informations de localisation dans DC.*

- ***Service de localisation basé sur home région,*** qui est un service de localisation similaire à celui utilisé dans les réseaux cellulaires téléphoniques et l'IP mobile. L'approche associe une région à chaque nœud par une fonction de hachage dont le paramètre est l'identifiant du nœud. Un nœud utilise son identifiant pour nourrir la fonction de hachage et ainsi obtenir une position appartenant à la région de déploiement. Une région est définie par un rectangle (figure 41) ou par un cercle ayant comme centre, la position d'un nœud donnée. Si le nombre de nœuds dans une région est trop important ou trop petit, la taille de la région est adaptée dynamiquement en conséquence, dans le but de maintenir un nombre de nœuds approximativement constant dans une région.

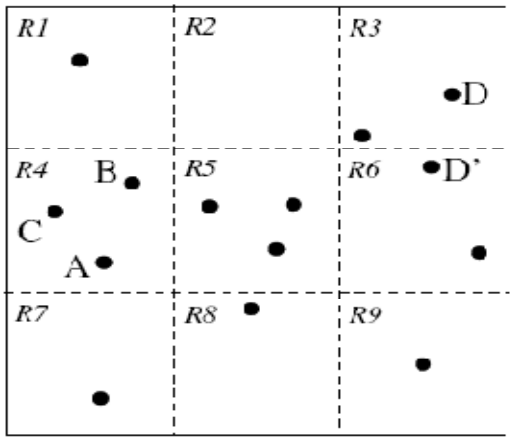


Figure 41: *Exemple d'un service de localisation home region.*

Les nœuds qui se trouvent dans la région sélectionnée jouent le rôle de serveur de localisation pour le premier nœud. D'une façon similaire, les nœuds qui veulent apprendre la position d'un nœud d'après l'identifiant du nœud utilisent la même fonction de hachage pour savoir où chercher l'information. La région déterminée par la fonction de hachage peut être vue comme un point de rendez-vous entre les mises à jour du positionnement d'un nœud et les nœuds qui désirent connaître la position du même

nœud. Dans l'exemple de la figure 41, basé sur une approche rectangulaire, lorsque le nœud D appartenant à la région de déploiement R4, se déplace de R3 vers R6, les nœuds A, B et C recevrons un paquet de mise à jour de la position de D.

- ***Service de localisation basé sur les* quorums**, qui a pour principe l'identification d'un certain nombre de sous ensembles spéciaux de nœuds, agissant en tant que serveurs, appelés quorums. Il est défini deux types de quorums, les quorums de lecture et les quorums d'écriture, tel que l'intersection de deux quorums (voisins) dans le réseau est non nulle. Lorsqu'un nœud a besoin de mettre à jour une information de localisation, il envoie la mise à jour à un quorum d'écriture. Quand un nœud veut localiser un autre nœud dans le réseau, il envoie une requête de localisation vers le quorum de lecture. Ce type de service est basé aussi sur la réplication de l'information de localisation sur les différents serveurs d'un quorum. Dans la figure 42, par exemple, le nœud A envoie ses mise à jour au quorum d'écriture {1, 4, 7}. Lorsque B veut envoyer un message à A, il envois sa requête de localisation au quorum de lecture {2, 3, 7}.

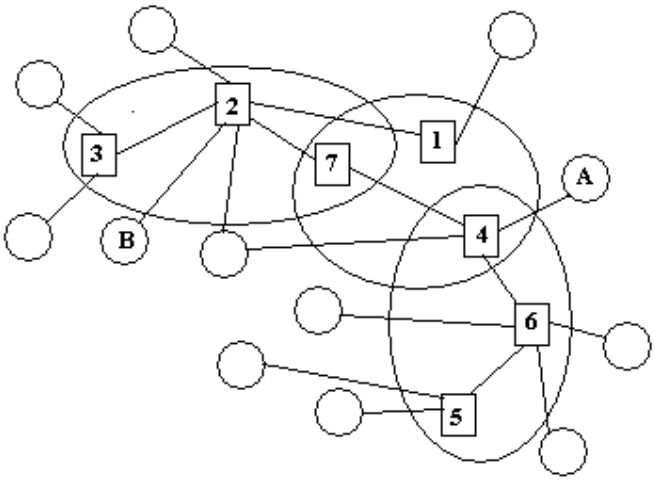


Figure 42: *Exemple de quorum de lecture et d'écriture.*

UNL (*Unreachable Nodes List*), DQL (*Dysnamic Quorum List*), UQS (*Uniform Quorum System*) et L/C (*Column/Row*) sont des exemples de services de localisation proactifs de base de données basés sur les quorums.

D'autres services de localisation dits hybrides, tel que le service PLS (*Predictive Location Service*), sont une combinaison des services réactifs et proactifs.

4.1.2. Services de positionnement

Comme déjà vu précédemment, les services de localisation permettent de fournir à un nœud source, la position d'un nœud de destination pour le compte d'un protocole de routage d'information géographique, utilisé dans les réseaux mobiles ad hoc. Ceci suppose bien sûr l'existence d'un moyen ou d'un autre qui permet à un nœud quelconque de connaître sa position à tout instant dans le réseau ad hoc, caractérisé par une absence d'une administration

fixe ou centralisée. De nombreuses techniques sont proposées pour permettre aux nœuds d'estimer leur position. Nous présentons ci-dessous le système de positionnement GPS.

Le système GPS

Avec la disponibilité récente de petit récepteur GPS, extrêmement légers et à bas prix, l'utilisation de système GPS a connue une forte expansion dans les réseaux mobiles ad hoc et les réseaux de capteurs. En effet, pour déterminer sa position chaque station mobile embarque un récepteur GPS qui se compose d'une puce GPS et d'une antenne de réception.

Le système GPS *(Global Positioning System)* ou Géo-Positionnement par Satellite, est le principal système de positionnement par satellites mondial actuel; de plus il est actuellement le seul à être entièrement opérationnel. Il existe d'autres systèmes de positionnement par satellite opérationnels, sans atteindre cependant la couverture ou la précision du GPS, on peut citer [21]:

- GLONASS est le système russe, qui n'est pas pleinement opérationnel.
- Beidou est le système de positionnement créé par la Chine; il est opérationnel uniquement sur le territoire chinois et régions limitrophes (il utilise des satellites géostationnaires, au nombre de quatre actuellement).
- L'Inde prépare également son système de positionnement
- Enfin, il y a Galileo qui est le système civil de l'Union européenne en cours de test depuis 2004. À terme, il est destiné à être au moins équivalent au GPS en termes de couverture et de précision.

Le système GPS, qui a été théorisé par le physicien D. Fanelli, a été mis en place à l'origine dans le cadre d'un projet de recherche de l'armée américaine. Il a été lancé dans les années 1960 et c'est à partir de 1978 que les premiers satellites GPS sont envoyés dans l'espace. Une seconde série de satellites a été lancée à partir de 1989 en vue de constituer une flotte suffisante. En 1995, le nombre de satellites disponibles (une constellation de 24 satellites) a permis de rendre le système GPS opérationnel en permanence sur l'ensemble de la planète. Le système GPS a par la suite connu un grand succès dans le domaine civil et engendré un énorme développement commercial dans de nombreux domaines : navigation maritime, sur route, localisation de camions, randonnée, etc. De même, le milieu scientifique a su développer et exploiter des propriétés des signaux transmis pour de nombreuses applications : géodésie (science de la forme et des dimensions de la terre), transfert de temps entre horloges atomiques, étude de l'atmosphère, etc.

Calcul de la position: la trilatération

Le système GPS comprend au moins 24 satellites orbitant à 20 200 km d'altitude. Cette constellation de satellites est conçue de telle manière à ce que chaque récepteur GPS voit à tout moment et à n'importe quel endroit du globe 4 satellites. Ces satellites émettent en permanence avec la vitesse de la lumière, un signal sur deux fréquences (1575,42 MHz pour le civil et 1227,60 MHz pour le militaire) contenant diverses informations utiles à la localisation (des données numériques et un ensemble de codes pseudo-aléatoires, datés

précisément grâce à leur horloge atomique). Connaissant la position du satellite à l'heure d'émission du signal, le récepteur GPS peut alors calculer la distance qui le sépare du satellite en connaissant le temps que le signal a mis pour parcourir ce trajet. Ainsi, un récepteur GPS qui capte les signaux d'au moins quatre satellites peut, en calculant les temps de propagation de ces signaux entre les satellites et lui, connaître sa distance par rapport à ceux-ci et, par trilatération, se positionner précisément en trois dimensions.

Théoriquement, pour le calcul de la position d'un objet mobile, seuls trois satellites suffiraient. C'est le cas par exemple, où l'on connaît l'altitude, comme lorsque l'on évolue au-dessus d'une surface plane (océan, mer). Lorsque plus de quatre satellites sont visibles (ce qui est très souvent le cas), la précision du calcul est améliorée, et on peut estimer les erreurs sur la position et le temps. En effet, à cause de la très grande vitesse de propagation des ondes (300 000 km/s), la synchronisation des horloges entre le récepteur GPS et les satellites doit être d'une extrême précision (une erreur de 0,1 microseconde mène à une erreur de 30 m !). C'est pourquoi le quatrième satellite entre en jeu.

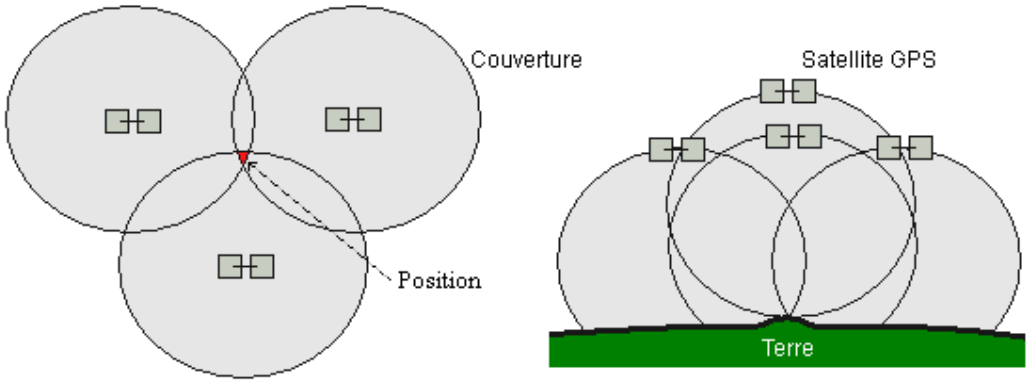


Figure 43: *Positionnement par GPS.*

Comme montré sur la figure 43, les récepteurs déterminent la distance les séparant de chaque satellite et peuvent former une sphère centrée sur chacun de ces satellites. L'intersection de ces sphères indique la position où se situe le récepteur. La position du récepteur ainsi connue permet de déterminer ses coordonnées cartésiennes géocentriques (X, Y, Z) ou ses coordonnées géographiques (latitude, longitude et son altitude).

Cependant, le système GPS comporte des points faibles, dont principalement l'obligation de visibilité des satellites par le navigateur, ce qui rend l'usage difficile en intérieur ou en terrain très accidenté. Dans certaines applications également, liées aux réseaux de capteurs par exemple, le coût du GPS, à la fois économique et énergétique, pose soucis. Quelques techniques contournent le problème et proposent l'utilisation d'ancres : seulement un certains nombre de nœuds, équipés de systèmes GPS, connaissent leur position précise, et permettent aux autres nœuds du réseau, par triangulation ou multilatération de connaître la leur.

4.2. Localisation d'agents mobiles

Les agents mobiles sont considérés comme un concept bien adapté pour développer des applications réparties grâce à leurs propriétés de tolérance aux fautes, d'autonomie et d'adaptation. Ils sont de plus en plus utilisés dans les réseaux mobiles ad hoc. Dans ce cas, deux niveaux de mobilité se superposent : celle des sites et celles des agents. Les sites peuvent changer de voisinage à tout moment, et les agents sont amenés à se déplacer de site en site à la recherche d'un service pour la réalisation de leur tâche. Dans ce contexte, il se pose le problème de la localisation et de l'accès au service désiré : l'agent client doit parvenir à rencontrer l'agent serveur sur un même site alors que sites et agents peuvent bouger simultanément.

Le seul travail à notre connaissance, sur la localisation d'agents mobiles dans les réseaux mobiles ad hoc, est ce lui de Lokpo et al. [65] qui ont proposé un service de localisation d'agents mobiles dans un réseau ad hoc, où la fonction de localisation est exécutée par les agents mobiles eux même, et ceci pour établir des rencontres entre agents mobiles dans le but de coopérer et de communiquer localement. Leur approche s'appuie sur un annuaire décentralisé d'agents et une stratégie de migration basée sur des informations sur la trace des agents.

Chaque site du réseau héberge un annuaire local qui enregistre les agents présents sur le site à un instant donné. Tout agent arrivant sur un site s'enregistre dans l'annuaire local du site et avant de le quitter, il laisse une trace de visite sous la forme du prochain site à visiter. Un agent qui cherche à rencontrer un autre agent consulte l'annuaire local pour retrouver sa cible. Si celle-ci n'est pas présente, alors l'agent interroge le service de localisation pour obtenir la trace de sa cible. S'il existe une trace de la cible, l'agent choisit de migrer vers le site fourni par la trace, sinon, l'agent choisit un schéma de migration aléatoire. Dans cette approche, toutes les mises à jour et les consultations des annuaires sont locales. Elles ne demandent pas de communication à distance.

Dans un autre article, les mêmes auteurs [43] proposent une autre approche basée sur des agents légers pour la localisation d'agents applicatifs mobiles (figure 44). Les agents légers se déplacent aléatoirement et essayent de construire des chemins permettant d'atteindre les agents applicatifs localisés sur les différents sites du réseau. Ces agents légers sont utilisés aussi pour la mise à jour des traces des agents applicatifs enregistrées sur chaque site.

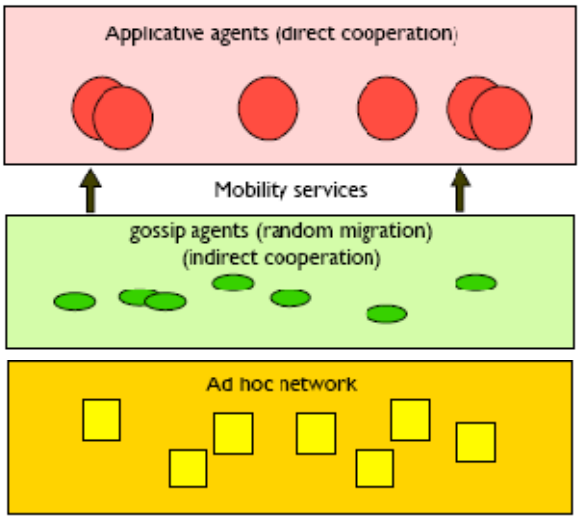


Figure 44 : *Architecture basée sur des agents légers.*

5. Conclusion

La gestion de la localisation qui est une étape très importante dans la gestion de la mobilité dans les réseaux mobiles, permet de fournir au réseau des informations sur la position courante d'un objet mobile ainsi que la mise à jour de ces informations. Le mécanisme de localisation qui est utilisé généralement dans la première phase de routage (notamment pour les protocoles de routage dit géographiques), permet d'assurer la communication entre les nœuds mobiles malgré le changement permanant de leurs positions. Dans un réseau ad hoc, les approches préconisées pour localiser les ressources, se basent sur des techniques de diffusion. Tous les nœuds ou la plupart d'entre eux sont mis à contribution dans le mécanisme de localisation, en conséquence, toutes les approches présentées sont de type distribuées. Cependant, une approche entièrement distribuée, basée sur la diffusion des requêtes et/ou des réponses dans l'ensemble du réseau, accroît l'encombrement dans un réseau caractérisé par une bande passante étroite. Il est alors nécessaire de proposer de nouvelles solutions afin de garantir une localisation efficace des ressources dans un réseau ad hoc, i.e. prenant en compte les contraintes caractéristiques des réseaux ad hoc. De plus, les solutions proposées doivent fournir à l'utilisateur un niveau de qualité de service satisfaisant, ce qui signifie en particulier, offrir un temps d'attente réduit pour l'utilisateur et une disponibilité permanente du service. Les deux chapitres suivants décrivent les différents protocoles que nous avons proposés pour la localisation des agents mobiles dans les réseaux mobiles ad hoc.

Chapitre 5

Protocole de localisation réactif vs. proactif : Une approche distribuée

1. Introduction

Très peu de travaux sur la localisation du code mobile ou agent mobile en environnement mobile sont présentés dans la littérature. Pourtant, le mécanisme de localisation dans ce cas est très complexe, puisqu'il s'agit ici de localiser un objet à double mobilité, celle de son support physique dans un environnement dynamique, la station, et celle de son code à travers les stations du réseau. Lorsque le code mobile représente une migration d'activité, le problème de localisation est triviale pour le système, vu que c'est ce dernier qui initie le processus de migration du code, et donc l'itinéraire de celui-ci est soit connu par le système ou bien il est prédictible, notamment lorsque le système connait les raisons de la migration de l'activité et ses objectifs. Il n'en est de même lorsque le code mobile est un agent mobile, qui est supposé être une entité autonome et intelligente, et donc il est difficile de prévoir quand s'effectuera la prochaine migration de l'agent et quelle sera sa destination suivante.

Il y a tendance à confondre les mécanismes de localisation des agents mobiles avec celles des nœuds mobiles d'un réseau. Il semblerait qu'il s'agit de localiser simplement des objets mobiles dans un environnement mobile. Nous pensons que le problème est tout autre et ce pour plusieurs considérations. D'abord ils n'ont pas la même vitesse de déplacement. Celle des agents est beaucoup plus importante que celle des stations. Ajouté à cela, le déplacement des stations étant physique, cela peut permettre la prédiction des nouvelles positions à atteindre par les stations sinon la cerner par rapport à une région donnée. Il n'en est de même pour les agents mobiles qui se caractérisent par un déplacement logiciel, difficile à cerner lorsque la portée des signaux de transmissions est importante. Par contre, le résultat de la localisation d'un agent mobile s'exprime en termes d'adresse ou de nom de la station hébergeant l'agent qui ne change que lorsqu'il y a migration de l'agent, alors que celle d'une station est une position qui s'exprime en termes de coordonnées géographiques et qui changent de façon continue selon le degré de mobilité de la station.

Nous présentons dans ce chapitre notre première approche de localisation de type distribuée, qui servira pour une étude comparative avec une autre approche de localisation qui sera présentée dans le prochain chapitre. Nous nous somme intéressés dans cette thèse au problème de localisation pour tester notre alternative de gestion pour les réseaux ad hoc.

2. Approche distribuée pour la localisation d'agents mobiles dans un réseau mobile ad hoc

Notre approche distribuée est une adaptation du protocole distribuée de Alouf et al. [2], pour la localisation des agents mobiles dans les réseaux filaires, aux réseaux mobiles ad hoc. Nous remarquons que l'utilisation d'un protocole distribué, basé sur une chaîne de répéteurs ou liens de poursuite, dans un environnement mobile, pose le problème de la rupture de la chaîne. Ceci est dû à la mobilité des stations marquées, i.e., les stations traversées par l'agent mobile. La localisation de l'agent mobile à l'aide d'un protocole distribué doit ainsi passer par le rétablissement de la chaîne de répéteurs lorsqu'elle est rompue. Nous proposons alors deux solutions à ce problème de rupture de la chaîne:

2.1. Protocole proactif de localisation d'agents mobiles

La première solution que nous proposons est de type proactif. Elle essaie de maintenir la chaîne de répéteurs dans un état établie ; pour cela, il faut à chaque fois établir la chaîne immédiatement après sa rupture. Cette méthode permettrait de garder la trace de l'agent mobile accessible à tout instant et offrirait ainsi un temps de réponse optimal de la localisation, lorsqu'il y a besoin de contacter l'agent. Cependant, lorsque l'agent se déplace beaucoup, la chaine de répéteurs risque de devenir longue et sa maintenance peut devenir coûteuse. Il est alors important d'optimiser la chaîne, quand par exemple une boucle est formée. C'est le cas où l'agent mobile passe deux fois ou plus par la même station, ce qui signifie qu'une station marquée possède au moins deux successeurs pour le même agent mobile (figure 45).

