N° d'ordre : 02/2004-M/IN

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

Ministère de l'enseignement supérieur et de la recherche scientifique

Université des sciences et de la technologie HOUARI BOUMEDIENNE

Faculté d'Electronique et d'Informatique

MEMOIRE

Présenté pour l'obtention du diplôme de Magister

En : INFORMATIQUE

Option : Programmation et Systèmes

par

ZEBBANE Bahia

Sujet

La livraison multicast de messages en environnement mobile cellulaire.

Soutenu le 03-07-2004 devant le jury composé :

Mme A. AISSANI, Professeur (USTHB) : Présidente.

Mme Z. ALIMAZIGHI, Maître de conférence (USTHB) : Examinatrice.

Mme N. BENSAOU, Maître de conférence (USTHB) : Examinatrice.

Mme B. KADRI, Chargé de cours (USTHB) : Invitée.

Mr N. BADACHE, Professeur (USTHB) : Rapporteur.

Remerciements

Je tiens à remercier le tout puissant de m'avoir donné le courage et la patience jusqu'à l'achèvement de ce modeste travail.

J'exprime ma profonde reconnaissance et mes vifs remerciements à mon promoteur Mr. N. BADACHE pour m'avoir encadrer. Je le remercie également pour sa disponibilité, son suivi et ses critiques constructives.

Je remercie Mme A. Aissani, Mme Z. Alimazighi, Mme N. Bensaou et Mme B. Kadri d'avoir accepter de juger ce travail.

Mes remerciements s'adressent aussi à ma famille, en particulier ma mère, mes amis et tous ceux qui m'ont aidé et contribué de prés ou de loin par une quelconque façon, que ce soit par leur amitié, leurs conseils ou leur soutient moral ou matériel.

A tous ceux qui m'aime, je dédié ce modeste travail.

Table des matières

Table des figures

Introduction générale

L'évolution rapide de la technologie dans les domaines des ordinateurs portables et les communications sans fil a fourni la base pour un nouveau environnement de calcul, appelé ***environnement mobile*** ou ***nomade***.

En utilisant conjointement ces ordinateurs portables et les médiums de communication sans fil, (ondes radiofréquence, ondes lumineuses), il devient possible de rester connecter à son réseau habituel et de communiquer avec les autres unités mobiles tout en se déplaçant. Cependant, les utilisateurs munis de portables peuvent établir une communication sans fil et de continuer à bénéficier des services réseau quels que soit leurs situations géographiques et leurs déplacements.

Ces développements et cette grande souplesse d'utilisation ont cependant un prix, les challenges techniques qui permettront de mettre en œuvre l'environnement de calcul mobile n'ont rien de trivial. Les nouvelles contraintes introduites par l'environnement mobile font qu'il est nécessaire de réviser les algorithmes des systèmes répartis classiques (exclusion mutuelle, l'élection ,etc.).

En effet, lorsque le calcul distribué a été étudié avec précaution, la mobilité pose de nouveaux défis qui n'ont pas adressé au paravent. Donc, le calcul mobile est devenu un axe de recherche très important et qui reflète une tendance technologique et sociale dominante vers les accès omniprésents aux ressources de communication et de calcul.

Un des problèmes fondamentaux qui a été posé dans le calcul distribué et qui se pose aussi dans le calcul mobile est la livraison fiable des messages d'une source : soit à un nœud mobile unique ***unicast***, soit à un groupe de nœuds mobiles ***multicast***.

Plusieurs travaux ont adressé ce problème qui s'avère difficile dans un environnement mobile. Cependant, A. Murphy et al. ont proposé des algorithmes de livraison unicast de messages aux unités mobiles en se basant sur l'algorithme de snapshot de Chandy-Lamport. En plus, ils suggèrent une autre manière de penser à la

mobilité et ce *en traitant les unités mobiles comme étant des messages persistants en mouvement dans le réseau.*

Un autre concerne dans le domaine du calcul mobile est le multicast. Bien que le problème de livraison unicast d'un message à un unique récepteur soit important, ces dernières années, le problème de livraison multicast d'un message à plusieurs récepteurs est devenu aussi important et crucial.

Dans cette thèse, nous nous sommes intéressés au problème de livraison multicast de messages aux unités mobiles dans un réseau cellulaire. Pour adresser ce problème, nous allons utiliser la nouvelle stratégie proposée par A. Murphy et al. et qui traite les unités mobiles comme des messages vue les avantages qu'elle offre. En plus, nous allons nous servir de la notion de snapshot déjà utilisée dans l'algorithme de livraison unicast de messages proposé par A. Murphy et al.

Cette thèse est organisée comme suit :

Le chapitre 1 introduit l'environnement du calcul mobile et les principaux concepts liés á ce nouveau environnement. Le fait que les systèmes mobiles sont émergés des systèmes distribués classiques, nous allons présenter tout d'abord le modèle de système distribué standard. Par la suite, le chapitre décrit le modèle de système distribué avec sites mobiles, les modes de fonctionnement de ces sites, quelques caractéristiques de l'environnement mobile et ainsi quelques problèmes liés à la mobilité. Enfin, on va montrer l'impact de la mobilité et des caractéristiques physiques des unités support de calcul mobile sur les algorithmes distribués.

Le chapitre 2 présente une nouvelle manière d'introduire la mobilité et ce en traitant les unités mobiles comme étant des messages nomades qui préservent leurs identités lorsqu'ils traversent le réseau. Nous allons voir tout d'abord, comment cette notion peut être appliquer aux deux types de la mobilité : physique et logique. Par la suite, le chapitre pose un problème majore qui s'impose avec ce genre de modèle qui est la livraison de messages entre les couples des unités mobiles (physique ou logique). Pour cela, et afin de mieux voir le problème fondamental dont les schémas typiques de livraison de messages souffrent, nous nous présenterons deux approches de livraison de messages.

Le chapitre 3 décrit l'application de la nouvelle stratégie de conception, qui considère les unités mobiles comme des messages, pour résoudre le problème de

livraison de messages. Tout d'abord, nous parlerons du problème de livraison de messages aux unités mobiles en se basant sur le modèle cellulaire. Par la suite, nous présenterons, en bref, la livraison snapshot du moment où l'algorithme proposé par Amy Murphy pour résoudre le problème de livraison de messages, qu'on va présenter, est une adaptation de l'algorithme classique de Chandy-Lamport. Enfin, nous allons donner quelques limitations de ces algorithmes.

Le chapitre 4 présente des protocoles qui traitent le problème du *multicast* en environnement mobile. Ces protocoles représentent, en effet, un schéma de base pour la conception de plusieurs protocoles qui sont venus par la suite. Tout d'abord, nous allons définir le problème du multicast en présentant les effets de la mobilité des unités mobiles sur la livraison fiable des messages multicast. En suite, nous allons présenter les algorithmes proposés par A. Acharya et B. Badrinath dans **[30]** suivis d'une discussion. Finalement, nous présentons un aperçu sur la notion du groupe multicast.

Le chapitre5 présente la contribution d'un nouveau algorithme pour la livraison multicast de messages à des unités mobiles dans un réseau mobile cellulaire. Le protocole est basé sur une nouvelle stratégie qui traite les unités mobiles comme étant des messages circulant dans le réseau. Cet algorithme exploite les avantages offerts par l'algorithme du snapshot de Chandy-Lamport ainsi que l'algorithme d'*unicasting* proposé par Amy Murphy. Donc, l'objectif de notre contribution est d'étudier l'applicabilité du mécanisme du snapshot et la nouvelle stratégie de Amy Murphy pour l'implémentation d'un nouveau protocole qui résout le problème de la livraison multicast de messages en environnement mobile. Tout d'abord, nous présentons en premier lieu les hypothèses sur lesquelles se base notre protocole. Par la suite, nous citons les différentes structures de données utilisées par l'algorithme ; et avant de donner les détails de notre algorithme, nous expliquons tout d'abord son fonctionnement. Finalement, nous terminerons par la présentation de la preuve de correction de notre solution proposée ainsi que certaines critiques.

Chapitre 1

Environnements mobiles

1.1 Introduction

L'évolution rapide de la technologie dans les domaines des ordinateurs portables et les communications sans fil a produit un nouveau environnement de calcul, appelé ***environnement mobile*** ou ***nomade***. Cependant, cet environnement permet à un utilisateur muni de portable d'établir une communication sans fil et de continuer à bénéficier des services de réseau, quel que soit sa situation géographique et ses déplacements.

L'informatique mobile accroît donc considérablement l'intérêt de transporter un ordinateur, quel qu'il soit (portable, *notepad*, etc.).Combiner les services du réseau et la mobilité engendrent de nouveaux services et applications. Ces développements et cette grande souplesse d'utilisation ont cependant un prix, les challenges techniques qui permettront de mettre en œuvre l'environnement de calcul mobile n'ont rien de trivial **[10]**. Les nouvelles contraintes introduites par l'environnement mobile font qu'il est nécessaire de réviser les algorithmes des systèmes répartis classiques (exclusion mutuelle, l'élection ,etc.) **[5]**.

Toutes ces contraintes découlent d'une part de la mobilité, qui induit que le travail ne s'exécute plus seulement sur un réseau statique mais aussi avec des machines qui se déplacent, ce qui complique bien sûr considérablement les exécutions réparties et ainsi introduit de nouveaux problèmes non encore considérés dans la construction des systèmes distribués. Par exemple, les structures logiques (anneau, arbre,…), exploitées

traditionnellement par les algorithmes répartis **[1]** ne peuvent plus être utilisées efficacement dans un environnement où les sites changent de place fréquemment **[2, 3, 4]**. D'autre part, l'architecture des systèmes supportant des ordinateurs mobiles est différente de celle des réseaux fixes. Il peut en effet être nécessaire, par exemple, de supporter la notion de cellule de communication sans fil (zone au sein de laquelle la communication sans fil est possible) et de station support (station du réseau fixe qui a la charge d'une cellule de communication sans fil), etc. **[5]**

En outre, les communications sans fil créent des problèmes de déconnexion, de faible largeur de la bande passante ou de besoins de communication extrêmement variables. Et enfin, la nécessité de portabilité du matériel induit de nombreuses limitations et oblige á gérer la consommation d'énergie, les problèmes de stockage, etc. L'impact du nouvel environnement de calcul est donc important, tant au niveau du réseau que des applications et des systèmes.

Ce chapitre présente l'environnement du calcul mobile et les principaux concepts liés á ce nouveau environnement. Le fait que les systèmes mobiles sont émergés des systèmes distribués classiques, nous allons présenter tout d'abord le modèle de système distribué standard. Par la suite, le chapitre décrit le modèle de système distribué avec sites mobiles, les modes de fonctionnement de ces sites, quelques caractéristiques de l'environnement mobile et ainsi quelques problèmes liés à la mobilité. Enfin, on va montrer l'impact de la mobilité et des caractéristiques physiques des unités support de calcul mobile sur les algorithmes distribués.

1.2 Modèle de système distribué statique

Un système distribué est composé d'un ensemble fini de sites autonomes géographiquement dispersés qui sont interconnectés á travers un réseau de communication filaire. Chaque site possède une mémoire locale et un emplacement de stockage stable et exécute un ou plusieurs processus. Sans perte de généralité, on assume qu'il n'y a qu'un seul processus par site. Les états locaux de tous les processus sont assumés d'être disjoints, ce qui signifie que les processus ne partagent pas de mémoire commune, et ils communiquent seulement par échange de messages á travers

les canaux du réseau de communication. C'est la multiplicité des sites connectés á travers un réseau qui rend le *système distribué*.

Il n'y a aucune supposition sur la vitesse relative des processus et sur les délais de transfert des messages qui sont finis mais imprévisibles. Cette absence des limites temporelles rend le système distribué *asynchrone*.

Dans un système distribué, le comportement de chaque processus se compose des changements de l'état local, et des émissions de messages aux autres processus. Ces actions sont complètement déterminées par un algorithme local, qui détermine aussi la réaction envers les messages reçus. L'exécution concurrente et coordonnée de tous les algorithmes distribués forme un *calcul distribué*. De plus, une propriété essentielle des systèmes distribués est l'absence d'une horloge globale ou d'horloges locales parfaitement synchronisées.

Les occurrences d'actions réalisées par les algorithmes locaux sont dites *événements.* D'un point de vue abstrait, un calcul distribué peut être décrit par les types et l'ordre relatif des événements produits dans chaque processus. Dans un système distribué, deux activités générales peuvent prendre place : les *activités locales* qui sont réalisées de manière indépendante par chaque processus et les *activités de synchronisation* durant lesquelles deux ou plusieurs processus agissent entre eux et échangent des messages.

Les deux activités générales peuvent être décomposées en trois types d'événements : les *événements d'émission*, les *événements de réception*, et les *événements internes*. Un événement d'émission reflète le fait qu'un message ait été émis ; un événement de réception dénote la réception d'un message en même temps que l'état local change suivant le contenu du message. Les événements internes affectent seulement l'état local du processus.

1.3 Modèle de système avec sites mobiles

Le terme "*mobile*" implique être capable á se déplacer tout en préservant la connexion réseau **[6]**. Le modèle de système avec des sites mobiles et qui a tendance à se généraliser, est composé de deux ensembles d'entités distincts : les sites fixes d'un réseau de communication filaire classique et des sites mobiles **[7]**. Certaines sites fixes,

appelée *station de support mobile (MSS – Mobile Support Station)* ont comme rôle de relier les *sites mobiles (MH – Mobile Host)* au réseau. Les MSS_s sont munies d'interface de communication sans fil et la zone géographique couverte par une *MSS* est appelée *cellule*. A chaque *MSS* correspond une cellule à partir de laquelle les *MHs* peuvent émettre et recevoir des messages. Tout site mobile est initialement enregistré dans une et une seule *MSS* appelée *station d'enregistrement (ou Home base)* **[8]**. En plus, une unité mobile peut communiquer avec d'autres unités à travers la *MSS* à laquelle elle est directement rattachée. Un *MH* a la possibilité de quitter sa cellule pour rejoindre une autre cellule. Dans ce cas, la *MSS* de l'ancienne cellule cède les responsabilités de communication du *MH* á la *MSS* de la nouvelle cellule.