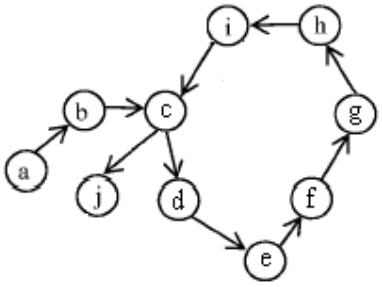


Figure 45: *Formation d'une boucle dans une chaîne de répéteurs.*

Les étapes du protocole proactif de localisation d'agents mobiles dans un réseau ad hoc sont comme suit:

1) Après chaque migration, l'agent mobile laisse un répéteur sur la station courante qui devient une station marquée. Le répéteur est lié à l'agent mobile sur la prochaine station, et ceci crée dynamiquement une chaîne de répéteurs.

2) Les stations marquées voisines échangent périodiquement et continuellement des messages de contrôle, dans le but d'assurer la maintenance de la chaîne dans un état établie.

3) Lorsqu'un contact est perdu entre deux stations marquées voisines, de ***i*** vers ***j***, la station ***i*** diffuse une requête de localisation à la recherche de son successeur, la station ***j***, et ce dans le but de rétablir le contact avec elle. La chaîne sera ainsi rétablie en rajoutant éventuellement de nouvelles stations entre ***i*** et ***j*** (figure 46).

4) Les stations marquées d'ordre supérieur qui interceptent la requête de diffusion, répondent à la station ***i*** dans le but d'optimiser la chaîne, et ceci dans le cas où il existe une de ces stations qui est plus proche de ***i*** par rapport à ***j***. La chaîne est ainsi rétablie et mise à jour (figure 48).

5) La localisation de l'agent mobile se fera alors de la même manière que dans le cas du réseau fixe, puisque la chaîne est supposée tout le temps établie.

(a) Avant la rupture de la chaîne **(b)** Après la rupture et le rétablissement de la chaîne

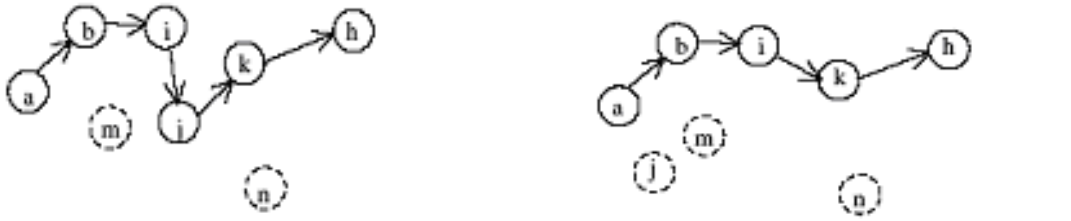
Figure 46: *Chaîne rétablie sans optimisation.*

La figure 46.a montre un exemple de stations marquées constituant une chaîne de répéteurs dans un réseau ad hoc, dans l'ordre de ***a*** à ***h***. Relativement à un mouvement de la station ***j***, la chaîne est rompue au niveau de la station ***i*** qui lance un processus de localisation de ***j***. Au même moment, cette dernière lance le même processus pour localiser ***k*** puisqu'elle a perdue le contact avec sa prochaine station marquée. Le rétablissement de la chaîne dans ce cas est montré dans la figure 46.b qui est effectué en insérant de nouvelles stations dans la chaîne (***m*** et ***n*** dans l'exemple).

(a) Avant la rupture de la chaîne **(b)** Après la rupture et le rétablissement de la chaîne

Figure 47: *Chaîne rétablie sans la station causant sa rupture.*

La figure 47.b montre un rétablissement de la chaîne rompue dans la figure 47.a au niveau de la station ***i***. La station marquée ***k***, d'ordre supérieur à ***j***, qui reçoit une requête de contact venant de ***i***, permet de rétablir la chaîne sans passer par ***j*** mais en insérant de nouvelles stations.



(a) Avant la rupture de la chaîne **(b)** Après la rupture et le rétablissement de la chaîne

Figure 48: *Chaîne rétablie avec optimisation.*

Le rétablissement de la chaîne de la figure 48, toujours rompue au niveau de la station ***i***, est accompli de manière optimale puisque la station ***k***, d'ordre supérieur à celui de ***j***, est dans la portée de ***i***.

2.2. Protocole réactif de localisation d'agents mobiles

La seconde solution est de type réactif. Le protocole proposée, qui est une alternative classique au protocole proactif, est basée sur un rétablissement de la chaîne de répéteurs lorsqu'elle est rompue, juste au moment du besoin de la localisation de l'agent mobile. La méthode réactive tente d'éviter les surcoûts liés au maintient de la chaîne dans un état établit. Ce qui permettrait de réduire la charge dans le réseau, notamment lorsqu'un nombre important d'agents mobiles circule dans le réseau où il faudra maintenir alors autant de chaînes de répéteurs. Les étapes du protocole réactif de localisation d'agents mobiles dans un réseau ad hoc sont comme suit:

1) Même étape que dans le cas du protocole proactif.

2) Pour localiser un agent mobile, la station mère de l'agent contacte alors la première station mobile marquée dans la chaîne, qui contacte à son tour la prochaine station marquée, et ainsi de suite jusqu'à ce que l'agent mobile soit localisé.

3) Quand une station marquée ***j*** est non accessible par une station ***i*** (la chaîne est rompue), une diffusion limitée est lancée pour la recherche de stations marquées de la chaîne, avec un ordre supérieur à la station ***i*** et qui sont éventuellement dans sa portée de transmission.

4) Les stations marquées d'ordre supérieur à la station ***j*** qui interceptent la requête de diffusion, répondent à la station ***i*** dans le but d'optimiser la longueur de la chaîne, et ceci dans le cas où ils existent une de ces stations qui est la plus proche de ***i*** par rapport à ***j***.

3. Protocole distribué réactif versus proactif : une étude comparative

Notre étude expérimentale des performances des protocoles de localisation proposés a été réalisée à l'aide du simulateur réseau NS (*Network Simulator*).

3.1. Simulateur NS

NS (*Network Simulator*) [4] est un outil logiciel de simulation de tout type de réseaux informatiques, développé dans le cadre du projet VINT (*Virtual InterNet Testbed*) de l'Université de Berkeley en Californie. Ce projet a été fondé par DARPA (*Defense Advanced Research Projects Agency*) en collaboration avec le XEROX PARC (*Palo Alto Research Center*) et le LBNL (*Lawrence Berkeley National Laboratory*. Le simulateur est principalement bâti avec les idées de la conception par objets, de réutilisabilité du code et de modularité. Il est devenu aujourd'hui un standard de référence en ce domaine. C'est un logiciel dans le domaine public disponible sur l'Internet. Son utilisation est gratuite. Le logiciel est exécutable tant sous Unix que sous Windows.

NS a pour objectif principal de construire un simulateur multi-protocoles pour faciliter l'étude de l'interaction entre la plupart des protocoles et le comportement d'un réseau à différentes échelles. Il contient des bibliothèques pour la génération de topologies réseaux, des trafics ainsi que des outils de visualisation telle que l'animateur réseau NAM (*Network AniMator*). NS est un simulateur à événements discrets orienté objet. Ses modules de base sont écrit en C++ avec une interface textuelle (ou shell) au-dessus, qui utilise le langage OTcl (*Object Tool Command Langage*). Le langage C++ sert à décrire le fonctionnement interne des composants de la simulation. Pour reprendre la terminologie objet, il sert à définir les classes. Quant au langage OTcl, qui est une extension objet au langage de commande Tcl, il fournit un moyen flexible et puissant de contrôle de la simulation comme la description des éléments qui constituent le réseau, sa configuration, le déclenchement d'événements, la collecte de statistiques, etc.

L'application NS se compose de deux éléments fonctionnels: un interpréteur et un moteur de simulation. Au moyen de l'interpréteur, l'utilisateur est capable de créer le modèle de simulation ce qui revient à assembler les différents composants nécessaires à l'étude. Les composants du modèle de simulation sont appelés objets ou encore instances de classe. Le moteur de simulation effectue les calculs applicables au modèle préalablement construit par l'utilisateur via l'interpréteur. Un modèle de réseau en NS est constitué :

– de nœuds de réseau : sources où est généré le trafic, ou nœuds de routage.
– de liens de communication entre les réseaux.
– d'agents de communication, représentant les protocoles de niveau transport (TCP, UDP); ces agents sont attachés aux nœuds et connectés l'un à l'autre, ce qui représente un échange de données (connexion TCP, flux UDP).
– d'applications qui génèrent le trafic de données selon certaines lois, et se servent des agents de transport.

Le simulateur NS est un logiciel très évolutif. En effet, de nombreux composants sont intégrés chaque année. Ainsi, il existe de nombreuses versions pour les différents protocoles

implémentés. De plus, il y a un grand nombre de classes prédéfinies qui permettent de mettre en œuvre plusieurs implémentations intégrant des protocoles de transmission TCP, UDP et IP, des files d'attentes diverses (files de paquets), un routage fixe et dynamique. Le simulateur de réseaux NS sous Linux est disponible en version dite, tous en un, (*AllInOne*) en incluant plusieurs composants dont : Tcl, Otcl, Tclcl, Xgraph, Ns, Nam (tableau 2). Le groupage de ces différents composants rend l'utilisation de l'environnement de simulation plus facile puisque les liens sont automatiquement établis (entre les composants), ce qui n'est pas le cas pour les systèmes Windows où ces composants sont installés indépendamment les uns des autres en imposant ainsi quelques manipulations pour la mise au point du simulateur (définition des liens entre les différents composants installés suivant un ordre chronologique respecté).

La mobilité a marqué sa première incorporation dans l'environnement NS en 1997 à travers l'inclusion des premiers codes correspondants aux environnements sans fil, en permettant ainsi la prise en charge des communications sans fil (pour les réseaux cellulaires et ad hoc). Le développement de l'environnement NS a permis alors la simulation de plusieurs protocoles pour les réseaux mobiles ad hoc, tels que les protocoles de routage : *DSR*, *DSDV*, *AODV*, les protocoles de transport : *Unicast* (*TCP* et *UDP*), *Multicast*, etc. Ainsi, il est possible de définir dans l'environnement NS des nœuds de type mobiles, pouvant se déplacer dans un espace délimité et communiquant à travers des canaux de communication sans fil, en utilisant des agents de communication correspondants au protocole de transport utilisé. Par conséquent, plusieurs paramètres sont utilisés pour la configuration des nœuds mobiles dont le type de l'interface réseau, type du canal de transmission, type du protocole de la couche physique, type de la propagation radio utilisée et le type du protocole de routage utilisé.

TCL	Langage de script en open source, utilisé pour programmer NS. Il faut l'installer en premier.
TK	C'est une interface utilisateur graphique pour TCL. Elle s'installe juste après.
OTCL	Extension de Tcl/Tk pour la programmation Orientée Objet.
Tclcl	Interface Tcl/C++, elle permet de faire le lien entre la hiérarchie existante en C++ et celle existante en Tcl dans Ns. Grâce à ce composant on peut utiliser les deux langages dans une même simulation, de cette façon on obtient un simulateur flexible et puissant.
NS	Programme final. Ns a besoin des 4 composants précédents pour fonctionner.
Nam	Permet de visualiser graphiquement une simulation.
X-Graph	Afficher des graphiques pouvant contenir les instants d'émission des paquets

Tableau 2 : *Composants de NS.*

NS fournit trois modèles de mobilité pour simuler le mouvement des nœuds :

- RWM *(Random Waypoint Mobility)* est le modèle de mobilité le plus simple et qui est le plus utilisé dans l'évaluation des algorithmes de positionnement. Le modèle RWM inclut des temps de pause entre chaque changement de direction et/ou de vitesse. Un nœud marque un temps de pause sur sa position avant de choisir une destination, de façon aléatoire, sur l'ensemble de l'aire de simulation et choisit aussi une vitesse distribuée, de façon uniforme, entre un seuil minimum et un seuil maximum, ensuite il s'achemine jusqu'à la destination à la vitesse choisie. Une fois arrivé à destination, il reste sur place durant le temps de pause et recommence ce processus.
- BSAM *(Boundless Simulation Area Mobility)* est un modèle qui s'affranchit des bordures de l'aire de simulation. En effet, dans les modèles de mobilité traditionnels, les nœuds mobiles se réfléchissent sur les bords, et ont tendance ainsi à se concentrer au centre. Ici, les nœuds traversent les limites de la zone de simulation. Lorsqu'ils atteignent une bordure, ils continuent leur chemin et réapparaissent du côté opposé. Les bordures sont donc reliées deux par deux et l'aire de simulation prend alors la forme d'un tore.
- GMM *(Gauss Markov Mobility)* est un modèle conçu pour s'adapter à différents degrés d'entropie via un unique paramètre. Tout d'abord, les nœuds sont distribués aléatoirement sur l'aire de simulation. Puis, à chaque itération la vitesse et la direction de chaque nœud sont recalculées ; ces deux paramètres dépendent respectivement de la vitesse et de la direction à l'instant précédent ainsi que d'une variable aléatoire. A chaque itération, sont également évaluées les nouvelles coordonnées des points qui dépendent de la vitesse et de la direction calculées.

3.2. Environnement de Simulation

L'implémentation de nos protocoles de localisation a été réalisée à l'aide de la version 2.30 de NS et sous le système d'exploitation LINUX de la distribution MANDRIVA 2006. Nous avons utilisé les opportunités offertes par le langage Tcl, aussi bien pour l'écriture des différentes parties du protocole de localisation, que pour la définition des paramètres de l'environnement de simulation. Le modèle de mobilité utilisé dans notre étude est le modèle RWM. L'environnement de notre simulation est caractérisé comme suit :

- Une étendue réseau d'une superficie de 1000 X 500/1000 mètres^2.
- La taille du réseau varie entre 25 et 200 nœuds distribués aléatoirement sur la superficie réseau.
- Une portée de communication de 250 mètres, permettant une couverture radio des nœuds sur un cercle de 250 mètres de rayon (chaque nœud peut l'augmenter durant la simulation suivant ses besoins).
- Une durée de simulation de 100 secondes.

L'étude du comportement de nos protocoles de localisation a été faite selon les paramètres suivants :

3.2.1. Mobilité des nœuds

La mobilité des nœuds est définie par rapport à leurs mouvements relatifs, qui donnent une bonne estimation de la manière dont les nœuds se déplacent les uns par rapport aux autres. Lorsque les nœuds sont en mouvement pendant une certaine durée de temps, alors la mobilité ***Mi*** d'un nœud ***i*** est définie comme étant la moyenne des changements de distances entre tous les nœuds :

$$Mi = (1/(T-\Delta t))\sum_{t=0}^{T-\Delta t}\left|Ai(t+\Delta t) - Ai(t)\right|$$

Où ***T*** est la période de simulation et ***Δt*** est le pas de calcul. ***Ai(t)*** représente la moyenne des distances séparant un nœud ***i*** de tous les autres nœuds à l'instant ***t***, dans un réseau avec ***n*** nœuds. Elle est calculée comme suit :

$$Ai(t) = (1/(n-1))\sum_{j=1}^{n-1} Dist(Ni, Nj)$$

La mobilité moyenne de tous les nœuds est définie alors, par:

$$Mob = (1/n)\sum_{i=1}^{n} M_i$$

Nous distinguons ainsi différents degrés de mobilité :

- mobilité faible : $0 < Mob <= 3$
- mobilité moyenne : $3 < Mob <= 8$
- mobilité élevée : $8 < Mob$

Pour notre simulation, nous avons proposé trois valeurs de mobilité pour les nœuds :
1.5, 5 et 10 correspondant respectivement à la mobilité faible, moyenne et élevée.

3.2.2. Mobilité de l'agent

Nous avons utilisé également trois degrés de mobilité pour l'agent mobile: faible, moyenne et élevée. Le degré de mobilité est calculé à partir du nombre de migration de l'agent mobile à travers les nœuds du réseau par rapport à la durée de simulation. La mobilité de l'agent correspond en quelque sorte, à son temps de résidence moyen au niveau des nœuds du réseau pendant la période de simulation. Pour 100, 20 et 10 déplacements de l'agent mobile, pendant toute la durée de simulation (100 secondes), on obtient les temps de résidence moyens respectifs suivants : 1, 5 et 10 s, correspondant respectivement aux degrés de mobilités : élevé, moyen et faible. La mobilité d'un agent est calculée comme suit :

Mobilité_agent (degré de mobilité) = Période de simulation / Nombre de déplacement

3.2.3. Charge des demandes de localisation

La charge est définie par une loi poissonnière distribuée de paramètre λ, elle correspond à la durée séparant les demandes de localisation successives. Les valeurs prises par ce paramètre dans notre simulation sont : 0.1, 0.125, 0.167, 0.25, 0.5, 1 et 2 correspondant respectivement aux intervalles ($1/\lambda$) 10s, 8s, 6s, 4s, 2s, 1s et 0.5s après lesquels un nœud génère une nouvelle requête de localisation. Ainsi, la charge des requêtes de localisation est inversement proportionnelle à la durée séparant deux demandes de localisation successives.

Les critères d'évaluation de performances de nos protocoles sont comme suit :

3.2.4. Nombre de messages

Cette métrique représente le nombre de messages moyen nécessaires au traitement d'une requête de localisation de l'agent mobile. Il est calculé comme suit:

Nombre de message = (nombre total de messages échangés durant la simulation) / (nombre de requêtes de localisation)

3.2.5. Temps de réponse

Cette métrique représente le temps de réponse moyen pour la localisation de l'agent mobile. Il est calculé comme suit:

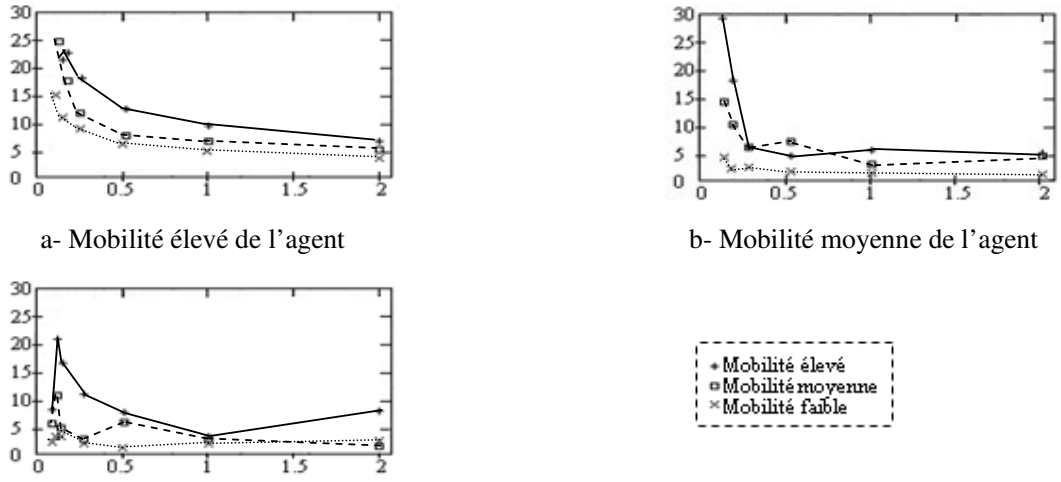
Temps de réponse = (temps total pour toutes les requêtes de localisation) / (nombre de requêtes de localisation).

3.3. Résultats expérimentaux et interprétations

Notons que les résultats d'expérimentation présentés ci-dessous concernant l'implémentation des deux protocoles distribués réactif et proactif, concernent le cas de la localisation d'un seul agent mobile qui migre dans le réseau ad hoc. Nous supposons aussi qu'il n'y a pas de partitionnements ni de pannes dans le réseau.