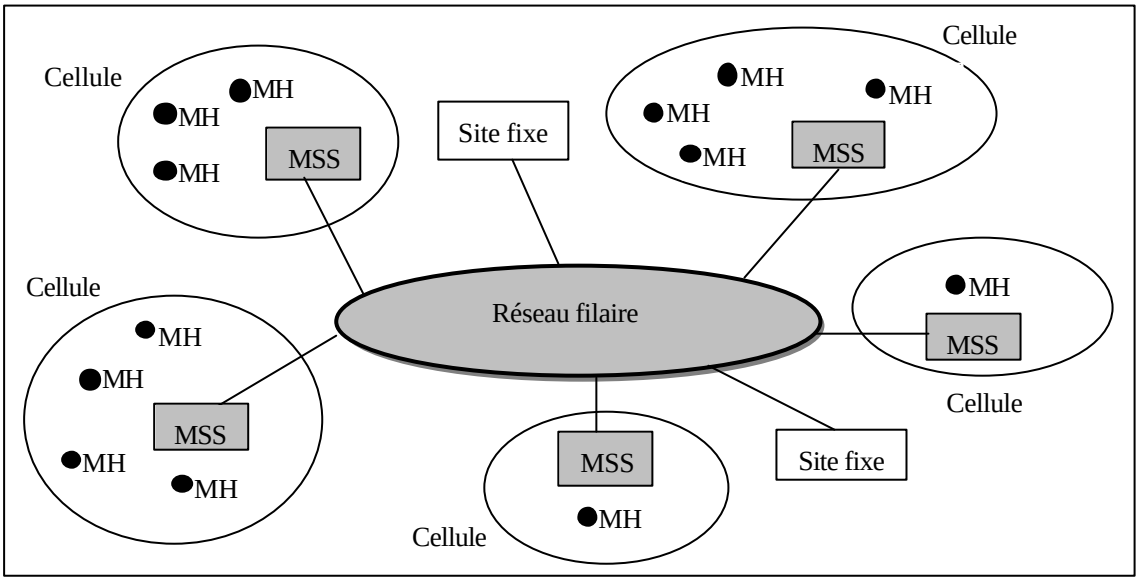


Figure 1.1 – *Architecture d'un système mobile*

1.4 Modes de fonctionnement des mobiles

Dans un système réparti sans ordinateur mobile, une machine ne peut travailler que dans deux modes différents, soit connectée au réseau, soit totalement déconnectée. Par contre en environnement mobile, il existe des degrés variés de déconnexion **[9]**. Le degré de déconnexion est relatif à la largeur de bande passante disponible allouée à la liaison sans fil. Un site mobile dispose donc de davantage de modes de travail. Actuellement, nous dénombrons quatre modes de fonctionnement différents employés sur les sites mobiles, nous les définissons dans ce qui suit **[5]**. La Figure 1.2 résume les

différents modes d'opération supportés par un mobile ainsi que les différentes transitions possibles entre ces états.

a) Mode connecté

Dans ce cas de figure, le mobile dispose d'une connexion normale au réseau, à la manière d'une station classique. La connexion est réalisée par une interface de communication sans fil, qui fournit des débits plus faibles qu'une liaison câblée.

b) Mode partiellement connecté

Le mobile ne dispose plus pour communiquer avec le réseau que d'un lien à faible largeur de bande (connexion faible ou déconnexion partielle). Cette perte de capacité de la bande passante peut être due à des perturbations, à des surcharges de la station de base qui gère les communications des mobiles se trouvant dans sa cellule.

c) Mode veille

Ce mode est utilisé par les mobiles pour préserver leurs ressources énergétiques. La vitesse de l'horloge est alors réduite et les exécutions des applications de l'utilisateur sont suspendues. La liaison avec le réseau est, malgré tout, maintenue, le mobile n'envoie plus de messages, mais peut encore en recevoir et repasser ainsi en mode connecté.

d) Mode déconnecté

Un site mobile peut bien sûr se trouver totalement déconnecté du réseau, à la fois parce qu'il n'y est plus physiquement relié ou parce qu'il est impossible de maintenir une connexion sans fil (volontairement, du fait de fortes interférences ou de surcharges momentanées, par exemple). Les déconnexions totales ou partielles étant très fréquentes pour un mobile connecté via une liaison sans fil, celles-ci ne doivent pas être traitées comme des pannes. Une machine mobile doit être capable de continuer ses exécutions même avec une connexion au réseau ou plus de connexion du tout **[14]**.

Il est á noter que les déconnexions dans un environnement mobile, contrairement á un environnement réparti classique, peuvent le plus souvent être détectées et un

protocole spécifique peut donc être mis en œuvre pour les prendre en compte **[9]**. Le protocole de déconnexion doit être exécuté avant que la machine mobile ne soit physiquement détachée du réseau fixe. Ce protocole est tenu d'assurer que suffisamment d'informations sont présentes localement pour garantir au mobile son autonomie durant la déconnexion.

Le protocole de déconnexion partielle a pour charge de préparer le mobile á exécuter des opérations dans un mode où toutes les communications avec le réseau fixe seront aussi réduites que possible. Finalement, les protocoles de *hand-off* permet á un mobile de sortir des bornes d'une cellule pour entrer dans une autre, tout en conservant la connexion et en mettant á jour des informations sur le réseau fixe (la localisation du mobile, par exemple) et sur le mobile (le nom de la station support locale). Des informations d'état se référant au mobile peuvent, á ce moment, être transférées vers la station support de la nouvelle cellule.

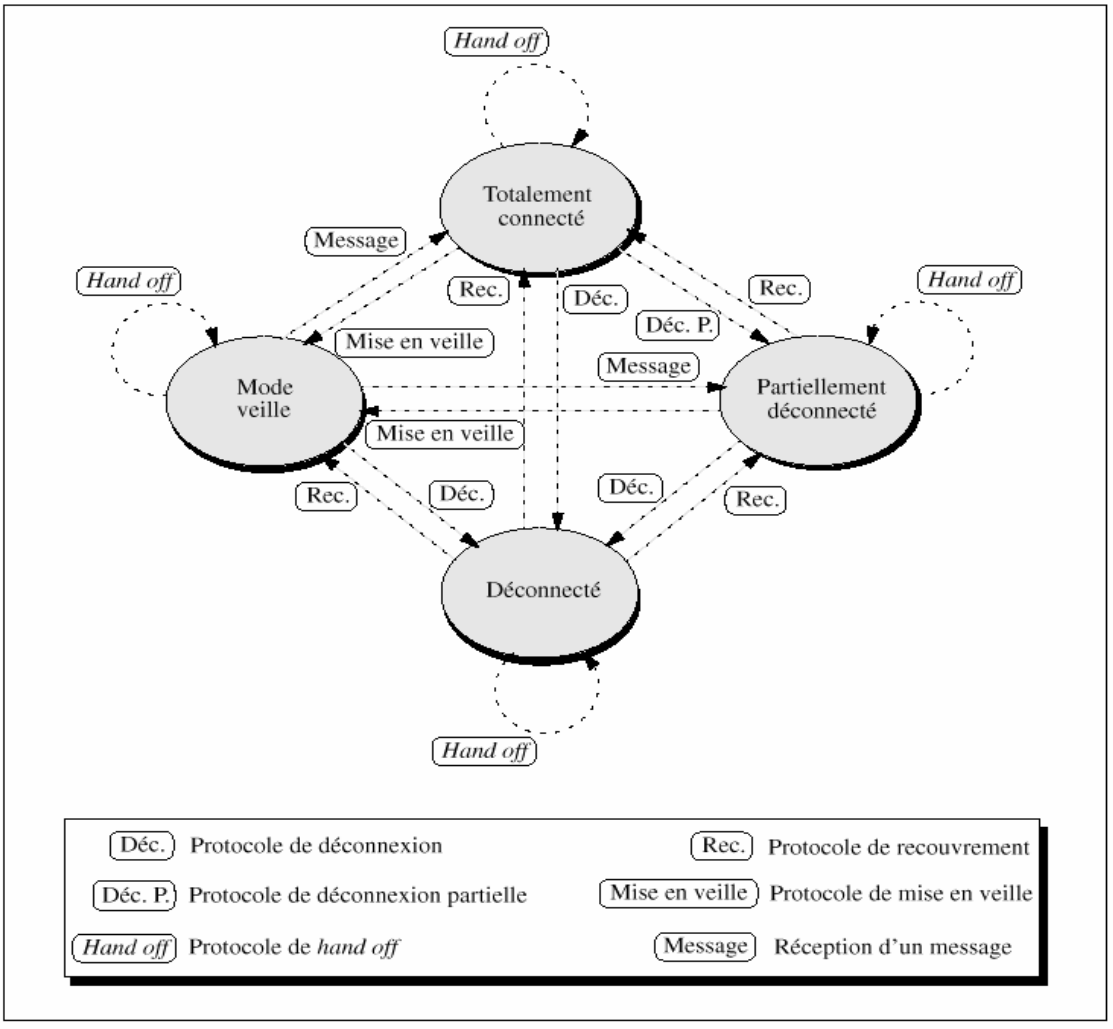


Figure 1.2 – *Modes de fonctionnement d'un site mobile*

1.5 Les caractéristiques des environnements mobiles

L'informatique mobile offre aux utilisateurs la capacité de pouvoir se déplacer tout en restant connecté au réseau et d'être indépendants de toute localisation **[5]**. Pour permettre aux utilisateurs un accès continu aux services et aux ressources du réseau, il est nécessaire de disposer d'interfaces de communication sans fil, á la fois sur certaines stations fixes du réseau et sur les mobiles. C'est alors seulement que la mobilité acquiert tout son intérêt, les utilisateurs étant enfin à même de se déplacer tout en conservant active leur connexion au réseau.

1.5.1 Les spécificités des connexions sans fil

Les mécanismes permettant de mettre en œuvre les connexions sans fil sont multiples, ils impliquent la prise en compte de l'environnement qui interagit avec le signal, le bloque, ou introduit du bruit, de l'écho **[5]**. Les communications sans fil sont donc de moins bonne qualité que les communications filaires : les largeurs de bande sont plus réduites et les besoins très variables, les taux d'erreur plus élevés, les déconnexions fréquentes. Ces facteurs peuvent augmenter les temps de latence dus aux retransmissions, aux délais d'attente entre ces retransmissions, aux exécutions de protocole de contrôle d'erreur et aux courtes déconnexions. Dans un environnement sans fil, les connexions peuvent être perdues ou dégradées du fait de la mobilité et du passage de cellule en cellule, ou du fait d'interférences. A la différence des réseaux fixes, le nombre de machines connectées dans une cellule peut être très variable, et des concentrations importantes d'utilisateurs dans une même cellule peuvent surcharger le réseau **[10]**.

Les points qui suivent passent en revue les différents problèmes induits par l'utilisation de communications sans fil : déconnexions fréquentes, faible bande passante, variabilité de la demande de bande passante. Nous verrons aussi les problèmes d'hétérogénéité des réseaux et d'augmentation des risques du point de vue de la sécurité, qui sont également liés aux médiums de communication sans fil.

a) Les déconnexions

Les systèmes actuels dépendent essentiellement du réseau, ils tendent donc à ne plus fonctionner pleinement durant les pannes qu'ils subissent. Les pannes réseau

concernent encore plus l'informatique mobile que l'informatique classique, puisqu'elle est sujette à de fréquentes déconnexions lors des transmissions sans fil **[10]**. Deux possibilités se présentent alors : soit consommer plus de ressources sur le réseau pour essayer d'éviter ces déconnexions, soit consommer ces ressources pour permettre aux systèmes de mieux gérer ces déconnexions et de continuer à fonctionner malgré les inconvénients qu'elles induisent. Donc, plus un ordinateur sera indépendant, (puissance, données), mieux il pourra supporter les déconnexions du réseau **[10]**.

Le système Coda **[11]** fournit un bon exemple pour la gestion des déconnexions, bien qu'il soit prévu pour les *notebooks* actuels où les déconnexions sont beaucoup moins fréquentes, plus prévisibles et plus longues qu'en environnement mobile **[5]**. Des informations sur le profil utilisateur sont conservées pour optimiser les mécanismes de cache de fichiers en choisissant les fichiers les plus appropriés.

b) La faible largeur de la bande passante des médiums sans fil

L'environnement de calcul mobile est davantage concerné par les problèmes de limitation et de consommation de largeur de bande passante que l'environnement fixe ; les réseaux sans fil ayant des largeurs de bande plus petites que les réseaux filaires. Les débits crête pour les communications sans fil sur portables atteignent seulement un Mega bit par seconde pour l'infrarouge, deux Mega bits par seconde pour la radio, neuf à quatorze Mega bits par seconde pour la téléphonie cellulaire, tandis qu'un réseau Ethernet fournit dix Mega bits par seconde, FDDI cent Mega bits par seconde et ATM cent cinquante cinq Mega bits par seconde, et même un réseau sans fil non portable comme le Motorola Altair fournit 5,7 Mega bits par seconde **[10]**.

En plus de ces limitations déjà existantes, il faut se souvenir que, dans une cellule, la largeur de bande est partagée entre les différents utilisateurs. La largeur de bande disponible pour chaque utilisateur offre donc un meilleur aperçu de la capacité du réseau que la largeur de bande totale. Cette mesure dépend en effet du nombre et de la distribution des mobiles.

Pour augmenter la capacité du réseau, il est possible d'installer davantage de cellules de communication sans fil sur une zone donnée. Il existe deux techniques: soit l'on fait se recouvrir des cellules fonctionnant sur des longueurs d'onde différentes, soit l'on

réduit les zones de couverture de chaque cellule **[5]**. Ces techniques autorisent toutes deux davantage de communications simultanées dans une même zone géographique.

c) La variabilité des besoins utilisateur

La mobilité des utilisateurs est également synonyme de plus grandes variations dans les besoins de bande passante qu'en environnement statique. Les besoins en largeur de bande passante peuvent se voir multipliés par quatre dans un réseau sans fil. Les fluctuations de trafic sur réseaux fixes atteignent rarement de telles variations **[5]**. Une application peut supporter cette variabilité de trois manières différentes **[10]**. Elle peut :

- Supposer toujours disposer de connexions á large bande et n'opérer que via des liens de communication physiques ;
- Supposer disposer de connexions á faible bande et ne pas tirer avantage des plus grandes largeurs disponibles ;
- S'adapter dynamiquement aux ressources disponibles fournissant ainsi à l'utilisateur un niveau de qualité variable, mais qui tire parti au maximum des capacités réelles du réseau.

1.5.2 La mobilité

Le but de l'informatique mobile est de fournir à l'utilisateur une complète indépendance de mouvement, tout en conservant une transparence à la localisation variable en fonction des besoins. Cette indépendance peut passer par différents concepts de mobilité, qu'il est éventuellement possible de mettre en œuvre conjointement. Trois concepts de mobilité peuvent être distingués : la mobilité du matériel qui est à l'heure actuelle la plus fréquemment citée dans les articles sur les systèmes mobiles **[10, 13, 14]**, la mobilité des utilisateurs seuls **[15]** et la mobilité des interfaces des applications **[16]**.

La mobilité du matériel et de l'utilisateur avec lui constitue pour le moment le problème le plus étudié en environnement mobile. Les problèmes posés sont d'ordre multiple et couvrent aussi bien le domaine des réseaux (adressage, protocoles de transport) que la gestion des données (accès, stockage, cohérence) ou les mécanismes

systèmes (algorithmes de base, accès aux services et aux ressources). La mobilité est un comportement ayant des implications aussi bien pour les stations qui ne bougent pas et qui constituent le réseau fixe, que pour les machines qui se déplacent avec les utilisateurs et qui constituent le réseau sans fil ou mobile **[13]**.

Une notion de mobilité complémentaire à celles du matériel et des utilisateurs et encore relativement peut explorer, correspond au cas où les interfaces des applications aussi bien que les ordinateurs sont à même de se déplacer. Le but dans ce type de mobilité est de permettre à l'utilisateur de disposer de ses applications actives quelle que soit sa nouvelle localisation.

1.6 Les problèmes liés à la mobilité

L'environnement mobile, actuellement en rapide extension, entraîne de nouveaux problèmes qui sont causés par les nouvelles caractéristiques du système mobile. Ceci nécessite des mécanismes spécifiques pour s'adapter aux limitations qui existent, ainsi qu'aux facteurs qui rentrent en jeu lors de la conception.

1.6.1 L'hétérogénéité du matériel et des réseaux

Contrairement aux systèmes répartis classiques (fixes) où les machines sont connectées une fois pour toute à un réseau donné, dans un environnement mobile, se rencontre non seulement une multitude de types de sites mobiles mais également un grand nombre de réseaux à la fois avec ou sans fil. Les mobiles se retrouvent donc à naviguer dans un environnement hautement hétérogène et de même les réseaux risquent de voir passer un grand nombre de machines de tout ordre et d'utiliser un nombre important de protocoles d'accès différents.