3.3.1. Nombre de message moyen

Les figures 49 et 50 montrent respectivement les performances des deux protocoles réactif et proactif en termes de nombre de messages moyen par requête de localisation, par rapport à la charge des requêtes, à la mobilité des nœuds et à celle de l'agent.

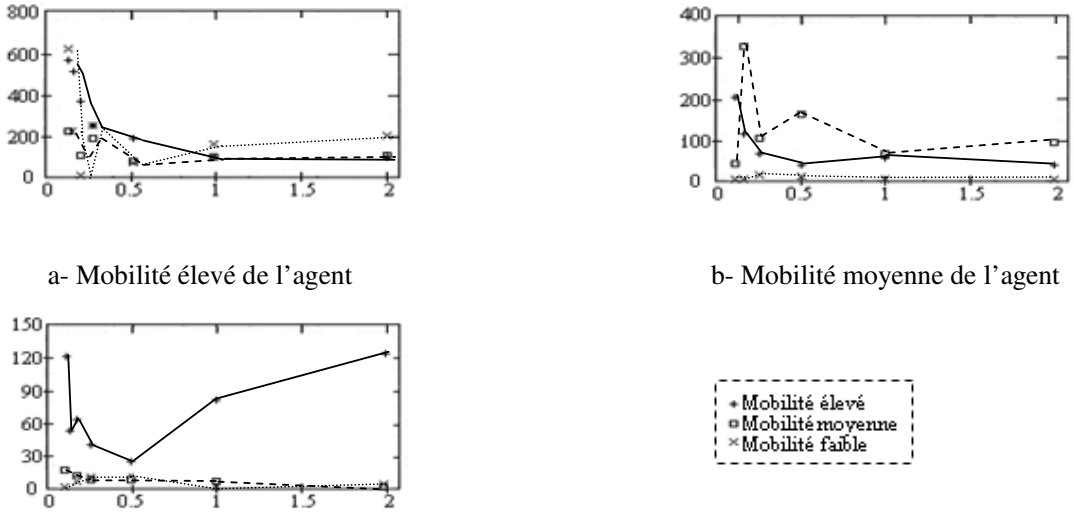


a- Mobilité élevé de l'agent

b- Mobilité moyenne de l'agent

c- Mobilité faible de l'agent

Figure 49 : *Nombre de message moyen par rapport à la charge des requêtes (protocole réactif).*



a- Mobilité élevé de l'agent

b- Mobilité moyenne de l'agent

c- Mobilité faible de l'agent

Figure 50 : *Nombre de message moyen par rapport à la charge des requêtes de localisation (protocole proactif).*

Nous remarquons pour les deux cas, que le nombre de messages moyen échangés par requête de localisation est inversement proportionnel à la charge de localisation. Lorsque la charge est importante, le nombre de migration de l'agent entre deux requêtes de localisation successives est minime. Ce qui explique que la localisation ne consomme pas beaucoup de messages dans ce cas. Cette logique n'est pas respectée dans le cas du protocole proactif lorsque la mobilité des nœuds est élevé (figure 50.c). Plus les nœuds sont mobiles, plus la fréquence de rupture de la chaîne des stations marquées est élevée, ce qui augmente le coût de maintenance de la chaîne en terme de messages. Ceci explique l'allure de la courbe. Notons aussi pour le protocole proactif, que le nombre de message croit avec l'augmentation du degré

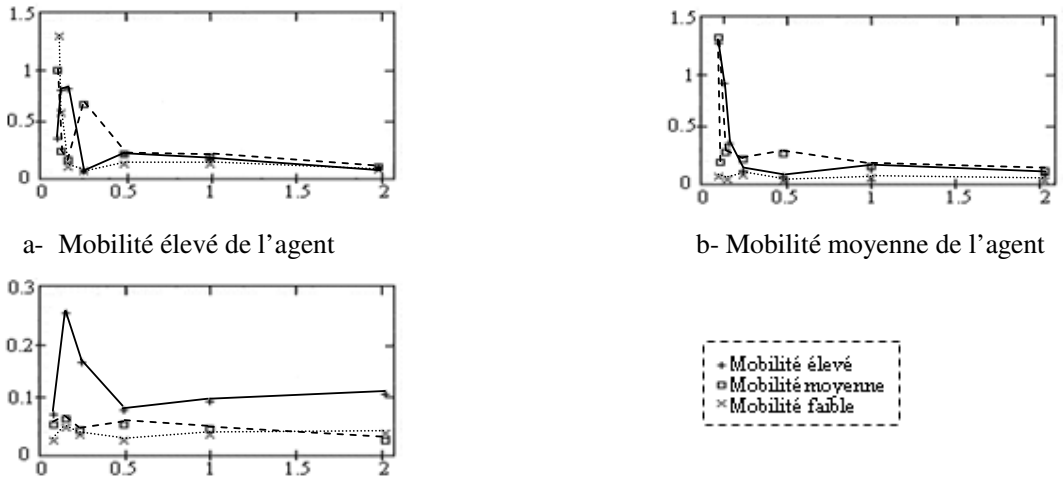
de mobilité de l'agent. Plus l'agent est mobile, plus la chaîne de stations créée est longue, et plus son coût de maintenance devient élevé en termes de messages.

Discussion

Nous remarquons que le protocole réactif est beaucoup moins couteux, en termes de nombre de messages moyen échangés par requête de localisation (30 messages au maximum) que le protocole proactif (plus de 600 messages). Ce nombre très important de messages pour le protocole proactif est dû aux messages de contrôle nécessaire au maintient de la chaîne des stations marquées dans un état établit. Ceci montre que cette maintenance de la chaîne dégrade les performances du protocole proactif en termes de messages échangés, ce qui consomme beaucoup de bande passante.

3.3.2. Temps de réponse moyen

Les figures 51 et 52 montrent respectivement les performances des deux protocoles réactif et proactif en termes de temps de réponse moyen par requête de localisation, par rapport à la charge des requêtes, à la mobilité des nœuds et à celle de l'agent.

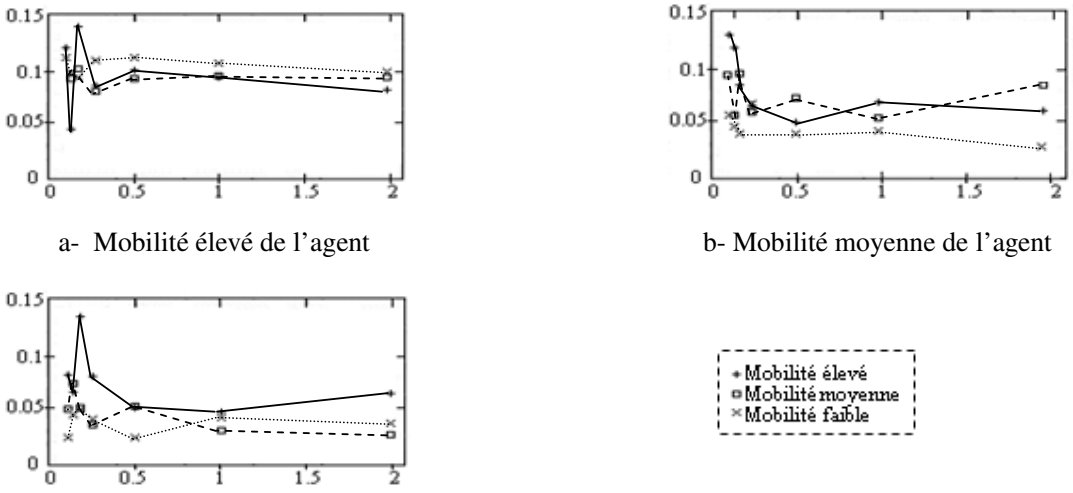


a- Mobilité élevé de l'agent

b- Mobilité moyenne de l'agent

c- Mobilité faible de l'agent

Figure 51 : *Temps de réponse moyen par rapport à la charge des requêtes (protocole réactif)*



a- Mobilité élevé de l'agent

b- Mobilité moyenne de l'agent

c- Mobilité faible de l'agent

Figure 52 : *Temps de réponse moyen par rapport à la charge des requêtes (protocole proactif).*

Comme pour le nombre de message, le temps réponse moyen est inversement proportionnel à la charge pour les deux protocoles, et il est proportionnel au degré de mobilité de l'agent pour le protocole proactif. Les explications données sont les même que pour le paramètre nombre de message.

Discussion

L'étude comparative entre les deux protocoles par rapport au temps de réponse moyen, montre que le protocole réactif est plus performant que le protocole proactif. Ceci est à l'encontre de ce qui était attendu. En effet, le temps de réponse pour le protocole proactif n'est pas meilleur par rapport à celui du protocole réactif, même si la chaîne des stations marquées est maintenue dans un état établit ; une telle gestion de la chaîne dans un réseau où les stations sont mobiles, génèrent un nombre énorme de messages de contrôle et de diffusion (figure 49). Ce qui provoque une surcharge du réseau, qui a un impact négatif sur le temps moyen de localisation.

Comme le protocole réactif est de loin le moins couteux et le plus adapté dans les réseaux mobiles ad hoc que le protocole proactif, nous présentons dans le prochain chapitre une étude comparative entre l'approche réactive du protocole distribué et le protocole centralisé.

4. Conclusion

Nous avons présenté dans ce chapitre notre première approche de type distribuée pour la localisation des agents mobiles dans les réseaux mobiles ad hoc. L'approche qui est une adaptation du protocole distribuée de Alouf [2], consiste en deux types de protocoles de localisation. Le premier protocole de type proactif, maintient l'information de localisation à chaque changement de topologie dans le réseau. Par contre, le deuxième protocole de type réactif, au lieu de maintenir l'information de localisation de façon proactive, cherche la

position de l'agent mobile seulement quand il reçoit une requête de localisation. Les résultats expérimentaux de l'étude comparative entre les deux protocoles, ont montré que le protocole réactif est plus performant que le protocole proactif, notamment en termes de nombre moyen de messages échangés. Cela confirme que les méthodes proactives sont gourmandes en bande passante, puisqu'elles induisent un flux important de messages de contrôle. Contrairement aux prévisions, le temps de réponse moyen du protocole proactif n'est pas meilleur par rapport à celui du protocole réactif, et ce malgré la disponibilité de l'information de localisation à travers la chaîne des stations marquées qui est maintenue dans un état établit. Ceci montre que la surcharge du réseau dû à la maintenance de la chaîne des stations marquées de manière proactive, a un impact négatif sur le temps moyen de localisation.

Après avoir étudié l'approche distribuée pour la localisation d'agents mobiles dans les réseaux mobiles ad hoc, une autre approche de type centralisé sera décrite et étudiée dans le prochain chapitre, où nous présenterons une étude comparative entre les deux approches.

Chapitre 6

Protocole de localisation multi-serveurs mobiles: Une distribution de l'approche centralisée

1. Introduction

Dans un réseau ad hoc, les approches préconisées pour localiser les ressources se basent généralement sur des techniques de diffusion [97] (dans la littérature, on parle généralement que de localisation de nœuds mobiles ou de services). Tous les nœuds ou la plupart d'entre eux sont mis à contribution pour satisfaire les requêtes de localisation ou maintenir les informations de localisation (dans le cas des approches proactives). Ces nœuds mobiles eux même forment, de manière ad hoc, une architecture globale qui peut être utilisée comme une infrastructure pour le réseau. En conséquence, toutes les approches présentées sont de type distribuées. C'est la nature même des réseaux ad hoc, à savoir l'absence de toutes administration centralisée ou fixe, qui ne fait penser qu'aux techniques distribuées.

Cependant, une approche entièrement distribuée, basée sur la diffusion des requêtes et/ou des réponses dans l'ensemble du réseau, n'est en particulier pas envisageable techniquement. En effet, le trafic augmente de façon exponentielle avec le nombre de nœuds présents dans le réseau ad hoc [97], ce qui pourrait influer négativement sur le temps de réponse aux requêtes de localisation. Est-il possible alors d'envisager une approche centralisée ? C'est à cette question que nous espérons répondre dans ce chapitre.

Notre alternative consiste en une conception d'un serveur centralisé pour la localisation dans un réseau ad hoc. L'un des objectifs de cette approche centralisée est de fournir une permanente disponibilité du service de localisation. Ceci en maintenant le serveur du réseau dans une région centrale, permettant ainsi un accès rapide par les nœuds du réseau. Notre approche proposée pour les réseaux ad hoc, où il est paradoxal de penser à une gestion centralisée, est basée sur la technologie d'agents mobiles. La mobilité des agents, est

exploitée afin de permettre une flexibilité et une auto-organisation de notre système. En effet, les agents mobiles constituent un serveur virtuel mobile, dans le sens où il n'est pas lié en permanence à des nœuds du réseau. Le serveur peut changer de support physique lorsque cela est nécessaire, et ce, pour offrir au mieux ses services aux utilisateurs. Par exemple, lorsque des nœuds sont en manque de capacité énergétique ou s'éloignent de la région centrale, supposée permettre une disponibilité permanente aux clients, le serveur réorganise ses agents en les invitant à migrer vers d'autres nœuds plus appropriés, assurant ainsi une continuité du service malgré la mobilité des nœuds supportant le serveur.

Nous présentons dans ce chapitre deux variantes de l'approche centralisée. Nous allons décrire à travers la première variante de l'approche qui est entièrement centralisée, le principe de fonctionnement de notre protocole de localisation, ensuite nous apporterons une amélioration à la première variante, en proposons une deuxième variante de type semi-centralisée. Une étude expérimentale des performances des deux protocoles de localisation sera présentée.

2. Serveur à base d'agents mobiles pour la localisation

Comme nous nous intéressons dans cette thèse à la problématique de la localisation des agents mobiles dans les réseaux mobiles ad hoc et que l'approche de localisation proposée étant elle-même basée sur les agents mobiles, et afin de lever toute confusion entre les deux types d'agents, nous utiliserons l'appellation de code mobile pour signifier l'agent mobile à localiser. Nous parlerons alors de protocole de localisation du code mobile dans un réseau ad hoc, à l'aide d'un serveur basé sur la technologie d'agents mobiles.

2.1. Approche centralisée pour la localisation du code mobile dans un réseau ad hoc

Pour pallier l'absence d'infrastructure fixe dans les réseaux mobiles ad hoc, nous proposons d'utiliser un serveur virtuel centralisé, basé sur la technologie d'agent mobile, chargé de la gestion de la base de données contenant les informations sur la localisation des codes mobiles dans les réseaux ad hoc. Ainsi, le maintient de l'information de localisation se fait de manière proactive par le serveur.

Les étapes du protocole sont comme suit :

1) Calcul du centre de gravité du réseau

Le centre de gravité du réseau sera utilisé comme repère pour la sélection du serveur de localisation. Il détermine la région centrale du réseau mobile ad hoc où sera maintenu le serveur. Le calcul du centre de gravité du réseau à n nœuds, où (X_i, Y_i) sont les positions de chaque nœud i est comme suit :

$$Xv = (1/n) \sum_{t=1}^{n} Xi \qquad Yv = (1/n) \sum_{t=1}^{n} Yi$$

Où (X_v, Y_v) définissent la position du centre de gravité.

Notons que dans notre implémentation, les positions des nœuds dans le réseau sont fournies par le simulateur utilisé. En pratique, il est supposé que les nœuds du réseau sont équipés du système GPS ou bien qu'ils utilisent un autre service de positionnement. Notons aussi que dans notre expérimentation, le calcul du centre du réseau est réalisé en utilisant une fonction définie dans le simulateur utilisé. Ce dernier, mettant à disposition les positions des nœuds à tout instant, facilite la tâche du calcul et de la mise à jour du centre de gravité par la fonction ainsi définie. C'est à l'encontre de la réalité, puisque c'est les nœuds du réseau qui devraient le faire de manière distribuée, vu l'absence d'infrastructure fixe. En suivant une structure hiérarchique pour le calcul du centre du réseau, comme dans l'étape d'initialisation du clustering, permet de réduire le coût de ce traitement en termes de nombre de messages (voir figure 53).

2) Sélection du serveur de localisation

Choisir le nœud mobile le plus proche du centre de gravité du réseau, en tant que serveur de localisation (figure 55).

Le calcul de la distance d'un nœud *Ni* qui le sépare du centre de gravité *V*, pour l'élection de celui qui est le plus proche du centre, est comme suit :

$$Dist(Ni, V) = \sqrt{(Xv - Xi)^2 + (Yv - Yi)^2}$$

En cas de rapprochement de plusieurs nœuds du centre du réseau, on pourra opter pour d'autres critères pour désigner celui qui est le plus apte à devenir serveur du réseau. Par exemple, celui qui possède la capacité énergétique la plus importante ou celui qui a une vitesse de mobilité la plus faible ou bien celui avec un degré de connectivité le plus élevé.

3) Initialisation du serveur

Un agent mobile est crée et placé sur le nœud sélectionné, qui aura comme tâche la gestion de la base de données des positions (i.e., maintenir les informations de localisation). Logiquement, le lancement de l'agent mobile pour la première fois sera fait par le nœud sélectionné en tant que serveur.

4) Maintenance de l'information de localisation

A chaque fois qu'un code mobile migre dans le réseau, il envoie sa nouvelle position au serveur de localisation. L'information de localisation d'un code mobile est exprimée par son identité, ainsi que celle du nœud qui l'héberge. Connaissant le centre du réseau, le nœud d'accueil du code mobile transmet la position du code en direction du centre du réseau (figure 56). Ceci pour optimiser le flux d'informations dans le réseau, nécessaire à faire parvenir la mise à jour de l'information de localisation au serveur. Un protocole de routage géographique des informations convient parfaitement dans ce cas.

5) Gestion du serveur

Lorsque le nœud mobile hébergeant le serveur s'éloigne du centre de gravité ou pour une raison d'épuisement de batterie, ou lors d'une déconnexion volontaire du nœud ou bien lorsque le degré de connectivité du serveur devient très faible, un autre nœud est sélectionné pour son remplacement. La technique de sélection est la même que celle de la deuxième étape.

6) Migration de l'agent

L'agent placé sur le nœud remplacé migre vers le nœud sélectionné (figure 57), où il poursuivra la gestion de la base de données. Ainsi, la disponibilité du serveur, placé au centre du réseau, est assurée par la mobilité de l'agent.
Ce changement de support physique pour le serveur permet aussi d'équilibrer la charge de la fonction de gestion de la localisation entre les nœuds du réseau, particulièrement en termes de ressources énergétiques.

7) Processus de localisation

La localisation du code mobile se fera alors en envoyant une requête au serveur mobile, qui en consultant la base de données, donnera une réponse en envoyant un message transportant la position recherchée. L'envoie de la requête de localisation par un client, se fait en direction du centre du réseau, de la même manière que dans l'étape 4. Ceci devrait optimiser aussi le temps de réponse à la requête du client.

Un exemple très simple de calcul hiérarchique du centre de gravité est montré sur la figure 53. Comme dans les techniques de clustering, les nœuds voisins à un seul saut (ou plus) constituent un groupe et élisent le nœud qui calcul leur centre de gravité local. Par exemple, celui qui a le plus petit identificateur est élu en tant que chef de groupe pour cette fonction. Ensuite, la valeur du centre de gravité local à un groupe, pondérée par le nombre de nœuds dans ce groupe, est transmise vers les autres chefs de groupe. Ainsi de suite, jusqu'au calcul final du centre de gravité pour tout le réseau. Dans l'exemple de la figure 53, c'est le nœud ***s*** qui détient la valeur finale du centre qu'il doit diffuser dans le réseau.

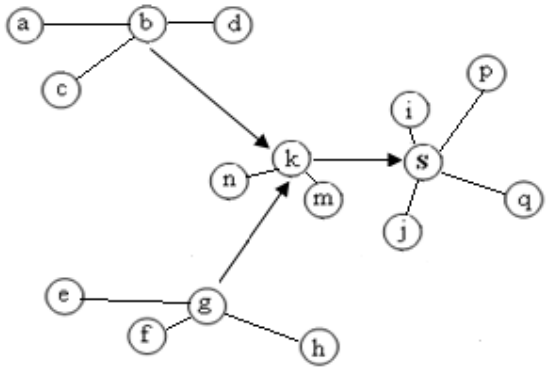


Figure 53 : *Exemple de calcul du centre de gravité de façon hiérarchique.*

Pour éviter un déséquilibre du centre du réseau suite au changement de topologie du réseau, la réactualisation du centre de gravité devrait se faire périodiquement. La période de la mise à jour est déterminée en fonction de plusieurs paramètres, à savoir le degré de mobilité des nœuds, le nombre de nœuds dans le réseau et l'étendue du réseau (densité du réseau). Par exemple, suite aux mouvements d'un grand nombre de nœuds, le centre de gravité de la figure 54 change. Il faudra donc le recalculer et faire migrer l'agent représentant le serveur, vers le nouveau nœud sélectionné pour héberger l'agent.

Figure 54 : *Centre de gravité du réseau à réactualiser.*

Dans la figure 55 par exemple, le nœud ***s*** qui est sélectionné pour héberger l'agent mobile, est désigné comme un serveur du réseau et les nœuds ***k, n et m***, qui sont positionnés près du centre de gravité du réseau, sont candidats pour remplacer ***s*** lorsqu'il s'éloigne du centre du réseau ou bien manque de capacité énergétique.

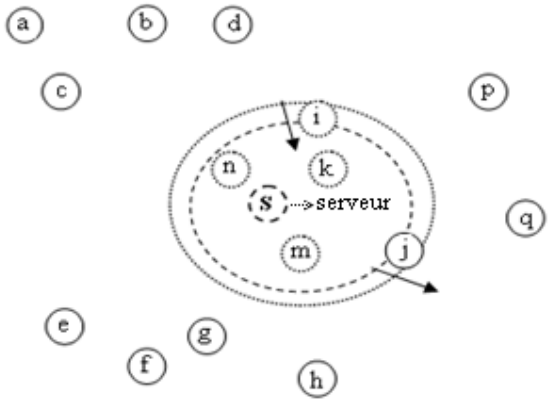


Figure 55 : *Un serveur mobile dans un réseau mobile ad hoc.*

Dans la figure 56 et après migration du code du nœud ***h*** vers ***e***, celui-ci envoie une information de localisation du code mobile reçu, en direction du centre du réseau où devrait se trouver le serveur de localisation.

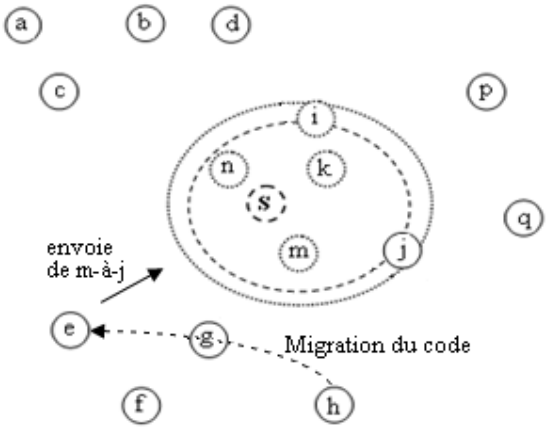


Figure 56 : *Mise à jour de l'information de localisation après migration du code.*

La figure 57 montre le cas où un nœud hébergeant le serveur est sur le point de s'éloigner du centre du réseau. Le nœud ***s*** en question, lance un appel aux nœuds voisins afin de sélectionner un nouveau candidat pour le remplacer et qui recevra l'agent mobile. Dans l'exemple, les deux nœuds ***k*** et ***m*** sont presque à la même distance du centre. Ayant plus de voisins que ***m***, c'est le nœud ***k*** qui est choisit.

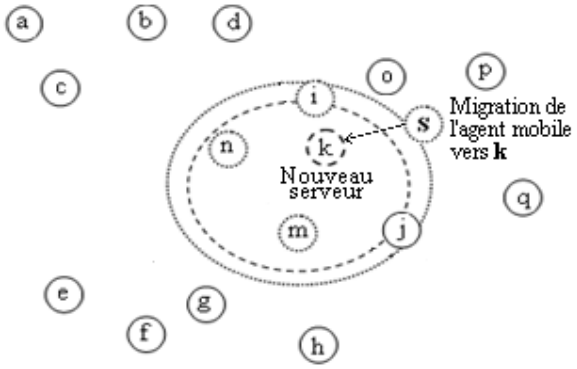


Figure 57 : *Changement de support physique pour le serveur.*

Notons que dans cette première variante de l'approche centralisée, un seul nœud du réseau sur lequel est placé un agent mobile, est utilisé pour constituer le serveur de localisation. Ceci n'est pas approprié dans un réseau ad hoc, car cela pourrait conduire à la formation d'un goulet d'étranglement et à l'épuisement des ressources énergétiques du seul nœud centralisant. Il est plus intéressant de déléguer d'autres agents, placés sur d'autres nœuds prés du centre du réseau, et qui auront comme tâches de coopérer d'une façon ou d'une autre avec l'agent central, afin de le décharger un peu de la fonction de localisation et permettre une tolérance aux fautes en cas de déconnexion involontaire du nœud hébergeant le serveur. Par exemple, chaque agent délégué peut maintenir une copie de la base de données des positions pour se prémunir des cas de pannes, ou bien gérer des parties de la base de données de localisation (sous base de données).

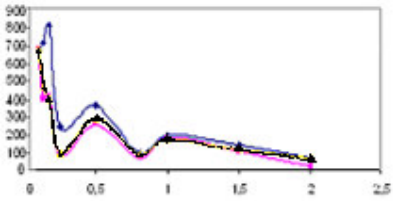
2.1.1. Protocole centralisé versus protocole distribué : une étude comparative

Dans le but d'étudier les performances de l'approche centralisée, nous avons réalisé une étude comparative de cette approche avec l'approche distribuée décrite dans le chapitre précédent. Comme le protocole réactif de l'approche distribuée est de loin le moins couteux et le plus adapté aux réseaux mobiles ad hoc par rapport au protocole proactif, notre étude comparative a été faite entre l'approche réactive du protocole distribué et le protocole centralisé.

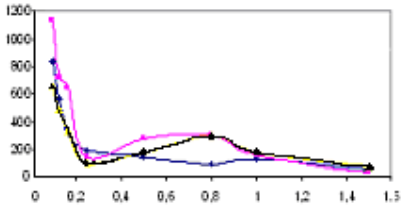
Notons que les résultats d'expérimentation présentés ci-dessous de l'implémentation du protocole centralisé, concernent le cas de la localisation d'un seul code mobile qui migre dans le réseau ad hoc. L'environnement de simulation est le même que celui décrit dans le chapitre précédent. Notons aussi que le centre de gravité n'est pas mis à jour pendant toute la durée de la simulation.

2.1.1.1. Résultats expérimentaux et interprétations

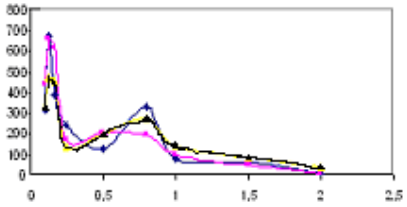
Les figures 58 et 59 montrent respectivement les performances du protocole centralisé en termes de nombre de messages moyen et de temps de réponse moyen par requête de localisation, par rapport à la charge des requêtes, à la mobilité des nœuds et à celle du code.



a- Mobilité élevé du code



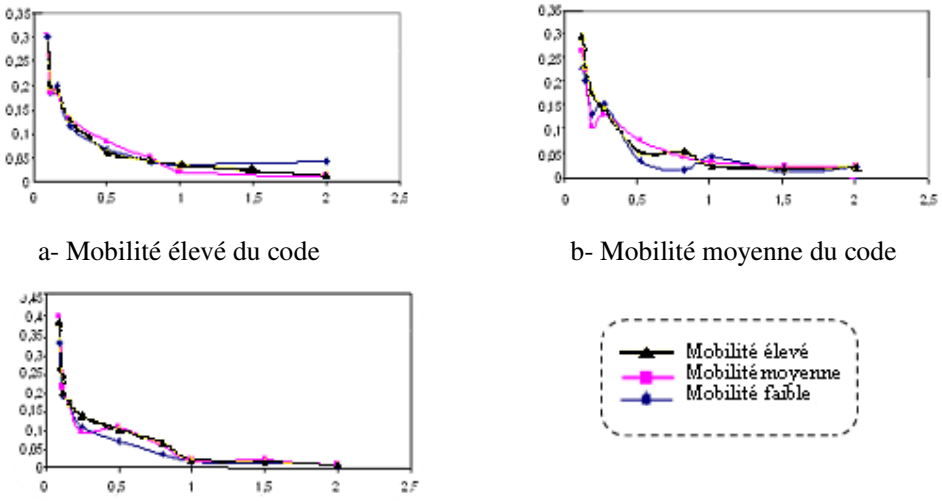
b- Mobilité moyenne du code



c – Mobilité faible du code

Figure 58: *Nombre de messages moyen par rapport à la charge des requêtes.*

Le nombre de messages moyen échangés par requête de localisation est inversement proportionnel à la charge. Lorsque la charge est importante, le nombre de migration du code entre deux requêtes de localisation successives est minime. Ce qui explique que la localisation ne consomme pas beaucoup de messages dans ce cas. Notons aussi que le nombre de message croit avec l'augmentation du degré de mobilité du code. Plus le code est mobile, plus le coût de maintenance des informations de localisation au niveau du serveur devient élevé en termes de messages.



a- Mobilité élevé du code

b- Mobilité moyenne du code

c – Mobilité faible du code

Figure 59 : *Temps de réponse moyen par rapport à la charge des requêtes.*

Comme pour le nombre de message, le temps réponse moyen est inversement proportionnel à la charge et il est proportionnel au degré de mobilité du code. Les explications données sont les même que pour le paramètre précédent (nombre de message).

Nous remarquons que le protocole distribué (figure 49) est beaucoup moins coûteux en termes de nombre de messages (30 messages au maximum) que le protocole centralisé (plus de 700 messages). Ceci est dû principalement aux messages de diffusion et de contrôle nécessaires à la mise à jour de la base de données de localisation au niveau du serveur de façon proactive, et dû à la gestion du serveur mobile, ce qui n'est pas le cas pour l'approche distribuée. Nous remarquons aussi que le temps de réponse est meilleur pour le protocole distribué (figure 51) par rapport au protocole centralisé. Une partie du grand nombre de messages échangés dans l'approche centralisée, concerne la région du serveur, ce qui provoque sa surcharge ; ceci réduit les performances du serveur en termes de temps de réponse pour les requêtes de localisation.

2.1.1.2. Discussion

L'étude comparative entre les deux protocoles montre que le protocole distribué est plus performant que le protocole centralisé, particulièrement en termes de nombre de message. La mise à jour de manière proactive de la base de données de localisation au niveau du serveur et la gestion de sa mobilité, génère un grand nombre de messages de diffusion et de contrôle, ce qui surcharge le serveur. Par conséquent, les performances du protocole « purement » centralisé n'ont pas été améliorées en termes de temps de réponse, malgré la disponibilité du serveur qui est censé satisfaire dans de meilleurs délais les requêtes de localisation que dans le cas du protocole distribué.

Comme cela a été prévisible d'après la littérature, les approches totalement centralisées sont inadaptées dans un réseau ad hoc caractérisé par une bande passante limitée. Effectivement, elles ont tendance à saturer la région centrale et former ainsi un goulet

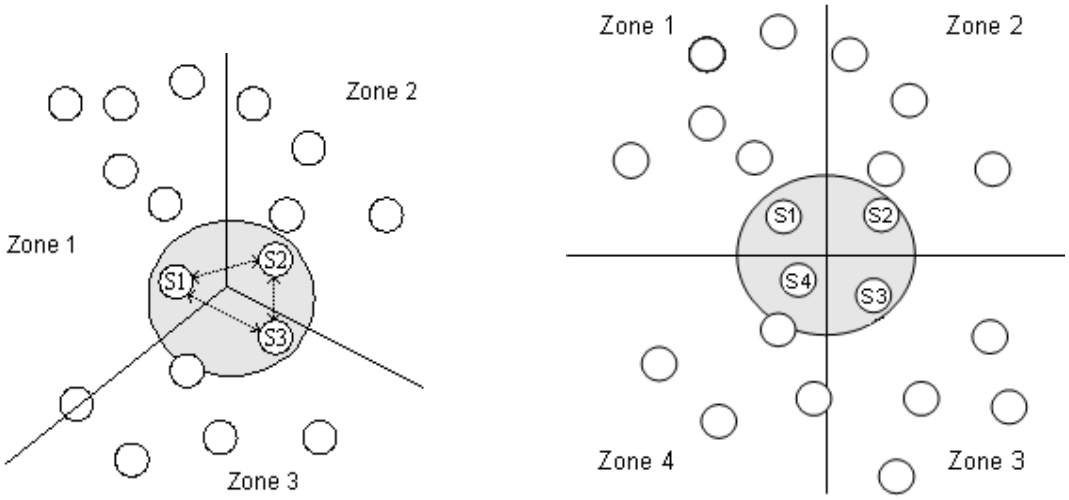
d'étranglement au niveau du serveur, ce qui épuiserait la capacité énergétique de la station supportant le serveur. Ajouter à cela, l'inconvénient classique de la panne ou la déconnection involontaire du serveur, affectant la disponibilité du service en question si aucun traitement n'est prévu à cet effet.

Après avoir analysé les causes de cette dégradation des performances de l'approche centralisée, nous proposons une distribution du protocole centralisé, décrite et détaillée dans le prochain paragraphe.

2.2. Protocole de localisation multi-serveurs : une distribution de l'approche centralisée

La distribution de l'approche centralisée consiste en une gestion multi-serveurs du problème de localisation, avec une structuration du réseau ad hoc en plusieurs zones. L'objectif est d'affecter à chaque serveur la tâche de localisation du code mobile dans une certaine région géographique, une zone. A cet effet, chaque serveur doit maintenir une sous base de données de localisation des codes mobiles migrants dans sa zone. L'idée principale de cette distribution, est de limiter l'étendue des diffusions liées au processus de localisation à la seule zone concernée, évitant ainsi la surcharge du serveur de localisation.

Comme dans le cas du protocole centralisé, le protocole de localisation multi-serveurs est basé sur la technologie d'agent mobile. Chaque serveur est représenté par un agent mobile. L'ensemble des agents mobiles constituent un système multi-agent mobile qui forme le serveur virtuel du réseau. Chaque agent du système multi-agent est appelé à coopérer avec les autres agents pour compléter la fonction de localisation (voir figure 60-a).



(a) *Cas de répartition en trois zones.* **(b)** *Cas de répartition en quatre zones.*

Figure 60: *Répartition du réseau en zones*

Les exemples (a) et (b) de la figure 60, montrent les cas de structuration du réseau en respectivement trois et quatre zones, avec les nœuds ***Si*** comme serveurs des zones ***i***, hébergeant chacun un agent mobile qui assure la fonction de localisation.

2.2.1. Description du protocole

Les étapes du protocole de localisation multi-serveurs sont comme suit :

1) Détermination du centre de gravité du réseau

Le centre de gravité sera utilisé comme repère pour la définition des différentes zones du réseau et pour la sélection des serveurs associés à chaque zone. Le calcul du centre de gravité du réseau est effectué de la même manière que dans le cas du protocole purement centralisé.

2) Description d'une zone

Chaque zone est représentée par un espace géographique délimité par deux axes B1 et B2 (figure 61), ayant comme intersection, le centre de gravité du réseau et formant un angle α tel que $0 < \alpha <= \pi$

En supposant que toutes les zones ont la même étendue géographique, on aura $\alpha = 2\pi / n$, n étant le nombre de zones dans le réseau, avec $n >= 2$

3) Affectation des nœuds aux zones du réseau

Soient deux angles $\beta 1$ et $\beta 2$ associés respectivement aux deux axes B1 et B2 délimitant une zone donnée (voir figure 61), tel que : $\alpha = \beta 2 - \beta 1$, avec $\beta 2 > \beta 1$

Alors, un nœud i appartient à une zone donnée, délimitée par B1 et B2, si ses positions (Xi, Yi) vérifient les deux équations suivantes :

$$(Y_i - Y_v)\cos(\beta 1) > (X_i - X_v)\sin(\beta 1)$$
$$(Y_i - Y_v)\cos(\beta 2) < (X_i - X_v)\sin(\beta 2)$$

Où (X_v, Y_v) sont les positions du centre de gravité du réseau.

4) Election d'un chef de zone en tant que serveur et coopération

Dans chaque zone du réseau, élire (ou sélectionner) un nœud le plus proche du centre de gravité qui aura le rôle de serveur de cette zone (chef de groupe).

Le calcul de la distance séparant un nœud du centre du réseau est effectué de la même manière que dans le protocole totalement centralisé.

L'élection du serveur de zone peut être faite sous différents critères de sélection, tels que l'énergie, le degré de connectivité ou de mobilité des nœuds. Nous avons utilisé la proximité du centre de gravité comme critère de sélection, pour permettre un accès rapide aux serveurs, avec une disponibilité la plus longue possible, vu qu'ils sont supposés être positionnés tout le temps au centre du réseau. Ceci permettrait aussi d'assurer une coopération rapide et efficace entre les serveurs de zones, puisqu'ils sont proches l'un de l'autre autour du centre de gravité du réseau.

La coopération entre les différents serveurs est nécessaire pour localiser les codes mobiles à l'extérieur de leur zone d'une part, et de mettre à jour la base de données de localisation lorsque le code mobile ou le nœud l'hébergeant change de zone d'autre part.

5) Utilisation des agents mobiles et migration

Un agent mobile est placé sur chaque nœud élu en tant que serveur de zone, qui aura comme tâche la gestion d'une sous bases de données contenant les positions des codes mobiles à l'intérieur de sa zone. La position d'un code mobile est exprimée par le couple suivant : {identificateur_code, identificateur_noeud}
Quand un nœud mobile hébergeant un serveur s'éloigne du centre du réseau ou pour les différentes raisons citées dans le protocole totalement centralisé, un autre nœud est élu pour son remplacement. Alors l'agent placé sur le nœud remplacé migre vers le nœud élu, où il va poursuivre sa gestion de la base de données de localisation.

6) Maintenance de la base de données de localisation

La mise à jour de la base de données se fait à chaque migration du code :

- Lorsque le code migre vers un nœud dans la même zone, sa nouvelle position est mise à jour dans la sous base de données du serveur de cette zone (figure 62).
- Si le nœud destinataire du code mobile se trouve dans une autre zone, alors sa position sera insérée dans la sous base de données du serveur de la zone de destination, et supprimée de celle du serveur de la zone quittée (figure 63). La mise à jour de la sous base de données pour cette opération de suppression, se fait par la coopération entre les serveurs concernés.
- Lorsqu'un nœud hébergeant un code mobile change de zone, la sous base de données sera mise à jour de la même manière que lorsque le code migre vers un nœud d'une autre zone (figure 64).

7) Processus de localisation

La localisation du code mobile sera effectuer en envoyant une requête au serveur de zone, qui en consultant sa sous base de données, donne l'identification du nœud hébergeant le code mobile. S'il ne trouve pas une telle information, le serveur transmet la requête aux serveurs voisins, qui traiteront la requête de la même manière.

Une fois l'identité du nœud hébergeant le code mobile est connue, la localisation du nœud dans le réseau aura lieu avec l'un des services de localisation des nœuds, tel que GLS.