Les interfaces de communication sans fil risquent également de changer lors de déplacements entre l'intérieur et l'extérieur. Par exemple, les infrarouges relativement sensibles aux rayonnements solaires sont plus facilement utilisés à l'intérieur. Et même si une interface radio est conservée, les protocoles sont susceptibles de changer lorsque nous passons d'une couverture cellulaire à une couverture satellite. La gestion de cette hétérogénéité implique des traitements plus complexes en environnement mobile qu'en environnement traditionnel (fixe) **[10]**.

Cependant, les performances sont limitées par le poids et la taille du mobile puisqu'il doit être facile à transporter. Donc, de nouvelles techniques sont nécessaire pour maximiser à la fois les performances et la disponibilité des sites mobiles.

1.6.2 Les risques de la sécurité

Précisément puisqu'il est facile de se connecter à un lien sans fil, la sécurité des communications sans fil peut être compromise bien plus facilement que celle des communications avec fils, tout spécialement lorsque les transmissions se font sur de très grandes distances (interception, génération de messages). Il est donc nécessaire d'inclure des mécanismes sécuritaires aux réseaux sans fil. La sécurité sur ce type de communications est d'autant plus complexe si les utilisateurs sont autorisés à traverser des domaines de sécurité de différents pays. Plus localement, par exemple, s'ils traversent différents services dans un hôpital, ou les bornes de divers réseaux.

Les problèmes rencontrés ont donc trait à l'usurpation d'identité, au refus de service, à l'écoute ou encore à la surveillance des déplacements des mobiles **[12]**. A la fois les mobiles et les réseaux fixes qu'ils visitent, doivent donc être protégés contre tous ces problèmes. Cette protection passe à l'heure actuelle par deux principes, l'authentification et le respect de l'anonymat (confidentialité des informations).

1.6.3 Le nommage de sites mobiles et le routage

Les utilisateurs se déplaçant, leur ordinateur mobile utilise différents points d'accès réseau, autrement dit, des "adresses". Actuellement, les protocoles réseau ne permettent pas des changements d'adresses dynamiques et les connexions réseau actives ne peuvent pas être émigrées vers de nouvelles adresses.

Les adresses tendent à désigner des machines précises, et non pas seulement une interface de connexion. Une fois une adresse est connue dans un système, pour un certain nom de machine, elle est conservée dans un cache pour un temps relativement long et il n'existe aucun moyen d'invalider les entrées obsolètes. Dans *Internet Protocol* (IP) par exemple, un nom de machine IP est lié à son adresse réseau et à sa localisation. Se déplacer vers une autre position implique l'acquisition d'une nouvelle adresse IP, et donc en général, une intervention humaine pour coordonner l'utilisation des adresses

[10]. Or, pour communiquer avec un site mobile, il va falloir connaître son adresse, ce qui va créer un certain nombre de nouveaux problèmes.

Lorsqu'un site mobile franchit les limites de son réseau, il doit donc obtenir un nom qui permettra aux messages ou données qui lui sont destinées d'être routées jusqu'au nouveau réseau, ou d'être redirigées à partir de l'ancien domaine. La plupart des protocoles de routage sont basés sur l'hypothèse que les ordinateurs se déplacent rarement. Dans *Internet Protocol* (IP) une adresse IP d'un site dépend du réseau auquel il est connecté, et les paquets de données sont routés en se basant sur le numéro du réseau. Un tel schéma est inadéquat pour les sites mobiles, puisque les sites mobiles ne sont pas connectés de manière permanente au même réseau.

1.6.4 La localisation de ressources et de sites mobiles

Les ordinateurs classiques ne se déplaçant pas, les informations dépendantes de la localisation sont définies statiquement, comme le serveur de nom le plus proche, les imprimantes disponibles et le fuseau horaire. Un des challenges de l'informatique mobile est de gérer intelligemment l'information et de fournir des mécanismes capables d'obtenir la configuration appropriée à la localisation courante **[10]**.

En environnement mobile, les mouvements fréquents des machines impliquent à la fois une gestion appropriée des informations de localisation et des mécanismes de recherche de sites mobiles, de manière à mettre à jour leur position **[18]**. Il est cependant nécessaire de faire la part des choses entre le maintien des localisations et les besoins d'information des mobiles. Pour localiser les mobiles, il est possible de partitionner l'environnement en utilisant une structure hiérarchique **[19, 20]** organisée en stations support (ou station de base) et en différents niveaux de serveurs de localisation. Dans un tel système, les mobiles sont chargés d'envoyer des notifications lors de sorties de cellules aux serveurs de localisation, qui sont eux-mêmes chargés de gérer toutes les mises à jour, d'où des surcharges en messages et une nécessité de partitionnement de l'information **[21, 22]**.

Le but d'une telle structure est de gérer les mises à jour simultanément pour éviter le goulot d'étranglement d'un système de mise à jour centralisé **[12]**. Elle correspond aux méthodes utilisées dans les architectures cellulaires actuelles **[23]** ou celle proposée pour Internet dans **[24]**.

1.6.5 La gestion des données

Du point de vue de la gestion de données, l'informatique mobile engendre des problèmes de récupération de données, dépendance à la localisation, diffusion de données, gestion des déconnexions et accès efficaces aux données (en prenant en considération les problèmes d'énergie) **[5]**. Globalement, les objectifs de recherche se classent, en fonction du fait qu'ils sont dépendants de la mobilité, des déconnexions, des modes d'accès aux données ou du dimensionnement du réseau mobile **[17]**.

Lorsque les utilisateurs posent des requêtes aux différentes bases de données auxquelles ils ont accès, il est nécessaire de connaître leur position, non seulement pour leur envoyer la réponse, mais aussi pour assurer des réponses correctes aux requêtes dépendantes de la localisation. Si la localisation contenue dans la base est incomplète, il va être nécessaire de procéder à la fois à une recherche dans la base de données et à une acquisition de données pendant le temps d'exécution de la requête. Un nouveau modèle de réponse aux requêtes devra être développé pour inclure les méthodes d'acquisition d'information **[20, 19]**.

D'autre part, la mobilité des utilisateurs implique souvent de répliquer les données couramment utilisées. La réplication est en effet le moyen le plus simple pour assurer la transparence aux utilisateurs mobiles. Un utilisateur qui s'est déplacé et qui utilisait auparavant certains fichiers, applications ou services doit pouvoir continuer à y accéder à sa nouvelle position.

Nous avons donc donné un aperçu des problèmes liés aux caractéristiques des environnements mobiles, à savoir les problèmes liés à la communication sans fil et à la mobilité. Nous ne donnerons pas davantage de détails sur ce sujet, nous allons plutôt nous attacher dés à présent aux problèmes des algorithmes système.

Remarque :

Il est à noter qu’il y a deux types de réseau mobile : le réseau mobile nomade avec infrastructure décrit dans le paragraphe 1.3 et le réseau ad hoc sans infrastructure.

1.7 Les mécanismes des systèmes répartis

L'intégration des ordinateurs mobiles avec les réseaux statiques existants introduit un nouvel ensemble de problèmes dans les systèmes répartis. Un site mobile peut se connecter au réseau à partir de différentes localisations à différents moments via un médium de communication sans fil, les mécanismes répartis (algorithmes, systèmes de gestion de fichiers, protocoles de diffusion de messages, etc.) ne peuvent donc plus supposer que les participants restent à une position fixe et universellement connue dans le réseau au cours du temps. Il devient donc nécessaire de revoir la structuration de ces mécanismes pour les adapter à l'informatique mobile.

1.7.1 Les nouvelles contraintes

Les problèmes de communication et de synchronisation dans les systèmes répartis ont depuis longtemps été résolus pour des réseaux ne comprenant que des sites fixes. Dans de tels systèmes, la connectique du réseau ne change pas en l'absence de pannes de liens ou de sites. A l'inverse, dans les systèmes intégrant des sites mobiles, capables de se déplacer tout en conservant leur connexion au réseau active via des techniques de transmission sans fil, la connectique est en constante évolution. De nouveaux problèmes sont donc induits par la mobilité **[25]** :

- ✓ Les mobiles se déplaçant, la connectique du réseau change, et donc la structure logique, utilisée dans de nombreux algorithmes répartis, ne peut plus être définie statiquement ;
- ✓ Pour délivrer un message à un site mobile, il est nécessaire tout d'abord de localiser le site mobile dans le réseau;
- ✓ Les communications entre les mobiles et le réseau fixe passent un médium sans fil qui supporte naturellement la diffusion de messages au sien d'une zone donnée (cellule), mais ne dispose que d'une faible largeur de bande par rapport aux liens filaires;
- ✓ Les sites mobiles ont d'importantes contraintes en terme de puissance et d'énergie et opèrent souvent en mode veille ou complètement déconnecté du réseau ;

- ✓ Les sites mobiles risquent, de plus, de faire très souvent appel aux différents mécanismes d'exécution distante.

Les algorithmes classiques (exclusion mutuelle, élection, terminaison, etc.), les protocoles de communication de groupe et de diffusion de messages ainsi que les systèmes de gestion de fichiers devront tenir compte de ces caractéristiques pour intégrer explicitement la mobilité, fournir des plates-formes à même de gérer efficacement les mobiles, et adaptées aux besoins des utilisateurs **[5]**.

1.7.2 Les structures logiques des algorithmes répartis

Dans les systèmes répartis, un modèle couramment utilisé comme structure sous-jacente des algorithmes est le graphe non orienté où les nœuds peuvent représenter les unités de calcul (sites) et les arêtes représentent les liens de communication physiques entre les sites **[26]**. Un site peut envoyer un message à n'importe quel autre tant qu'il existe une suite d'arcs permettant d'atteindre le destinataire.

Les algorithmes répartis sont souvent basés sur une telle structure logique définie entre les nœuds. Par exemple un arbre, un anneau ou un graphe complet fait implicitement correspondre à un arc un lien ou une suite de liens dans le réseau. La mobilité d'une machine implique que sa localisation par rapport au reste du réseau évolue au cours du temps, la connectique du réseau tout entier est modifiée, et donc la structure logique qui en découle **[25]**. En conséquence, un lien logique entre deux mobiles ne peut plus correspondre à une séquence de liens physiques du réseau.

Dans de nombreux algorithmes distribués (par exemple **[27,37]**), la structure logique crée entre les participants pour assurer la bonne transmission des messages. Le but principal d'une telle structure est de fournir un certain degré d'ordre et de prédictibilité dans les communications entre les participants **[28, 29]**. Les messages échangés en utilisant de telles structures suivent uniquement les chemins logiques prédéfinis. Un site mobile, qui va pouvoir changer de position, va impliquer la mise à jour de la structure. Ce modèle ne convient donc pas aux sites mobiles. Il va être nécessaire de revoir l'utilisation de telles structures logiques. Pour illustrer ce fait, nous pouvons prendre l'exemple d'un anneau logique entre trois sites mobiles (figure 1.3).

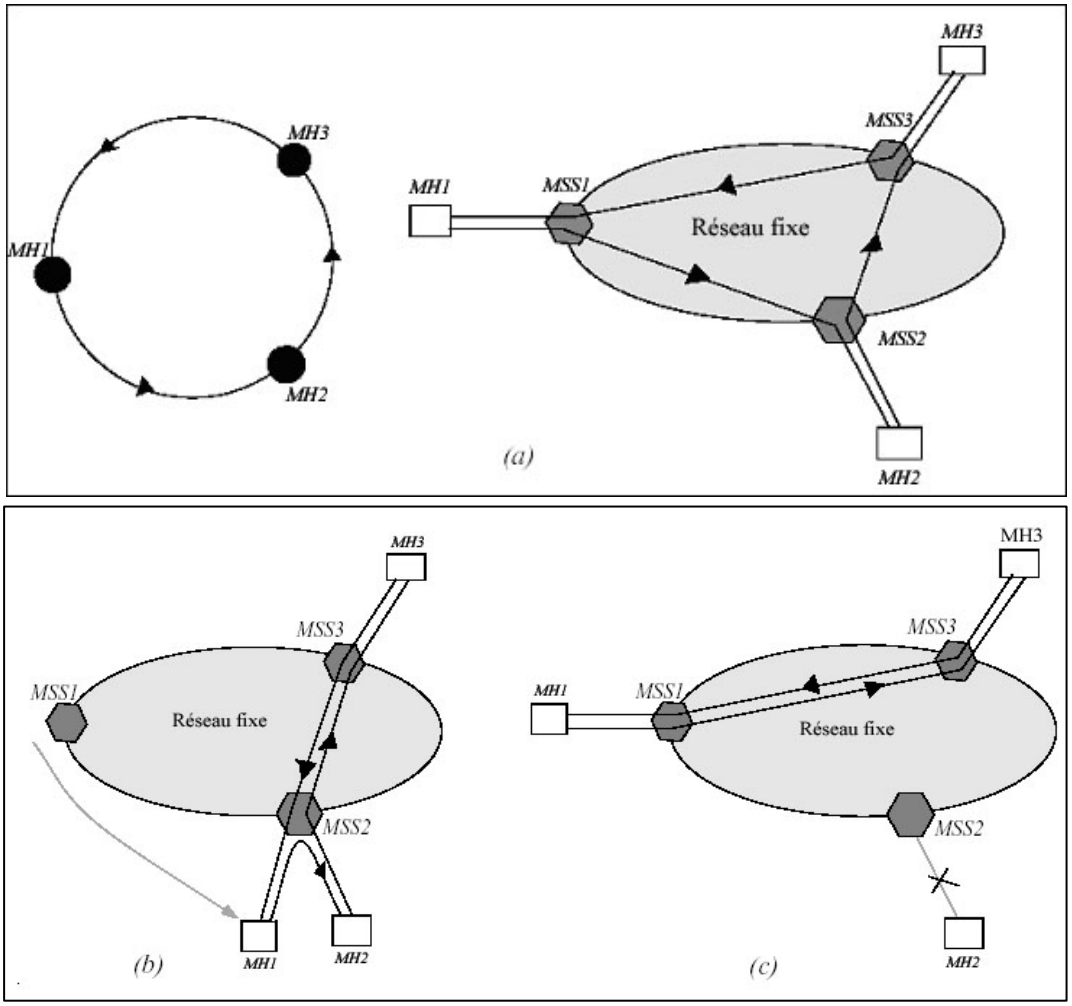


Figure 1.3 – *Effets de la mobilité sur un anneau logique unidirectionnel*

Pour un site mobile *MH* qui va pouvoir changer sa localisation dans le réseau, l'implémentation d'un lien logique nécessite d'être mise à jour à chaque déplacement des sites mobiles. Pour illustrer ce fait, considérant le cas d'un anneau logique entre trois sites mobiles MH_1, MH_2, MH_3 (Figure 1.3 (a)). Dans cet exemple, un seul lien logique dans l'anneau, exp. entre MH_1 et MH_2, se compose de: (1) la connexion sans fil entre MH_1 et sa *MSS* locale courante, MSS_1, (2) une connexion point-à-point entre MSS_1 et MSS_2, la *MSS* locale de MH_2 et (3) une connexion sans fil entre MH_2 et MSS_2. Si un mobile change de position, sa station support change également, l'anneau logique doit donc être reconfiguré (Figure 1.3 (b)), de même si le mobile se déconnecte ou même passe en mode veille (Figure 1.3 (c)).

L'impact de la mobilité sur les structures logiques se traduit par des reconfigurations à chaque déplacement d'une machine. Globalement, pour autoriser la conservation d'une structure logique dans un algorithme, les bénéfices apportés par

cette structure doivent justifier les coûts de maintien induits par les déplacements des sites mobiles.

En implémentant la structure logique uniquement entre les stations support **[25]** (Figure 1.4), il est possible d'obtenir les bénéfices escomptés par une structure logique, sans pour autant être confrontés aux inconvénients dus aux sites mobiles et aux besoins de reconfigurations fréquentes de la structure.