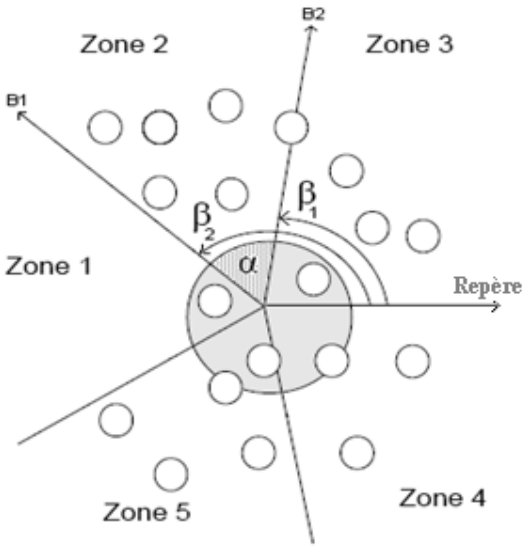


Figure 61: *Description d'une zone géographique.*

La mise à jour de l'information de localisation lorsqu'il y a migration d'un code entre nœuds de la même zone (intra-zone), est effectuée par le serveur de cette zone, comme le montre la figure 62. Dans ce cas, l'étendue des informations de contrôle se limite seulement à la zone en question.

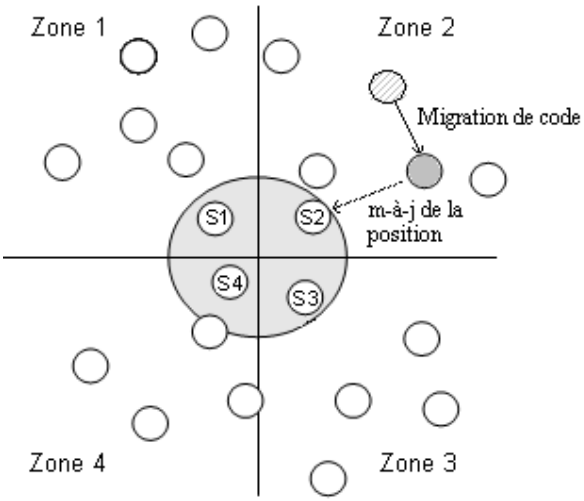


Figure 62 : *Mise à jour intra-zone.*

Comme le montre la figure 63, lorsqu'un code mobile migre d'une zone à une autre (inter-zones), le processus de maintenance de l'information de localisation concernera les serveurs des deux zones en question.

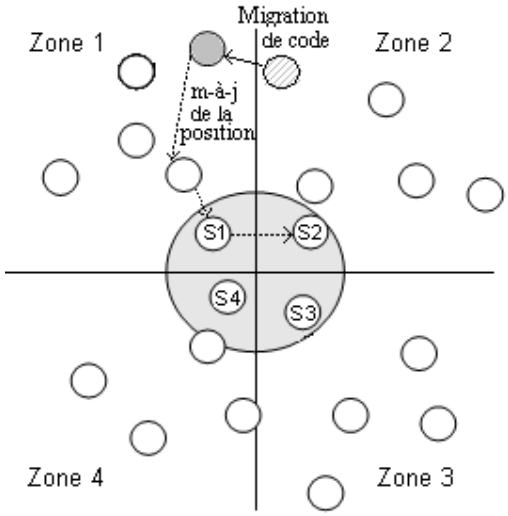


Figure 63 : *Mise à jour inter-zones.*

Dans la figure 64, on montre la mise à jour d'une information de localisation lorsqu'un nœud mobile hébergeant un code mobile, change de zone. La mise à jour se fait de la même manière que dans le cas précédent.

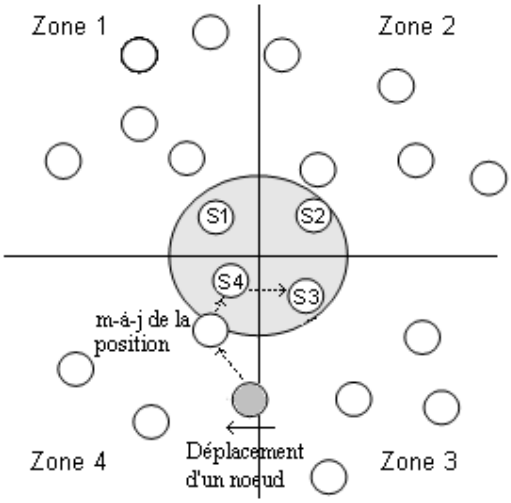


Figure 64 : *Changement de zone pour un nœud mobile.*

La réactualisation du centre de gravité doit se faire périodiquement, comme dans le cas du protocole totalement centralisé. Elle peut être aussi déclenchée lorsque le nombre de nœuds dans une zone est beaucoup plus petit ou plus grand d'une moyenne donné. En effet, le nombre de nœuds présents dans chacune des zones du réseau, pourrait renseigner sur la validité du centre de gravité en cours. Pour optimiser le coût de calcul du centre du réseau, nous proposons une technique de calcul approximatif du centre, basée seulement sur les nœuds du réseau positionnés sur le périmètre du réseau. Le centre calculé pour le réseau sera le centre de gravité de ces nœuds, caractérisés par un voisinage restreint et particulier (figure 65). Ceci éviterait d'impliquer tous les nœuds du réseau dans le processus de réactualisation du centre du réseau, permettant ainsi d'optimiser le nombre de messages échangés.

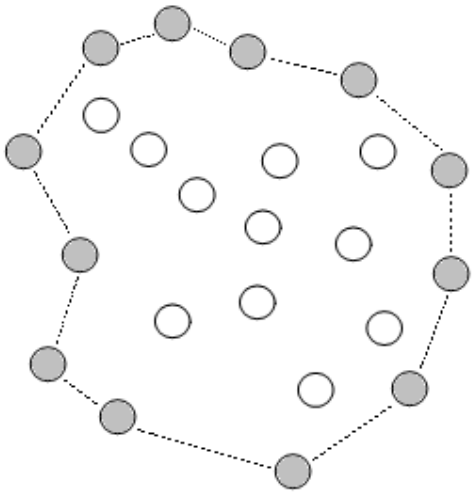


Figure 65 : *Calcul d'un centre de gravité approximatif à partir des nœuds situés sur le périmètre du réseau.*

Notons que ce protocole suppose qu'il n'y a pas de partitionnement dans le réseau, i.e., que tous les nœuds du réseau appartiennent à une même et unique partition. De plus, nous n'avons pas traité les cas de pannes ou de déconnections involontaires des nœuds du réseau.

2.2.2. Résultats expérimentaux et interprétations

Les résultats d'expérimentation présentés ci-dessous, de l'implémentation du protocole centralisé multi-serveurs, concernent le cas de la localisation de plusieurs codes mobiles qui migrent en même temps dans le réseau ad hoc et indépendamment l'un de l'autre, ce qui est plus réaliste par rapport aux résultats précédents. De plus, pour un nombre important de zones dans le réseau, cette étude de performances du protocole a pris en compte la taille du réseau. Pour une même superficie du réseau, ce paramètre renseigne aussi sur le degré de connectivité des nœuds.

2.2.2.1 Nombre de message moyen

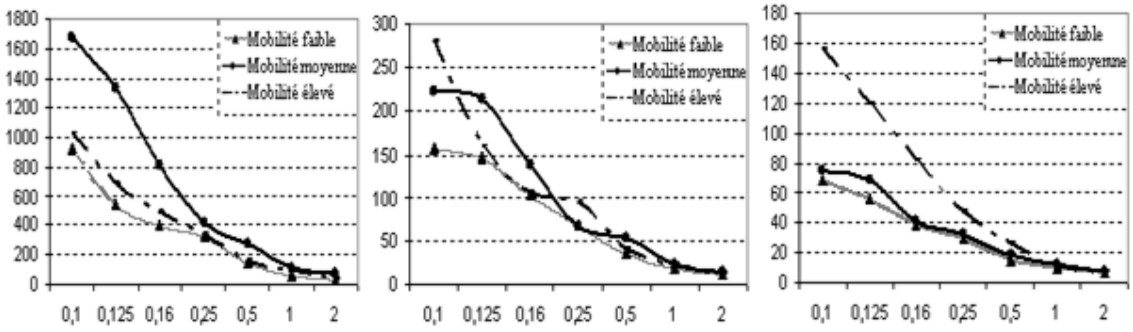
a) Cas d'un réseau avec une taille faible

Les figures 66, 67, 68 et 69 montrent respectivement les performances du protocole centralisé avec une répartition du réseau en une, deux, quatre et sept zones, en termes de nombre de messages moyen par requête de localisation, par rapport à la charge des requêtes, à la mobilité des nœuds et à celle du code. Les quatre figures présentent les performances du protocole dans le cas d'un réseau avec une taille faible (25 nœuds).

Nous remarquons sur les différentes figures que le nombre de messages moyen échangés devient plus important lorsque le degré de mobilité du code augmente. C'est tout à fait logique puisque une mobilité du code élevé induit une augmentation des messages de maintenance de l'information de localisation. De même, le nombre de messages est proportionnel par rapport au degré de mobilité des nœuds. Une forte mobilité des nœuds hébergeant les codes mobiles et ceux supportant les serveurs, implique dans certains cas, un

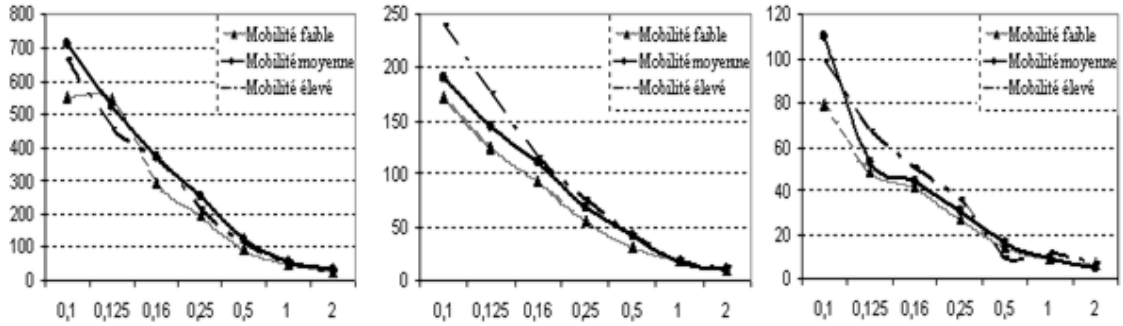
échange de messages pour le maintient de l'information de localisation et pour le maintient du serveur au centre du réseau.

Nous remarquons aussi que le nombre de messages moyen échangés diminue avec l'augmentation du nombre de zones dans le réseau. Les performances en termes de nombre de messages sont donc améliorées avec la distribution du protocole centralisé et ceci en limitant l'étendue des diffusions pendant le processus de localisation à la seule zone concernée, évitant ainsi la surcharge des serveurs mobiles. Notons aussi que le nombre de messages moyen échangés par requête de localisation est inversement proportionnel à la charge. La même explication est donnée que dans le cas du protocole totalement centralisé.



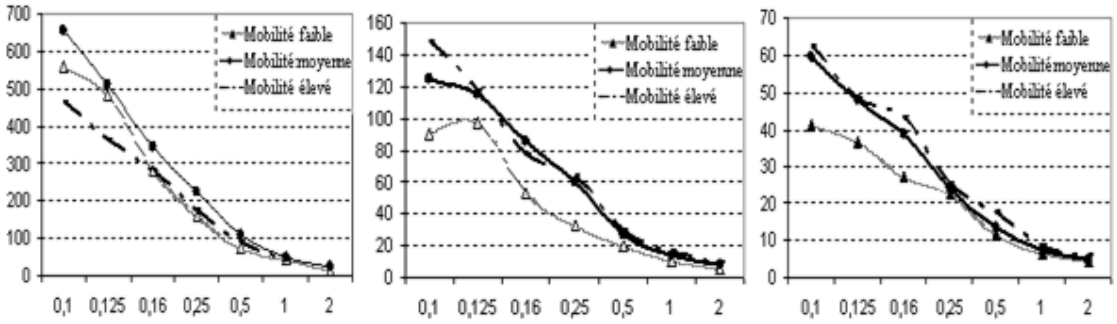
a) Mobilité élevé du code. b) Mobilité moyenne du code. c) Mobilité faible du code.

Figure 66 : *Nombre moyen de messages en fonction de la charge des requêtes : Cas d'une seule zone avec une taille faible du réseau.*



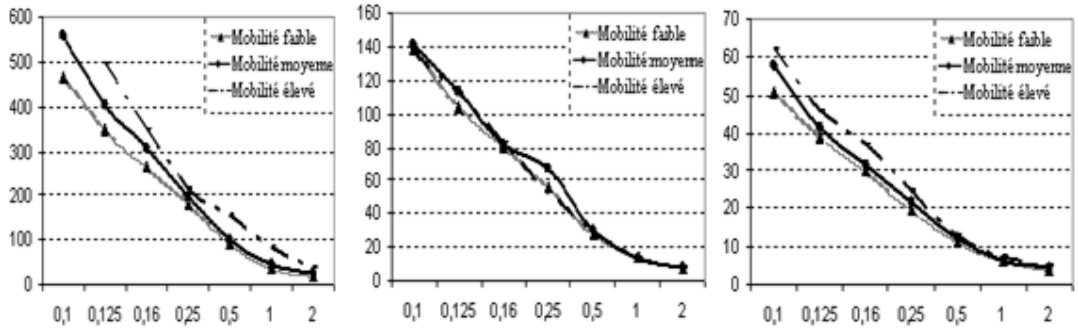
a) Mobilité élevé du code. b) Mobilité moyenne du code. c) Mobilité faible du code.

Figure 67 : *Nombre moyen de messages en fonction de la charge des requêtes : Cas de deux zones et taille faible du réseau.*



a) Mobilité élevé du code. b) Mobilité moyenne du code. c) Mobilité faible du code.

Figure 68 : *Nombre moyen de messages en fonction de la charge des requêtes : Cas de quatre zones et taille faible du réseau.*

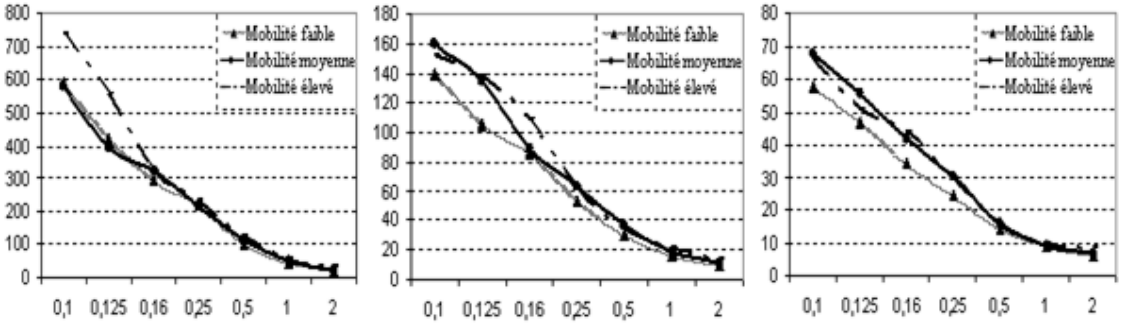


a) Mobilité élevé du code. b) Mobilité moyenne du code. c) Mobilité faible du code.

Figure 69 : *Nombre moyen de messages en fonction de la charge des requêtes : Cas de sept zones et taille faible du réseau.*

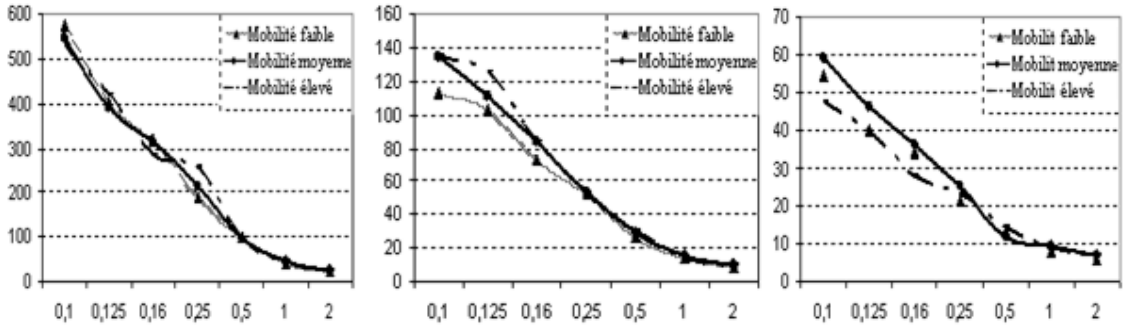
b) Cas d'un réseau avec une taille moyenne

Les figures 70 et 71 montrent l'étude des performances du protocole dans le cas d'un réseau avec une taille moyenne (50 nœuds) selon les mêmes critères que précédemment, et qui a été faite seulement pour le cas d'une répartition du réseau en quatre et sept zones.



a) Mobilité élevé du code. b) Mobilité moyenne du code. c) Mobilité faible du code.

Figure 70 : *Nombre moyen de messages en fonction de la charge des requêtes : Cas de quatre zones et taille moyenne du réseau.*



a) Mobilité élevé du code. b) Mobilité moyenne du code. c) Mobilité faible du code.

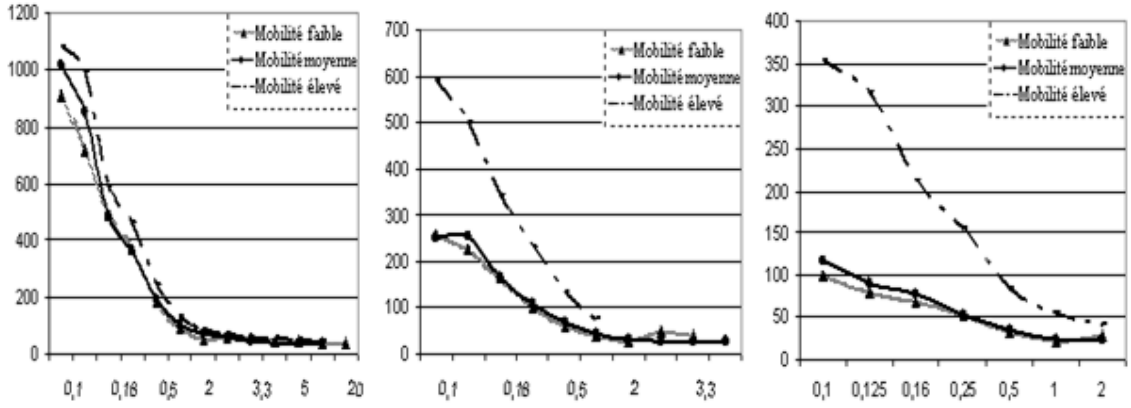
Figure 71 : *Nombre moyen de messages en fonction de la charge des requêtes : Cas de sept zones et taille moyenne du réseau.*

Nous remarquons que les performances du protocole sont améliorées, en termes de nombre de messages moyen échangés, avec l'augmentation du nombre de zones dans le réseau.

Nous remarquons aussi que le nombre de messages moyen échangés ne change pas beaucoup par rapport au cas d'un réseau avec une taille faible. Lorsque la taille du réseau passe de 25 à 50 nœuds, le nombre total des messages échangés dans le réseau augmente avec un taux presque similaire à celui du nombre des requêtes de localisation. Ce qui résulte en une moyenne des messages échangés presque inchangée.

c) Cas d'un réseau avec une grande taille

La figure 72 montre l'étude des performances du protocole dans le cas d'un réseau avec une grande taille (200 nœuds) selon les mêmes critères que précédemment, et qui a été faite seulement dans le cas d'une répartition du réseau en sept zones.



a) Mobilité élevé du code. b) Mobilité moyenne du code. c) Mobilité faible du code.

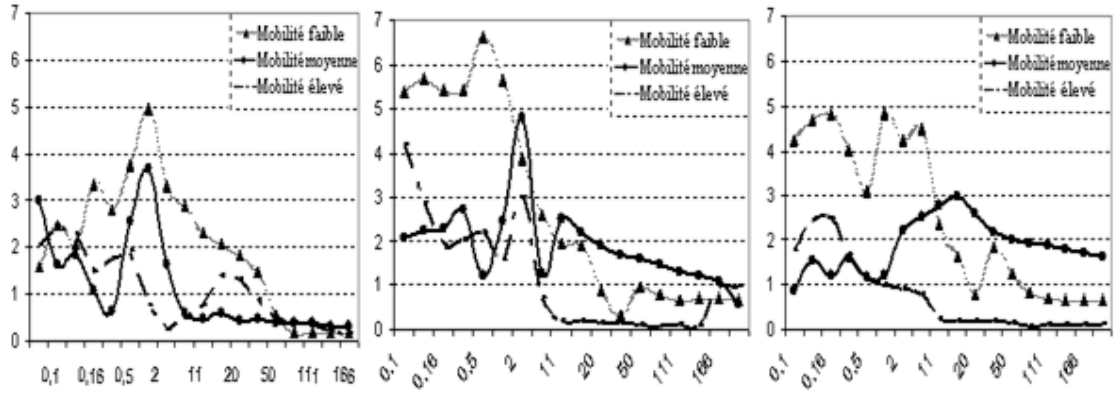
Figure 72 : *Nombre moyen de messages en fonction de la charge des requêtes : Cas de sept zones et grande taille du réseau.*

Les résultats de la figure 72 montrent qu'une taille très importante pour le réseau influe négativement sur les performances du protocole en termes de messages moyen échangés, notamment lorsque les nœuds du réseau sont très mobiles. Dans un réseau de grande taille, le nombre de messages de contrôle générés pour les traitements liés à la mobilité du code et pour la gestion de la mobilité des nœuds à travers les zones du réseau, devient plus important que le nombre de requêtes de localisation lancés.

2.2.2.2. Temps de réponse moyen

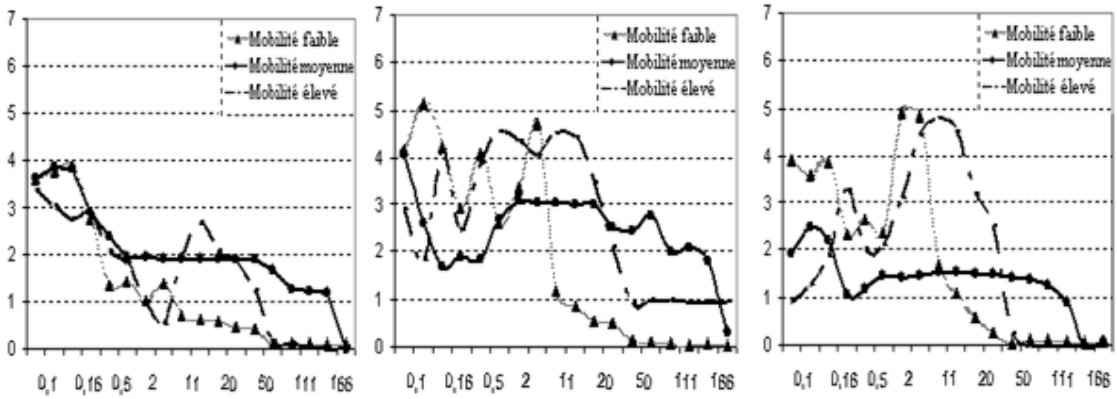
a) Cas d'un réseau avec une taille faible

Les figures 73, 74, 75 et 76 montrent respectivement les performances du protocole centralisé avec une répartition du réseau en une, deux, quatre et sept zones, en termes de temps de réponse moyen par requête de localisation, par rapport à la charge des requêtes, à la mobilité des nœuds et à celle du code. Les quatre figures présentent les performances du protocole dans le cas d'un réseau avec une taille faible (25 nœuds).



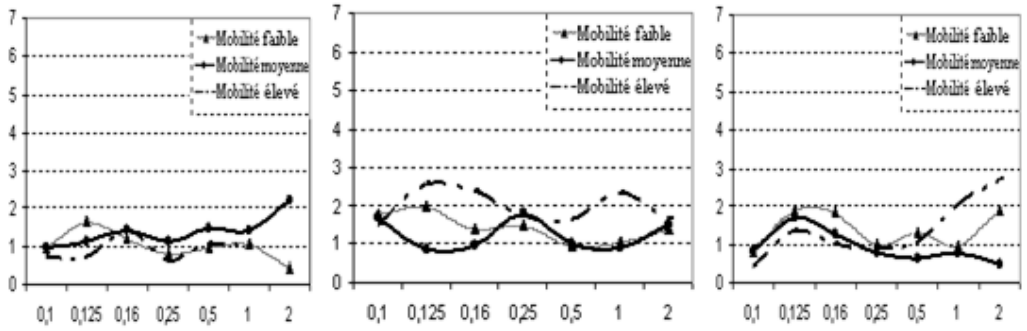
a) Mobilité élevé du code. b) Mobilité moyenne du code. c) Mobilité faible du code.

Figure 73 : *Temps de réponse moyen en fonction de la charge des requêtes : Cas d'une seule zone avec une taille faible du réseau.*



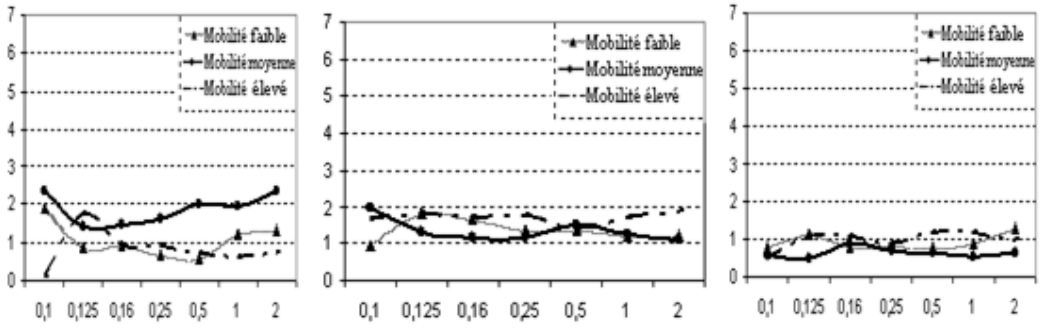
a) Mobilité élevé du code. b) Mobilité moyenne du code. c) Mobilité faible du code.

Figure 74 : *Temps de réponse moyen en fonction de la charge des requêtes : Cas de deux zones et taille faible du réseau.*



a) Mobilité élevé du code. b) Mobilité moyenne du code. c) Mobilité faible du code.

Figure 75 : *Temps de réponse moyen en fonction de la charge des requêtes : Cas de quatre zones et taille faible du réseau.*



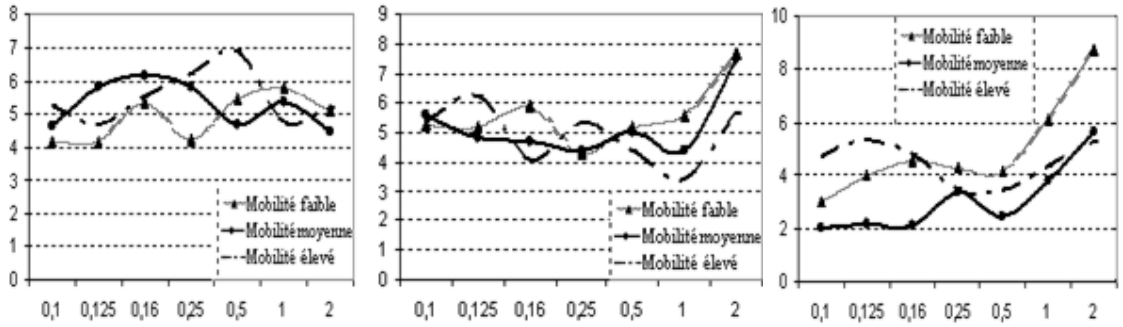
a) Mobilité élevé du code. b) Mobilité moyenne du code. c) Mobilité faible du code.

Figure 76 : *Temps de réponse moyen en fonction de la charge des requêtes : Cas de sept zones et taille faible du réseau.*

Comme pour le nombre de messages, le temps de réponse moyen est inversement proportionnel à la charge. Les mêmes explications sont données que pour le nombre de messages. Nous remarquons aussi que le temps de réponse a été amélioré avec la distribution du protocole centralisé (voir aussi la figure 80). La disponibilité de plusieurs serveurs non trop chargés, à l'inverse du cas d'un seul serveur, aussi bien que la coopération entre ces serveurs, a permis d'améliorer le temps de réponse dans le traitement des requêtes de localisation.

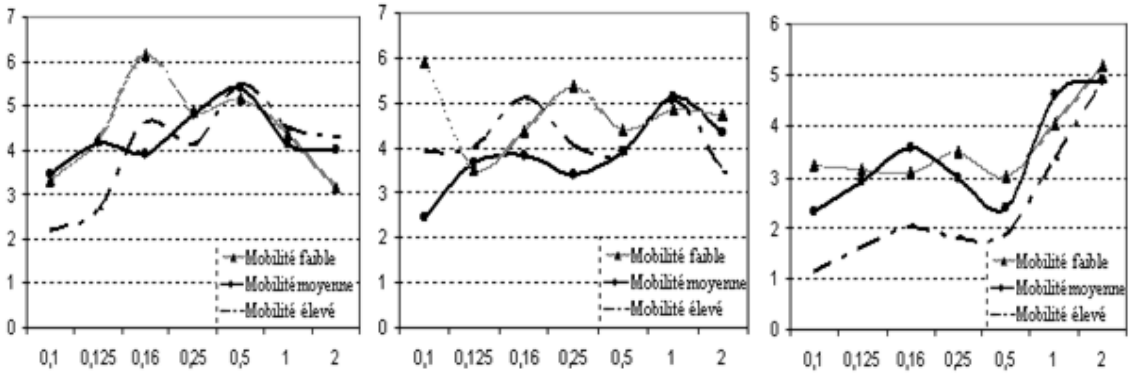
b) Cas d'un réseau avec une taille moyenne

Les figures 77 et 78 montrent l'étude des performances du protocole dans le cas d'un réseau avec une taille moyenne (50 nœuds) selon les mêmes critères que précédemment, et qui a été faite seulement dans le cas d'une répartition du réseau en quatre et sept zones.



a) Mobilité élevé du code. b) Mobilité moyenne du code. c) Mobilité faible du code.

Figure 77 : *Temps de réponse moyen en fonction de la charge des requêtes : Cas de quatre zones et taille moyenne du réseau.*



a) Mobilité élevé du code. b) Mobilité moyenne du code. c) Mobilité faible du code.

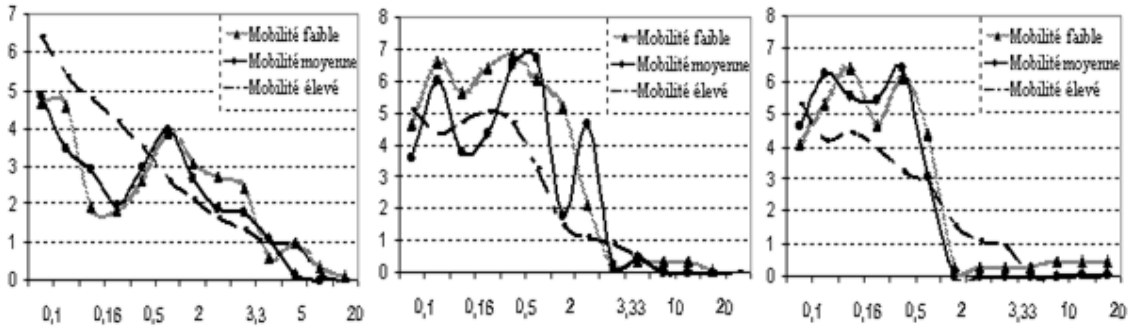
Figure 78 : *Temps de réponse moyen en fonction de la charge des requêtes : Cas de sept zones et taille moyenne du réseau.*

Nous remarquons que l'augmentation du nombre de zones dans le réseau, améliore le temps de réponse de localisation pour un réseau de taille moyenne. Les mêmes explications sont données que dans le cas d'un réseau de taille faible.

Nous remarquons aussi que le temps de réponse a augmenté par rapport au réseau de taille faible. L'augmentation du nombre de nœuds dans le réseau, et donc dans une zone, engendre plus de demande de localisation pour le serveur de zone, ce qui nécessite plus temps pour satisfaire ces requêtes.

c) Cas d'un réseau avec une grande taille

La figure 79 montre l'étude des performances du protocole dans le cas d'un réseau avec une grande taille (200 nœuds) selon les mêmes critères que précédemment, et qui a été faite seulement dans le cas d'une répartition du réseau en sept zones.



a) Mobilité élevé du code. b) Mobilité moyenne du code. c) Mobilité faible du code.

Figure 79 : *Temps de réponse moyen en fonction de la charge des requêtes : Cas de sept zones et grande taille du réseau.*

Nous remarquons que le temps de réponse augmente par rapport à un réseau de taille moyenne. Il est logique que le temps de réponse soit croissant par rapport à la taille du réseau, cependant nous remarquons que sa croissance devient moins importante lorsque le réseau passe d'une taille moyenne à une taille élevée (figure 81). Cette remarque qui concerne le cas de distribution en sept zones, incite à augmenter le nombre de zones lorsque la taille du réseau augmente.

La figure 80 montre le comportement du protocole centralisé en termes de temps de réponse lorsque le nombre de zones augmente dans le réseau.

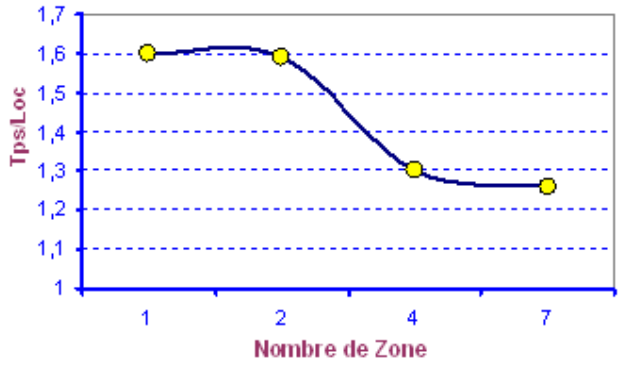


Figure 80 : *Influence du nombre de zones sur le temps de réponse moyen en fonction de la charge des requêtes de localisation.*

La figure 81 montre le comportement du protocole centralisé en termes de temps de réponse lorsque la taille du réseau augmente.

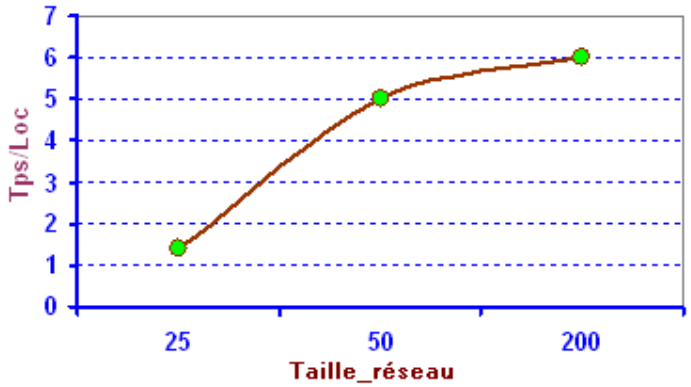


Figure 81 : *Influence de la taille du réseau sur le temps de réponse moyen en fonction de la charge des requêtes de localisation.*

Discussion

Les performances de notre protocole semi-centralisé, en termes de nombre de messages moyen et de temps de réponse, sont améliorées en fonction de la distribution des serveurs de zones. Cependant, les gains en performances ne sont pas très importants à partir de quatre zones. Lorsque le nombre de serveurs placés dans une même région centrale devient important, les surcoûts liés à la gestion de leur mobilité, entre autres, peuvent absorber le gain en performances lié à la distribution. Ajoutées à cela, les multiples sollicitations simultanées d'un grand nombre de serveurs pourraient engendrer une surcharge au niveau de la région centrale. Une autre proposition serait un redéploiement des serveurs, qui les éloignent de la région centrale, ce qui éviterait une telle saturation (voir figure 83).

Les performances du protocole sont dégradées en fonction de la taille du réseau (passage à l'échelle) dû à une surcharge au niveau de la région centrale. Par conséquent, le serveur multi-agents est ralenti par le flux très important de messages reçus dans le cas d'un nombre important de nœuds dans le réseau. Notre structure hiérarchique où les serveurs de zones sont centralisés, peut présenter ainsi un goulet d'étranglement dans la région centrale, pour les réseaux à grande échelle. Nous pensons qu'une augmentation du nombre de zones pour le réseau, ainsi qu'un redéploiement des serveurs correspondants (figure 83), pourraient réduire l'impact négatif du passage à l'échelle.

2.2.3. Structuration du réseau ad hoc en zones

Différentes techniques sont proposées dans la littérature pour la structuration des réseaux ad hoc en zones géographiques. Certaines organisent le réseau en plusieurs régions circulaires, de façon similaire aux réseaux cellulaires, d'autres se basent sur des régions rectangulaires ou carrées constituant une grille (figure 41). La structuration que nous avons proposé est basée sur le centre du réseau pour permettre, comme rapportée plus haut, une coopération rapide entre les différents serveurs d'une part, et d'assurer une disponibilité de services d'autre part. Notre approche possède des similitudes avec la technique appelée "œil de poisson" dans le protocole de routage FSR (*Fisheye State Routing*) [41] qui organise le

réseau en trois régions concentriques en fonction du nombre de sauts entre les nœuds, comme le montre la figure 82.

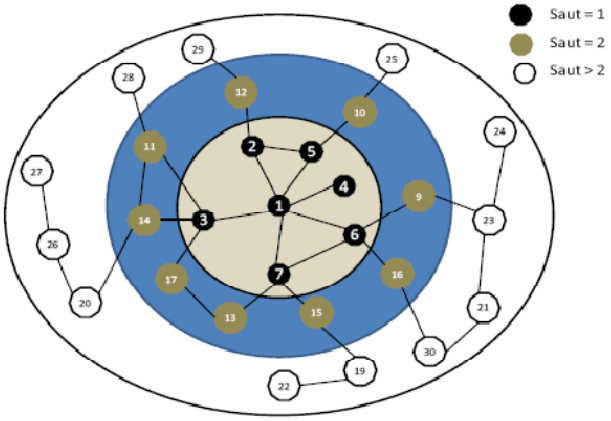
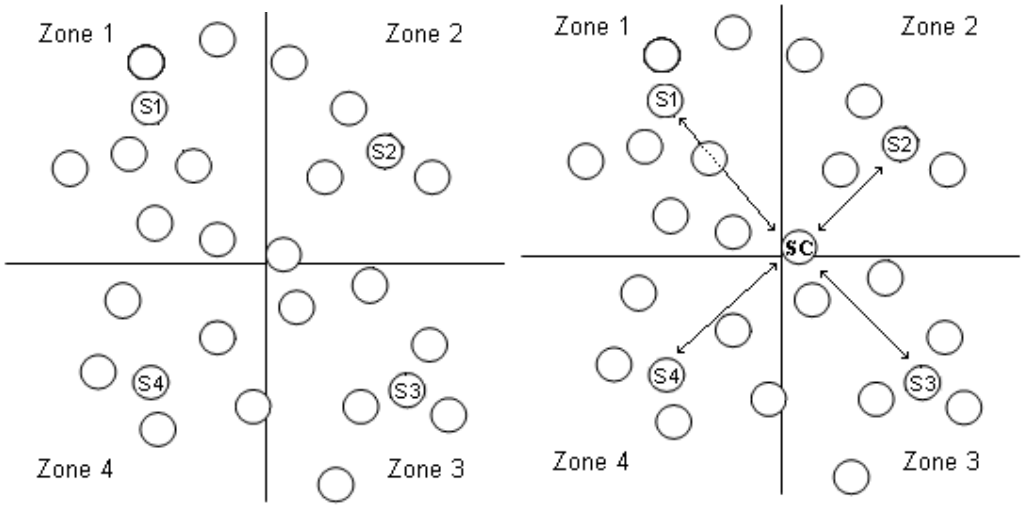


Figure 82 : *Technique "œil de poisson".*

Une autre proposition peut être faite pour le placement des serveurs de zone dans le réseau, lorsque la coopération entre ces serveurs n'est pas importante et que c'est l'activité des nœuds du réseau en direction des serveurs de zone qui devient la plus intense, ou bien pour éviter la surcharge de la région centrale pour les réseaux à grande échelle. Comme le montre la figure 83.a, les serveurs de zones sont plus près des nœuds de leur zone, mais pour coopérer, ils doivent utiliser des passerelles.