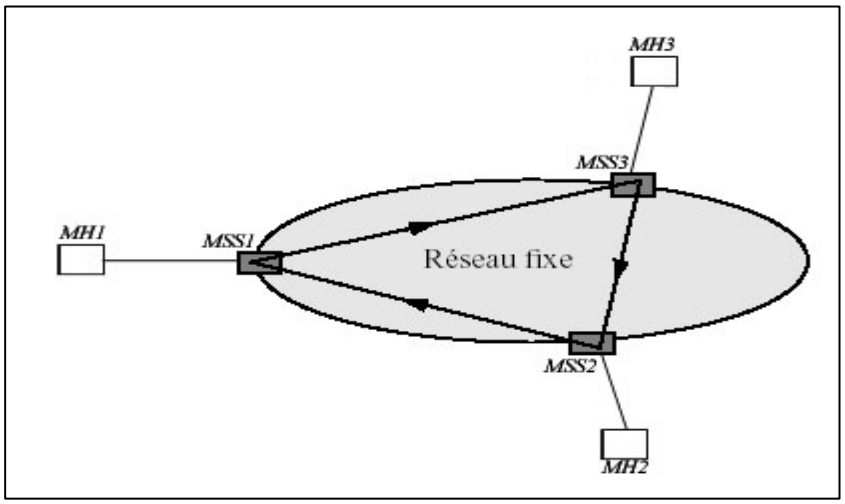


Figure 1.4 – *Anneau logique sur stations support*

1.7.3 Localisation et coût de recherche des sites mobiles

Pour envoyer un message à partir d'un site mobile MH_1 vers un autre site mobile MH_2, il est nécessaire que MH_1 transmette le message à sa station support locale à travers le réseau sans fil. Si les deux mobiles se trouvent dans la même cellule, la station retransmet directement le message au site destinataire, MH_2, **[28, 29]**. À l'inverse, si les deux sites mobiles ne se trouvent pas dans la même cellule, il va être nécessaire de localiser le mobile destinataire pour pouvoir faire suivre le message à la station support appropriée **[28, 29]**. La station distante ainsi trouvée sera alors chargée de la retransmission au mobile destinataire.

Ce besoin de localisation implique des coûts de recherche et de mise à jour des données de localisation des mobiles **[5]**. Le coût de communication induit dépend du coût lié à la distance à parcourir sur le réseau fixe pour atteindre le destinataire, du coût de communication sans fil et du coût de recherche des mobiles impliqués **[28, 29]**.

1.7.4 Connexion sans fil

Un inconvénient des communications sans fil est que la largeur de bande passante est relativement réduite par rapport à celle disponible pour les liaisons filaires. Par contre, les communications sans fil supportent physiquement la diffusion à l'intérieur d'une cellule, une station de base peut émettre un message à toutes les unités mobiles localisées dans sa cellule, en une seule opération de transfert. L'exploitation de ce mode de transmission permettra de diminuer les coûts des envois de données **[5]**.

Il est cependant important de tenir compte des problèmes classiques liés à la diffusion de messages dans des groupes de machines ou de processus. En l'occurrence, il faut assurer, la réception des messages par tous les membres du groupe, et ceux-ci exclusivement. Puis, selon les besoins, il est nécessaire de respecter un ordre au sein du groupe de communication (par exemple, réception selon un ordre quelconque identique pour tous les membres du groupe, selon l'ordre d'émission, l'ordre causal, etc.). Ces problèmes sont bien connus en informatique fixe, les résultats déjà obtenus peuvent donc être exploités avec bénéfice dans le cas de l'environnement mobile.

Les délais de transfert des messages entre les sites fixes étant arbitraires, un message émis par une même unité mobile peut être reçu à des instants différents par des stations de base différentes. Il est donc possible qu'un message diffusé dans une cellule puisse ne pas être reçu par une unité mobile destinataire, car son entrée dans la cellule a eu lieu après la diffusion du message ou bien sa sortie de celle-ci a eu lieu avant la diffusion **[25]**. En revanche, une unité peut recevoir plusieurs copies du même message, issues de stations de base différentes. Dans la figure 1.5, MH_1 se déplace de MSS_3 vers MSS_1, en perdant le message dans les deux cellules, cependant MH_2 reçoit deux copies du message dans MSS_4 et MSS_3. On trouve dans **[30]** une approche de mise en œuvre de la communication multi-destinataire dans un environnement mobile qui garantit la réception d'exactement une copie du même message par tous les destinataires. Notons par ailleurs que les transmissions de messages à partir de sites mobiles consomment davantage de puissance que les réceptions pour un message de taille identique **[30]**. La communication au sein d'une cellule doit de préférence être asymétrique de manière à exploiter au mieux la capacité de diffusion des communications sans fil et de réduire la consommation d'énergie sur les mobiles.

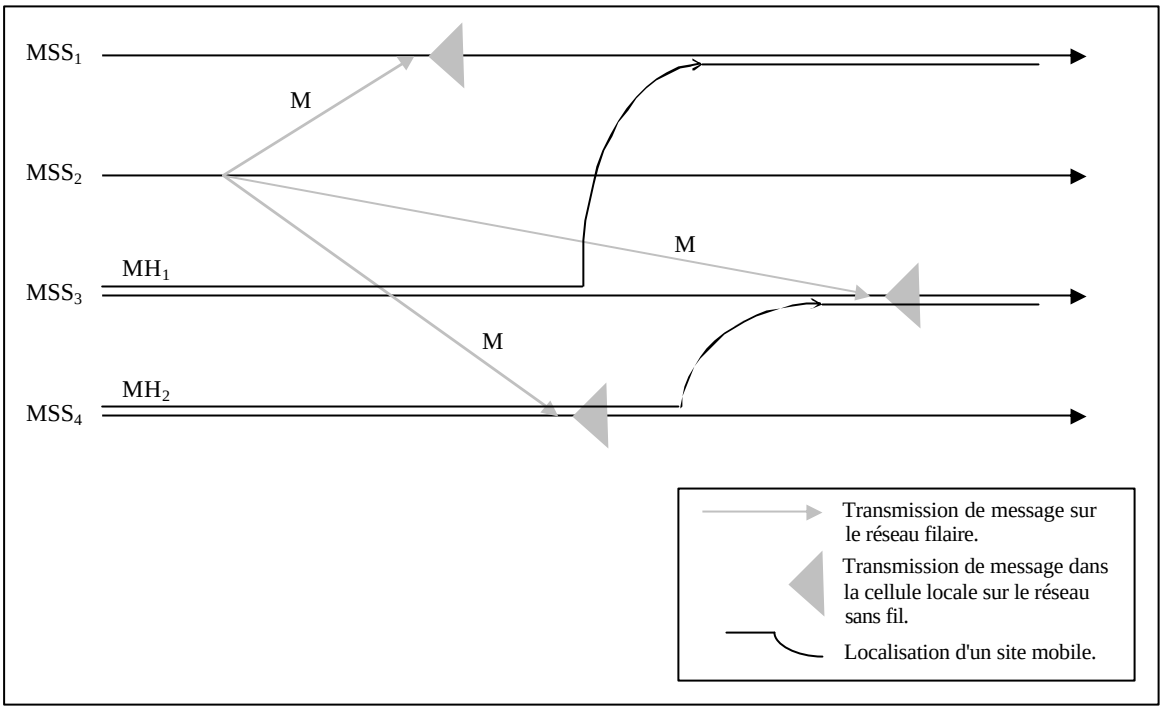


Figure 1.5 – *la communication sans fil et la mobilité*

1.7.5 Déconnexion et mode veille

Les sites mobiles sont souvent déconnectés du reste du réseau, il ne peut plus, par période, ni envoyer ni recevoir de messages. De même, s'ils se déconnectent au milieu d'une exécution d'un algorithme réparti, ils risquent de bloquer toute l'exécution jusqu'à sa reconnexion. Il est donc nécessaire, en environnement mobile, de bien distinguer les déconnexions (volontaires) des pannes (involontaires). Les déconnexions doivent être considérées comme des caractéristiques normales des mobiles, et étant prévisibles, il doit donc exister un protocole de déconnexion ainsi qu'un protocole de changement de cellule (*handoff*) pour assurer une poursuite correcte des exécutions auxquelles le mobile participait. De tels protocoles permettent à la fois au participant en cours de détachement de décharger des données pour continuer les exécutions locales, et d'envoyer à la station support les informations donc elle aura besoin pour poursuivre l'exécution du programme, le mobile étant par la suite inaccessible **[28, 29]**.