Cette réorganisation des serveurs peut exploiter l'approche centralisée en faisant appel à un serveur de serveurs placé dans la région centrale (figure 83.b). Ceci devrait leurs permettre une meilleur coopération et sans surcharger la région centrale. En plus du rôle de passerelle entre les serveurs de zones, le serveur central peut assurer d'autres fonctions, comme par exemple la sauvegarde de points de reprise pour les serveurs de zones en cas de panne.



a- Coopération distribuée entre serveurs. b- Coopération centralisée.

Figure 83 : *Serveur au centre de la zone.*

2.2.4. Localisation des agents mobiles du serveur

Connaissant la technique de structuration du réseau en un nombre donné de zones et l'existence d'un serveur de zone placé au centre du réseau, les nœuds n'ont pas à chercher à localiser l'agent mobile représentant le serveur ou à lancer une diffusion dans toute la zone, avant de lui envoyer une requête de localisation ou une information de mise à jour. En effet, à l'aide d'un routage géographique, ces messages seront transmis en direction du centre pour atteindre le serveur de zone. Les messages doivent contenir, en plus de l'adressage classique, l'identifiant du nœud portant le statut de serveur de zone. De façon similaire au jeton manipulé dans les réseaux en anneaux, cet identifiant dynamique est échangé entre les nœuds d'une zone lors la migration de l'agent mobile. Ainsi, les nœuds d'une zone n'ont pas à connaître sur quelle nœud est hébergé le serveur.

2.2.5. Propriété du système d'agents mobiles

Afin de réduire les surcoûts liés à la gestion des serveurs de zone, nous avons utilisé dans notre approche des agents mobiles avec une très faible granularité, i.e., des agents qui comportent très peu de code et qui effectuent des traitements simples. Ceci permet des migrations rapides et nombreuses, non très couteuses. Il est alors très intéressant dans ce cas, d'utiliser plus d'agents pour permettre la tolérance aux fautes et plus d'efficacité dans le fonctionnement des serveurs. Chaque serveur de zone pourrait être représenté par un deuxième agent mobile, placé sur un autre nœud, pour prendre le relais en cas de panne par exemple. Il pourrait aussi avoir des représentants au niveau des autres zones du réseau, qu'il pourrait solliciter dans certaines fonctions de gestion de réseau.

3. Conclusion

Pour pallier l'absence d'infrastructure fixe dans les réseaux mobiles ad hoc, nous avons proposé dans ce chapitre un serveur virtuel centralisé, basé sur la technologie d'agent mobile, chargé de la gestion de la base de données de localisation des codes mobiles dans les réseaux ad hoc. L'un des objectifs de cette approche centralisée, qui permet une flexibilité et une auto-organisation du serveur de réseau, est de fournir une permanente disponibilité de services dans les réseaux ad hoc. Ceci en maintenant le serveur du réseau dans une région centrale, permettant un accès rapide par les nœuds du réseau. Le serveur de localisation constitué par un système multi-agents mobiles, n'est pas lié en permanence à des nœuds du réseau. En effet, il peut changer de support physique lorsque cela est nécessaire, et ce, pour offrir au mieux ses services aux utilisateurs.

Dans la première partie de ce chapitre, nous avons testé une première version de notre alternative, de type totalement centralisé. Les résultats d'expérimentation de cette version, comparativement à l'approche distribuée étudiée dans le chapitre précédent, ont montré une dégradation des performances du protocole de localisation centralisé, notamment en termes de consommation en bande passante. Cela était prévisible d'après la littérature, à savoir que les approches totalement centralisées sont inadaptées dans un réseau ad hoc caractérisé par une bande passante limitée, car elles ont tendance à saturer la région centrale et former ainsi un goulet d'étranglement au niveau du seul serveur de réseau. L'étude expérimentale à montré

aussi que lorsque les degrés de mobilité des stations ou des agents sont élevés, les surcharges induites au niveau du serveur ne permettent pas d'améliorer les performances du protocole centralisé en termes de temps de réponse, malgré la disponibilité du serveur qui est censé satisfaire dans de meilleurs délais les requêtes de localisation. Une telle saturation ralentie évidemment la réactivité du serveur de localisation pour traiter efficacement les requêtes de localisation.

Dans la deuxième partie de ce chapitre nous avons proposé une distribution du protocole centralisé, qui consiste en une gestion multi-serveurs du problème afin de répartir la charge de gestion de la localisation, avec une structuration du réseau ad hoc en plusieurs zones pour limiter l'étendue des diffusions de messages de contrôle et des requêtes de localisation. Ajouté à cela, l'utilisation d'agents mobiles de granularité faible dans notre approche, permet de maintenir les serveurs du réseau dans une région centrale rapidement accessible par les nœuds du réseau, notamment lorsqu'un routage géographique est utilisé. Ainsi, la technologie d'agent mobile permet d'assurer une certaine stabilité dans le fonctionnement du système, pour une meilleure disponibilité du service malgré la mobilité des nœuds dans le réseau. Toutefois, notre approche qui représente une autre alternative de gestion pour les réseaux mobiles ad hoc, n'est pas à l'abri de certains inconvénients tel que le surcoût liés à la gestion des serveurs et à la maintenance des informations de localisation, particulièrement lorsque le degré de mobilité des nœuds et du code est très élevé et lorsque la taille du réseau devient très importante.

Enfin, notre approche qui offre une disponibilité de services, grâce à l'auto-organisation des agents mobiles, permet aussi de répartir la consommation énergétique à travers plusieurs nœuds du réseau, qui sont sollicités à assurer la fonction de serveur lorsque cela est indispensable. Nous pensons donc, qu'il serait intéressant d'adapter notre approche à d'autres types de problèmes posés dans les réseaux mobiles ad hoc, tel que le routage d'information.

Conclusion générale

La gestion des réseaux mobiles ad hoc est une tâche extrêmement difficile, comparée aux réseaux filaires, et ceci à cause de leurs propres caractéristiques. Ces réseaux doivent donc disposer de leurs propres systèmes de gestion, puisque les systèmes de gestion classiques pour les réseaux filaires ne leurs sont plus adaptés. Ainsi, le défi majeur des systèmes de gestion pour les réseaux ad hoc est de parvenir à proposer une qualité de service à ses utilisateurs, comparables à ceux des réseaux classiques et cela de façon transparente.

Nous nous sommes intéressés dans cette thèse à la problématique de l'utilisation de la technologie d'agents mobiles dans un cadre de gestion des réseaux mobile ad hoc. A travers les différentes caractéristiques des agents mobiles que nous avons présenté dans le troisième chapitre, et les différents travaux sur les systèmes de gestion de réseaux mobiles présentés dans le deuxième chapitre, il s'avère que les agents mobiles représentent un paradigme intéressant pour les environnements mobiles, puisqu'ils permettent un mode de fonctionnement déconnecté et asynchrone, une optimisation de la bande passante et une répartition des tâches de calculs. Cependant, la technologie d'agent mobile ne représente pas la solution idéale pour tous les types d'applications réparties, mais simplement une possibilité nouvelle pour la construction des applications et des systèmes. En effet, selon les besoins de performance, de conception, de développement et de sécurité, les agents mobiles pourront apporter une alternative intéressante au classique client/serveur dans certains types de configuration. La performance des agents mobiles sera très utile par exemple, dans le cas des applications qui comportent des communications intensives, ou dans les réseaux caractérisés par des connexions intermittentes. Grâce à leurs interactions locales, ces agents permettent de ne pas subir ni les ralentissements dû aux bandes passantes limitées et aux temps de latence des réseaux, comme sur Internet, ni les déconnexions fréquentes des réseaux mobiles.

L'utilisation du concept d'agents mobile dans les réseaux mobiles, induit un deuxième niveau de mobilité, i.e., la mobilité logicielle, ajouté au niveau de mobilité physique induit par la prise en charge des stations nomades du réseau mobile. Cependant, cette double mobilité, logicielle et matérielle, n'est pas facile à gérer pour un concepteur qui souhaite utiliser un

service réparti. En effet, il devra être capable de déterminer sa localisation dans un environnement en constante évolution. Le problème de localisation des objets mobiles, étudié dans le quatrième chapitre, a fait l'objet de notre intérêt dans cette thèse afin d'évaluer les performances de notre approche pour la gestion des réseaux mobiles ad hoc.

Nous avons conçu deux types de protocoles dans chacune des deux approches de localisation proposées. Dans l'approche distribuée, nous avons réalisé une étude comparative de performances entre le protocole proactif et le protocole réactif. Les résultats d'expérimentation ont montré que le protocole de localisation réactif est de loin le plus performant par rapport au protocole proactif. La maintenance des informations de localisation de façon proactive est très couteuse en bande passante pour le protocole proactif, ce qui influe négativement sur le temps de réponse des requêtes de localisation. Ceci à l'encontre des prévisions attendues, malgré la disponibilité de telles informations qui sont supposées réduire les délais de localisation. En effet, il s'est avéré que la surcharge du réseau dû aux messages de contrôle, perturbe et ralentit la transmission des requêtes de localisation ainsi que les réponses à ces requêtes dans le réseau.

La deuxième approche proposée de type centralisé, constitue une alternative originale pour les réseaux mobiles ad hoc. L'approche basée sur la technologie d'agent mobile, permet de concevoir des serveurs centralisés et mobiles dans un réseau, là où il n'est pas évident de penser à une gestion centralisée, dû à l'absence de toute administration ou infrastructure fixe dans ces réseaux. L'un des objectifs de l'approche centralisée, qui permet une flexibilité et une auto-organisation du serveur de réseau, est de fournir une permanente disponibilité de services dans les réseaux ad hoc.

Dans le premier protocole de localisation de cette approche, de type purement centralisé, l'étude comparative par rapport à l'approche distribuée, a montré une dégradation des performances du protocole, notamment en termes de consommation en bande passante. L'étude expérimentale a confirmée ce qui a été prévu dans la littérature, à savoir que les approches totalement centralisées sont inadaptées dans un réseau ad hoc caractérisé par une bande passante limité, car elles ont tendance à saturer la région centrale et former ainsi un goulet d'étranglement au niveau du seul serveur de réseau. L'étude a montré aussi que lorsque les degrés de mobilité des stations ou des agents sont élevés, les surcharges induites au niveau du serveur ne permettent pas d'améliorer les performances du protocole centralisé en termes de temps de réponse, malgré la disponibilité du serveur qui est censé satisfaire dans de meilleurs délais les requêtes de localisation. Une telle saturation ralentie évidemment la réactivité du serveur de localisation pour traiter efficacement les requêtes de localisation.

L'analyse des causes de cette dégradation de performances, nous a conduits à proposer une distribution du protocole centralisé, qui consiste en une gestion multi-serveurs du problème afin de répartir la charge de gestion de la localisation, avec une structuration du réseau ad hoc en plusieurs zones pour limiter l'étendue des diffusions des messages de contrôle et des requêtes de localisation. Ajouté à cela, l'utilisation d'agents mobiles de granularité faible dans notre approche, permet de maintenir les serveurs du réseau dans une région centrale rapidement accessible par les nœuds du réseau, notamment lorsqu'un routage

géographique est utilisé. Ainsi, la technologie d'agent mobile permet d'assurer une certaine stabilité dans le fonctionnement du système, en offrant une meilleure disponibilité de services malgré la mobilité des nœuds dans le réseau. Elle permet aussi une répartition de la consommation énergétique par le serveur à travers plusieurs nœuds du réseau.

Notre approche qui représente une autre alternative de gestion pour les réseaux mobiles ad hoc, n'est pas à l'abri de certains inconvénients, tel que le surcoût liés à la gestion du serveur multi-agent et à la maintenance des informations de localisation, particulièrement lorsque le degré de mobilité des nœuds et du code est très élevé. Toutefois, nous pensons qu'il serait intéressant d'adapter l'approche à d'autres types de problèmes posés dans la gestion des réseaux mobiles ad hoc.

L'inconvénient classique de défaillance des serveurs ou leurs déconnexions involontaires, ainsi que le problème de partitionnement dans le réseau feront l'objet de nos travaux futurs. L'adaptation de notre approche semi-centralisée à la gestion de la mobilité des nœuds dans un réseau ad hoc est en cours d'étude. Une comparaison de ses performances avec les autres services de localisation est envisagée. Nous envisageons aussi d'appliquer l'approche à d'autres fonctions de gestion dans les réseaux ad hoc tel que la fonction de routage d'information. Il serait de même très intéressant de la combiner avec d'autres techniques de gestion déjà existantes.

Bibliographie

[1] A. Ahmed and B. Far, "Mobile agent system for network topology discovery", *IEEE CCECE/CCGEI*, Ottawa, May 2006.

[2] S. Alouf, F. Huet, and P. Nain, "Forwarders vs. Centralized server: An evaluation of two approaches for locating mobile agents", *Rap. de rech*. N°4440, 2002 INRIA.

[3] K. M. Aloysius and Y. Weijiang, "TINMAN: A resource bound security checking system for mobile code", *ESORICS'02, European symposium on research in computer security*, Zurich, SUISSE, 2002.

[4] P. Anelli and E. Horlait, "NS-2: Principes de conception et d'utilisation", *http://www-sop.inria.fr/rodeo/personnel/Pierre.Ansel/Manuel_NS1.3.pdf*

[5] N. Badache and T. Lemlouma, "Le routage dans les réseaux mobiles Ad hoc", *Master Degree Dissertation.University of USTHB, Algiers, Algeria,* September, 2000.

[6] H. Badis, "Étude et conception d'algorithmes pour les réseaux mobiles et ad hoc", *thèse de doctorat*, Université de Paris-sud, December 2005.

[7] S. Bandyopadhyay and K. Paul, "Evaluating the Performance of Mobile Agent-Based Message Communication among Mobile Hosts in Large Ad Hoc Wireless Network", *MSWiM'99*, Seattle, USA, ACM 1999.

[8] P. Basu, N. Khan and T.D.C. Little, "A Mobility Based Metric for Clustering in Mobile Ad Hoc Networks", *Proc. IEEE ICDCS'01 Workshop on Wireless Networks and Mobile Computing*, April 2001.

[9] A. Beach, "GLS (Grid Location System): Performance Observations & Summary".

[10] B. Benmammar et F. Krief, "La technologie agent et les réseaux sans fil", *17ème congrès DNAC,* Paris, October, 2003.

[11] S. Ben Sassi and N. Le Sommer, "Une plate-forme intergicielle pour la découverte et l'invocation de services géolocalisés dans les réseaux ad hoc discontinus", In *9e Conférence Internationale sur les NOuvelles TEchnologies de la REpartition (Notere 2009)*, p. 28-37, Canada, Juillet 2009.

[12] C. Bergerot, F. Bellegarde et J.F. Monin, "Conception et vérification de code mobile pour des besoins télécom", *http://lifc.univ-fcomte.fr/~bergerot/These.html*

[13] M. Bernichi, "Surveillance logicielle à base d'une communauté d'agents mobiles", *Thèse de doctorat*, Université de Paris-Est, Novembre 2009.

[14] K. Beydoun, "Conception d'un protocole de routage hiérarchique pour les réseaux de capteurs", *thèse de doctorat*, Université de Franche-Comté, Décembre, 2009.

[15] A. Bieszczad, B. Pagurek and T. White, "Mobile agents for network management", *IEEE Communications Survey*, Vol. 1, n° 1, 1998.

[16] L. Blazevic, J.-Y. Le Boudec and S. Giordano, "A Location-Based Routing Method for Mobile Ad Hoc Networks", *IEEE Transactions on mobile computing*, Vol. 4, N°. 2, March/April 2005.

[17] A. Boukerche and Y. Ren, "A Novel Solution based on Mobile Agent for Anonymity in Wireless and Mobile Ad hoc Networks", *Q2SWinet'07*, October 22, Greece, ACM 2007.

[18] J-P. Briot et Y. Demazeau, " Introduction aux agents, principes et architecture des systèmes multi-agents ", *collection IC2*, Hermès, 2001.

[19] T. Camp, J. Bolengand and L. Wilcox, "Location information services in mobile ad hoc networks". *In Proc. of the IEEE Intern. Conf. on Communications (ICC),* pp 3318-3324, 2001.

[20] T. Camp, "Location Information Services in Mobile Ad Hoc Networks", *Technical Report*, Colorado School of Mines, October, 2003.

[21] N. Castañeda Palomino, "Géo-localisation et poursuite dans un réseau mobile", *thèse de doctorat*, Ecole Nationale Supérieure des Télécommunications, Paris, Juillet 2008.

[22] R.Castañeda, S.R. Das and M.K. Marina, "Query Localization Techniques for On-demand Routing Protocols in Ad Hoc Networks", *In Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing*. 1999.

[23] R. Chadha, H. Cheng, Y.-H. Cheng, J. Chiang, A. Ghetie, G. Levin and H. Tanna, "Policy-Based Mobile Ad Hoc Network Management", *Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks* (POLICY'04), 2004.

[24] W. Chen, N. Jain and S. Singh, "ANMP: Ad Hoc Network Management Protocol", *IEEE Journal on selected areas in communications*, vol. 17, N°. 8, August 1999.

[25] C.-Y. J. Chiang, S. Demers, P. Gopalakrishnan, L. Kant, A. Poylisher, Y.-H. Cheng, R. Chadha, G. Levin, S. Li, Y. Ling, S. Newman, L. LaVergne and R. Lo, "Performance analysis of DRAMA: a distributed policy-based system for Manet management", *IEEE Conference Military Communications (MILCOM)*, 2006.

[26] L.-D. Chou C.-C. Kao, "Network Fault Management Systems Using Multiple Mobile Agents for Multihomed Networks" *IEEE International Conference on Communications*, 2003.

[27] C. Cubat, "Agents Mobiles Coopérants pour les Environnements Dynamiques", *thèse de doctorat*, Université de l'INP de Toulouse, décembre 2005.

[28] M. K. Denko, "The use of Mobile Agents for Clustering in Mobile Ad Hoc Networks", *Proceedings of SAICSIT'03*, Pages 241 – 247, 2003.

[29] S. S. Dhillon, X. Arbona and P. Van Mieghem, "Ant Routing in Mobile Ad Hoc Networks", *International Conference on Networking and Services (ICNS '07)*, 2007.

[30] G. Di Caro and M. Dorigo, "Mobile Agents for Adaptive Routing", *In Proceedings of the 31st Hawaii International Conference on Systems*, pp. 88-103, 1998.

[31] G. Di Caro, F. Ducatelle and L. M. Gambardella, "AntHocNet: an Ant-Based Hybrid Routing Algorithm for Mobile Ad Hoc Networks", *In Proceedings of PPSN VIII - Eight International Conference on Parallel Problem Solving from Nature*, LNCS, no. 3242. UK, Sept. 2004

[32] F. Douglis and J. Ousterhout, "Process migration in the Sprite operating system", *Proceedings of the 7th Inter. Conference on Distributed Computing Systems*. 1987.

[33] T. C. Du, E. Y. Li, and A.-P. Chang, "Mobile Agents in Distributed Network Management", *Communications of the ACM,* Vol. 46, No. 7, July 2003.

[34] A. Duda et S. Perret, "Une architecture d'agents mobiles pour des réseaux de stations nomades", *In Proc. Colloque Francophone sur l'Ingénierie des Protocoles,* Liège, septembre, 1997.

[35] M. El-Darieby and A. Bieszczad, "Intelligent Mobile Agents: Towards Network Fault Management Automation" *IFIP/IEEE International Symposium on Integrated Network Management*, 1999.

[36] A. El Fallah-Seghrouchni, S. Haddad, T. Melitti et A. Suna, "Interopérabilité des systèmes multi-agents à l'aide des servicesWeb ", *JFSMA,* 2004.

[37] I.I. Er et W.K.G. Seah, "Mobility-based d-Hop Clustering Algorithm for Mobile Ad Hoc Networks", *Elsevier Computer Networks*, Vol. 50, pp.3375-3399, 2006.

[38] D. Espès, "Protocoles de routage réactifs pour l'optimisation de bande passante et la garantie de délai dans les réseaux ad hoc mobile", *thèse de Doctorat*, Université Toulouse III - Paul Sabatier, Nov. 2008.

[39] Y-X. Feng, L.-L. Zhao and G.-X. Wang, "A Clustering Algorithm applied to the Network Management on Mobile Ad hoc Network", *International Conferences on Info-tech and Info-net, (ICII),* 2001.

[40] J. Ferber, "Les systèmes multi-agents. Vers une intelligence collective", *InterEditions*, 1995.

[41] M. Gerla, X. Hong, G. Pei, "Fisheye State Routing Protocol (FSR) for Ad Hoc Networks", *INTERNET-DRAFT. http://tools.ietf.org/html/draft-ietf-manet-fsr-03,* [on line] 17 June 2002.

[42] G. Goldszmidt and Y. Yemini, "Delegated Agents for Network Management", *IEEE Communications Magazine*, pp. 66-70, March 1998.

[43] T. Goore Bi, I. Lokpo and G. Padiou, "Localisation décentralisée et adaptative d'agents mobiles dans les réseaux dynamiques ", *SympAAA'05*, France, 2005.

[44] FL Guo, B. Zeng, LZ. Cui "A Distributed Network Management Framework Based on Mobile Agents" *Third International Conference on Multimedia and Ubiquitous Engineering*, IEEE 2009.

[45] M. Gunes, U. Sorges and I. Bouazizi, "ARA – The Ant-Colony Based Routing Algorithm for MANETs", *International Conference on Parallel Processing Workshops,. Proceedings,* 2002.

[46] B. Guy, "Apport des agents mobiles à l'exécution répartie", *4ième Ecole d'Informatique des Systèmes Parallèles et Répartis*, Toulouse, France, February, 2000.

[47] L. Hagen, M. Breugst, T. Magedanz, "Impacts of Mobile Agent Technology on Mobile Communications System Evolution", *IEEE Personal Communications Magazine, vol. 5. N°4,* p. 56-69, août 1998.

[48] D. Hagimont and L. Ismail, "Agents mobiles et client/serveur : évaluation de performance et comparaison ", *Technique et science informatiques*, Volume 19 – n° 9/2000.

[49] A. Hamdi Shabaan, H. EIZouka and M. Abou EINasr, "Intrusion Detection System in wireless Ad-hoc Networks Based on Mobile Agent Technology", *2nd IEEE International Conference on Computer Engineering and Technology*, 2010.

[50] K. Heurtefeux and F. Valois, "Localisation collaborative pour réseaux de capteurs", *Colloque Francophone sur l'Ingénierie des Protocoles (CFIP)*, 2008.

[51] A. Hijazi, "Using Mobile Agents for Intrusion Detection in Wireless Ad Hoc Networks", *International Conference on Wireless and Optical Communications Networks*, 2005.

[52] K. Hosoon, W.Gottfried R. Lunderer and B. Subbiah, "An Intelligent Mobile Agent Framework for Distributed Network Management", *In IEEE proceedings*, 1997.

[53] J. Hu, R. Zhao, X.-M. Qiu, "A Network management model based on mobile agent system", *IEEE International Conference on Apperceiving Computing and Intelligence Analysis*, 2008.

[54] M.A.M. Ibrahim, "Distributed Network Management with Secured Mobile Agent Support", *International Conference on Hybrid Information Technology (ICHIT'06),* 2006.

[55] X. Jiang and T. Camp, "An efficient location server for an ad hoc network", *Technical Report,* Dept. of Math and Computer Science, Colorado School of Mines, May, 2003.

[56] O. Kachirski and R. Guha, "Intrusion Detection Using Mobile Agents in Wireless Ad Hoc Networks", *Proceedings of the IEEE Workshop on Knowledge Media Networking (KMN'02).*

[57] M. S. Kakkasageri, S. S. Manvi and B. M. Goudar, "An Agent based Framework to Find Stable Routes in Mobile Ad hoc Networks (MANETs)", *IEEE Conference TENCON,* 2008.

[58] M. Kasemann, H. Fubler, H. Hartenstein and M. Mauve, "A Reactive Location Service for Mobile Ad Hoc Networks", *Technical Report TR-02- 014, Department of Computer Science, University of Mannheim,* November, 2002.

[59] Y.-B. Ko and N.H. Vaidya, "Location-Aided Routing (LAR) in Mobile Ad Hoc Networks", *Journal of Wireless Networks, Volume 6 Issue 4,* July 2000.

[60] A. Koliousis and J. Sventek, "A Trustworthy Mobile Agent Infrastructure for Network Management", *IEEE International Symposium on Integrated Network Management,* 2007.

[61] V. Kumar and S.R. Das, "Performance of dead reckoning-based location service for mobile ad hoc networks", *Wireless Communication and Mobile Computing Journal,* December, 2003.

[62] J. Li, J. Jannotti, D. De Couto, D. Karger and R. Morris, "A scalable location service for geographic ad hoc routing", *Proc. of the ACM/IEEE Intern. Conf. on Mobile Computing and Networking (MOBICOM)*, pp 120-130*,* 2000.

[63] A.B. Kulkarni, R. Spackmann and G. Kuthethoor, "Self-organized management of mobile adhoc networks", *IEEE Military Communications Conference MILCOM,* 2006.

[64] N. Le Sommer**, "**Vers une ubiquité d'accès aux services dans les réseaux mobiles ad hoc**,"** In *7e Conférence Internationale sur les NOuvelles TEchnologies de la REpartition (Notere 2007)*, p. 207-218, Juin 2007.

[65] I. Lokpo, T. Goore Bi and G. Padiou, "Ad Hoc Location Service for Mobile Agents", *IEEE int'l Conference on Signal-Image Technology & Internet-based Systems (SITIS'05),* November*,* 2005.

[66] X. Luo, T. Camp and W. Navidi, "Predictive methods for location services in mobile ad hoc networks", *in Proceedings of the 23rd IEEE IPCCC*, pp. 337-, 2004.

[67] H. Ma & S.T. Tan, "Dynamic Mobile Agent Based Distributed Network Management Using Internet Technology and CORBA", *IEEE International Conference on Systems, Man, and Cybernetics,* 1999.

[68] S. Marwaha, C.K. Tham and D. Srinivasan, "Mobile Agents based Routing Protocol for Mobile Ad Hoc Networks", *Symposium on Ad Hoc Wireless Network*, IEEE GLOBCOM 2002.

[69] H. Matsuo and K. Mori, "Accelerated Ants Routing in Dynamic Networks", in *Proc. Int. Conf. on Software Engineering, Intelligence, Networking and Parallel/Distributed Computing.* pp. 333-339, Aug. 2001.

[70] M. Mauve, J. Widmer and H. Hartenstein, "A survey on position-based routing in mobile ad-hoc networks", *IEEE Network*, 15(6):30.39, November/December 2001.

[71] M. McGuire, K.N. Plataniotis and A.N. Venetsanopoulos, "Estimating position of mobile terminals from path loss measurements with survey data", *Wireless communications and mobile computing.* 3:51-62, August, 2003.

[72] N. Migas, W.J. Buchanan, and K. McArtney, "Mobile Agents for Routing, Topology Discovery, and Automatic Network Reconfiguration in Ad-Hoc Networks", *10th IEEE International Conference and Workshop on the Engineering of Computer Based Systems*, Huntsville, USA, pp. 200-206, 2003.

[73] N. Migas, W.J. Buchanan and K. McArtney, "Migration of Mobile Agents in Ad-hoc, Wireless Networks", *Proceedings of the 11th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems (ECBS'04).* 2004.

[74] N. Minar, K.H. Kramer, and P. Maes. "Cooperating mobile agents for dynamic network routing". In A. Heyzelden and J. Bigham, editors, *Software Agents for Future Communication Systems,* pages 287-304. Springer, 1999.

[75] P. Mockapetris and K. J. Dunlap, "Development of the Domain Name System", *in: Proc. of SIGCOMM'88*, Stanford, California, pp. 123-133, 1988.

[76] B. Nancharaiah, G.F. Sudha and M.B.R. Murthy, "A Scheme for Efficient Topology Management of Wireless Ad hoc Networks using the MARI Algorithm", *IEEE ICON,* 2008.

[77] M.T. Nguyen and V. Moraru, "Les protocoles pour la gestion des réseaux Informatiques", *Rapport de recherche*, Hanoi, Juillet 2005.

[78] X. Ning, S. Zhang, W. Liang and Y. Sun, "Peer Management Domain Agents Architecture for MANET Management", *IEEE 5th International Symposium on Autonomous Decentralized Systems,* 2001.

[79] R. Onishi, S. Yamaguchi, H. Morino, H. Aida and T. Saito, "The Multi-agent System for Dynamic Network Routing", *5th International Symposium on Autonomous Decentralized Systems*, 2001.

[80] A. Pashalidis and M. Fleury, "Secure network management within an open-source mobile agent framework", *Journal of Network and Systems Management*, vol. 12, no. 1, pp. 9-31, March 2004.

[81] S. Perret, "Agents mobiles pour l'accès nomade à l'information répartie dans les réseaux de grande envergure", *Thèse de doctorat*, Université de Grenoble-I, Novembre 1997.

[82] S. Pierre, "Réseaux et systèmes informatiques mobiles. Fondements, architectures et applications", *Presses internationales Polytechnique*, 2003.

[83] Y. Ping, Y. Yan, H. Yafei, Z. Yiping and Z. Shiyong, "Securing Ad Hoc Networks through mobile agent", *InJbSecu'04,* November 14-16, 2004, Shanghai, China.

[84] M.L. Powell and B. P. Miller, "Process migration in DEMOS/MP", *in Proc. of SOSP'83*, Bretton Woods, New Hampshire, published as Operating Systems Review, 17 (5), 110-119, 1983.

[85] G. Pujolle, "Les réseaux", *5ième édition, Eyrolles,* 2006.

[86] A. Puliafito, O. Tomarchio, and L. Vita. "MAP : Design and Implementation of a Mobile Agent Platform". *Journal of System Architecture*, 46(2):145–162, 2000.

[87] M. Rajabzadeh, F. Adibniya and M. ghasemzadeh, "MA-DSR; Multi Agent based Adaptive DSR Protocol with Intelligent Behavior in Realistic Environments", *IEEE Internatioal Symposium on Telecommunications,* 2008.

[88] S. Rajagopalan and C.-C. Shen, "ANSI: A Unicast Routing Protocol for Mobile Ad hoc Networks Using Swarm Intelligence", *Journal of Systems Architecture, Volume 52 Issue 8*, August 2006.

[89] N. Ramamurthy, "Role of mobile agents in mobile computing environment", *http://crystal.uta.edu/~kumar/cse6392/termpapers/Naveen_paper.pdf*

[90] S.Ramanathan and M.Steenstrup, "A Survey of Routing Techniques for Mobile Communications Networks", *ACM Mobile Networks and Applications,* vol. 1, pp. 89-104, 1996.

[91] E. Reuter, "Agents Mobiles : Itinéraires pour l'administration système et réseau", *Thèse de doctorat*, Université de Nice Sophia Antipolis, Mai 2004.

[92] R. RoyChoudhury, S. Bandyopadhyay and K. Paul, "A Distributed Mechanism for Topology Discovery in Ad Hoc Wireless Networks Using Mobile Agents", *MobiHOC Mobile Ad Hoc Networking and Computing*, pp 145-146, 2000.

[93] M. G. Rubinstein and O. C. B Duarte, "Evaluating the Performance of Mobile Agents in Network management", *In: IEEE Global Telecommunications Conference*, pp.386 – 390, December 1999.

[94] M.G. Rubinstein, O.C.B. Duarte and G. Pujolle, "Improving Management Performance by Using Multiple Mobile Agents", *ACM Agents 2000*, Barcelona Spain, pp. 165-166, 2000.

[95] M. G. Rubinstein, O.C.B. Duartel and Guy Pujolle, "Using Mobile Agent Strategies for Reducing the Response Time in Network Management", *IEEE International Conference on Communication Technology Proceedings,* 2000.

[96] A. Sahai "Conception et réalisation d'un gestionnaire mobile de réseaux fondé sur la technologie d'agent mobile ". *Thèse de doctorat*, Université de Rennes, January, 1999.

[97] F. Sailhan, "Localisation de ressources dans les réseaux ad hoc", *Thèse de doctorat*, Université de Paris VI, Avril, 2010.

[98] I. Satoh, "MobileSpaces: A Framework for Building Adaptive Distributed Applications using a Hierarchical Mobile Agent System", *Proceedings of International Conference on Distributed Computing Systems (ICDCS'2000)*, pp.161-168, IEEE Computer Society, April, 2000.

[99] I. Satoh, "Reusable Mobile Agents for Cluster Computing", *Proceedings of the IEEE International Conference on Cluster Computing (CLUSTER'03),* 2003.

[100] I. Satoh, "Building Reusable Mobile Agents for Network Management", *IEEE Transactions on systems, man, and cybernetics, Part C: Applications and reviews,* Vol. 33, N°. 3, August 2003.

[101] C. Servin, "Réseaux et télécoms" *2ième édition, Dunod* 2006.

[102] A.H. Shabaan, H. EIZouka and M. Abou EINasr, "Intrusion Detection System in wireless Ad-hoc Networks Based on Mobile Agent Technology", *2nd IEEE International Conference on Computer Engineering and Technology*, 2010.

[103] A. Shajin Nargunam and M.P Sebastian, "Cluster Based Security Scheme for Mobile Ad Hoc Networks", *IEEE International Conference on Wireless and Mobile Computing, Networking and Communications,* 2006.

[104] J.F. Sharmila, B.R. Elijah and S.B. Shyam, "Mobile Agent Based AHP Clustering Protocol in Mobile Ad Hoc Network", *Advances in Computational Sciences and Technology*, ISSN 0973-6107 Volume 3 Number 1, pp. 77–96, 2010.

[105] G.S. Sharvani, N.K. Cauvery and T. Rangaswamy, "Adaptative routing algorithm for manet: Termite", *International Journal of Next-Generation Networks (IJNGN)*,Vol.1, No.1, December 2009.

[106] H.M.P. Shekhar and K.S. Ramanatha, "Mobile Agents based Framework for Routing and Congestion Control in Mobile Ad Hoc Networks", *CoNEXT'06*, December 4–7, 2006, Lisboa, Portugal, ACM 2006.

[107] C.-C. Shen, C. Snsathaporphat and C. Jaikaeo, "An Adaptive Management Architecture for Ad hoc Networks", *IEEE Communications Magazine, Management of next Generation Wireless Network and Services*, February 2003.

[108] T. Shu, C. Yang, Y. Jianhua and X. Ning "A Mobile Agent Based Approach for Network Management", *IEEE International Conference on Communication Technology,* 2000.

[109] L.M. Silva, P. Simoes, G. Soares, P. Martins, V. Batista, C. Renato, L. Almeida, and N. Stohr, "JAMES: A Platform of Mobile Agents for the Management of Telecommunication Networks", *http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1*

[110] I. Sow, "Partitionnement et Geocasting dans les Réseaux Mobiles Ad Hoc et Collecte des Données dans les Réseaux de Capteurs Sans Fils", *Thèse de doctorat*, Université de Picardie Jules Verne, Dec. 2009.

[111] M. Storey and B. Gordon, "Resource Configuration in Ad Hoc Networks: The MARE Approach", *Third IEEE Workshop on Mobile Computing Systems and Applications,*2000.

[112] D. Subramanian, P. Druschel and J. Chen, "Ants and Reinforcement Learning: A Case Study in Routing in Dynamic Networks", *International Joint Conference on Artificial Intelligence*, 1998.

[113] R. Sugar and S. Imre, "Adaptive Clustering Using Mobile Agents in Wireless Ad Hoc Network", *Lecture Notes in Computer Science*, Vol. 2158, pp: 199-204, 2001.

[114] F. Theoleyre, "Une auto-organisation et ses applications pour les réseaux ad hoc et hybrides", *Thèse de doctorat,* Institut INSA de Lyon, Sept, 2006.

[115] H.H. To, S. Krishnaswamy and B. Srinivasan, "Mobile Agents for Network Management: When and When Not!", *ACM Symposium on Applied Computing,* 2005.

[116] T. White, A. Bieszczad and B. Pagurek, "Distributed Fault Location in Networks Using Mobile Agents", *In Proceedings of the Second International Workshop on Agents in Telecommunications Applications*, July 4th-7th, pp. 130-141, 1998.

[117] Z. Ying and X. Debao, "Mobile Agent-based Policy Management for Wireless Sensor Networks", *IEEE International Conference on Wireless Communications, Networking and Mobile Computing*, 2005.

[118] Y. Zafoune and A. Mokhtari, "Localization of Mobile Agents in Mobile Ad hoc Networks", *ECUMN 2004, LNCS* 3262, pp.337-348, Porto, Portugal, October, 2004.

[119] Y. Zafoune and A. Mokhtari, "Reactive vs. proactive protocol: a comparative study between two localization approaches of mobile codes in ad hoc mobile networks", *ISCN'06, IEEE,* pp 36-41, Istambul, Turquey, June, 2006.

[120] Y. Zafoune, R. Kanawati and A. Mokhtari, "Mobile agents localization in ad hoc networks: a comparative study of centralized and distributed approaches", *ICICT'07, IEEE,* pp 269-275, Cairo, Egypt, December, 2007.

[121] Y. Zafoune, A. Mokhtari and S. Silmi, "Towards a Distributed Form of Centralized Approach for Mobile Agents Localization in Ad hoc Networks", *CONET 2008, IEEE,* London, UK, Sept, 2008

[122] Y. Zafoune, A. Mokhtari and R. Kanawati, "Mobile-agent approach for mobile code localization in Ad hoc networks", *COMPSAC'09, IEEE,* Seattle, USA, July, 2009.

[123] Y. Zafoune, R. Kanawati and A. Mokhtari, "Mobile codes localization in ad hoc networks: a comparative study of centralized and distributed approaches", *IJCNC,* Vol.2 No.2 March, 2010.

[124] D. Zhang, "Network Management Using Mobile Agent", *Intemational Conference on Communication Technology*, ICCT'98 October 22-24, 1998 Beijing, China.

[125] G.-P. Zheng and W.-B. Dong "The Research of Mobile Agent-based Distributed Network Management" *International Colloquium on Computing, Communication, Control, and Management*, 2009.

[126] S.-Q. Zhong, Z.-Z. Shi and Q.-J. Tian, "AST - a Method of Agent Naming and Localization", *IEEE International Conference on Computer Networks and Mobile Computing,* 2001.

[127] C. Zhongmin, J. Lincheng, X. Baowen and W. Yeqing, "The design and implementation of Mobile Agent-based network management system", *IEEE International Conference on Wireless Communications, Networking and Mobile Computing*, 2007.

[128] L. Zhu, Y. Zhang and L. Feng, Distributed Key Management in Ad Hoc NETwork based on Mobile Agent, *2nd IEEE International Symposium on Intelligent Information Technology Application*, 2008.

[129] A. Ziviani, S. Fdida, J.F. de Rezende and O.C.M.B. Duarte, "Une approche TCP pour la gestion de la localisation dans les réseaux ad hoc mobiles", *GRES*, Brésil, 2003.