Un nouveau mode à prendre en compte correspond au mode veille dans lequel la vitesse de l'horloge est réduite et aucune exécution utilisateur n'est réalisée. Le site mobile attend alors simplement la réception d'un message pour reprendre son mode de fonctionnement normal. Comme dans le cas des déconnexions, ce mode de

fonctionnement est une opération volontaire, cependant les implications sont différentes. Dans le mode veille, un site mobile peut encore être atteint par le reste du système et donc peut être amener par le système à reprendre son mode de fonctionnement normal, alors que les déconnexions et reconnexions ne peuvent être initialisées que par le mobile lui-même.

Le mode veille est mal supporté par les algorithmes totalement décentralisés qui exigent la participation de chaque site mobile; de tels algorithmes empêchent ces sites mobiles d'opérer en mode veille même s'ils ne sont pas impliqués dans le calcul. Par conséquent les tentatives de conservation d'énergie dans ces sites mobiles en opérant en mode veille sont complètement contrariées **[31]**.

Un mécanisme réparti qui comprend des sites mobiles ne doit pas les contraindre à participer à chaque exécution, mais les autoriser à passer en mode veille ou en mode déconnecté. Si l'algorithme est exécuté directement sur des mobiles, il doit donc tenir compte des éventuelles déconnexions et devra fournir aux autres participants les informations nécessaires à la poursuite correcte de l'exécution. Ceci passe par l'adaptation des différents algorithmes et applications réparties, c'est à dire par ajout de protocoles de déconnexion, de chargement et déchargement de données, messages d'information, etc. **[5]**

Dans **[28, 29]** Badrinath et al. analysent l'impact de la mobilité sur un ensemble d'algorithmes répartis classiques d'exclusion mutuelle **[32, 27]**. L'idée qui sous-tend cette analyse est d'arriver à un découplage entre la mobilité des sites et la construction des algorithmes pour des environnements mobiles **[31]**. Pour cela, les algorithmes répartis sont structurés de manière à ce que la majeure partie des communications et du calcul induit par l'algorithme soit prise en charge par le réseau statique. La même démarche est suivie par **[33]** pour adapter l'algorithme de **[34]** pour la mise en œuvre de l'ordre causal dans un environnement mobile.

Pour illustrer cette approche, nous reprenons ci-dessous les deux variantes de l'algorithme réparti de la circulation d'un jeton dans un anneau logique telles que présentées dans **[2]**. La première variante est une mise en œuvre de l'algorithme sans tenir compte des particularités de l'environnement mobile et la deuxième variante est une restructuration de l'algorithme pour l'adapter à ce type d'environnement.

1.7.6 Etude de cas : Algorithme de circulation de jeton sur un anneau logique **[2]**

Un algorithme fondamental dans les systèmes distribués consiste dans la circulation d'un jeton entre les participants d'un anneau logique. Chaque participant se comporte comme suit :

- Attendre la réception du jeton à partir de son prédécesseur dans l'anneau ;
- Entrer dans <la section critique>, si désirer ;
- Envoyer le jeton à son successeur dans l'anneau.

L'algorithme satisfait deux propriétés importantes :

1. L'exclusion mutuelle est trivialement garantie pour le propriétaire courant du jeton, et
2. Il permet un accès équitable au jeton en permettant à chaque participant d'accéder au jeton au plus une fois pour une circulation donnée du jeton.

Cet algorithme a été utilisé à diverses fins telles que la détection de la terminaison **[35]**, l'exclusion mutuelle **[27]**, les protocoles d'ordonnancement de messages **[36]** dans les systèmes fixes. Dans un environnement mobile, le jeton peut être utilisé comme un mécanisme pour l'accès distant à une ressource partagée.

1.7.6.1 Algorithme R-MH

L'algorithme R-MH est une implémentation directe de l'anneau logique entre les sites mobiles, et représente le cas extrême pour l'exécution d'un algorithme existant sans se soucier de la mobilité des sites et des contraintes ressources associées. Malgré que la correction de l'algorithme soit assurée par cette approche, les contraintes de ressources spécifiques à l'environnement mobile ne sont pas prises en considération :

- *Coût de recherche élevé*

  Tous les messages de l'algorithme doivent être émis à chaque site mobile, ce qui fait entraîner un coût de recherche. Puisque l'algorithme fait circuler le jeton entre tous les MH_s, le coût de recherche total entraîné par l'algorithme est proportionnel au nombre des MH_s.

- *Usage excessif des liens sans fil*

  L'émetteur ainsi que le récepteur de chaque message est un *MH* ; le message est donc transmis à travers les liens sans fil entre les deux MH_s émetteur et récepteur, et leurs MSSs locales respectives.

- *Consommation d'énergie dans les MH_s*

  Chaque message dans cet algorithme consomme de l'énergie dans l'émetteur ; pour le transmettre à sa *MSS* locale, ainsi que dans le récepteur ; pour recevoir le message à partir de sa *MSS* locale.

- *Les modes veille et déconnecté*

  L'algorithme R-MH exige la participation de chaque MH pour maintenir la structure de l'anneau logique et par conséquent ne permet à aucun MH de se déconnecter. D'autre part, l'algorithme R-MH ne permet pas à un site mobile de se mettre à l'état veille sans être interrompu même s'il n'a pas besoin d'accéder à la ressource partagée (représentée par le jeton) ; il doit assurer la réception et l'expédition du jeton pour permettre aux autres MH_s d'accéder à la ressource.

Il est important de souligner ici que les inconvénients cités ci-dessus ne sont pas intrinsèques à l'algorithme, mais découlent plutôt d'une application inadéquate de l'algorithme, i.e. l'anneau logique est établit entre les sites mobiles qui sont caractérisés, contrairement aux sites fixes, par une connectivité physique qui nécessite d'être redéfinie suite à chaque déplacement d'un site mobile. De plus, les sites fixes ne souffrent pas des contraintes de consommation d'énergie et des connexions sans fil à bande passante limitée, par conséquent une solution à ces inconvénients est d'appliquer l'algorithme à l'environnement mobile selon le principe deux-tiers.

1.7.6.2 Restructuration de l'anneau logique selon le principe deux tiers

Le principe deux-tiers suppose que l'anneau logique soit établi sur le réseau fixe entre toutes les MSS_s avec le jeton visitant chaque *MSS* dans une séquence prédéfinie. Un site mobile qui désire accéder au jeton a besoin de soumettre une requête à sa *MSS* locale. Quand le jeton visite cette *MSS*, toutes les requêtes en attente sont servies. Cependant, un *MH* peut changer sa localisation après l'émission de sa requête, et donc la localisation d'un tel *MH* exige d'être explicitement gérée. Deux stratégies de gestion de la localisation ont été proposées, la stratégie *Chercher* et la stratégie *Informer*.

Stratégie *Chercher*

Les actions exécutées par une MSS M

- A la réception d'une requête pour l'obtention du jeton à partir d'un *MH* local, *M* insère la requête à la queue de sa file *request*.
- Quand *M* reçoit le jeton à partir de son prédécesseur dans l'anneau logique, elle exécute les étapes suivantes:

1. fait un transfert des requêtes de la file *request* vers la file *grant*.
2. *Répéter*
   - Défiler la requête de la tête de file *grant*;
   - Si le *MH* qui a établi la requête est local à *M*, alors délivrer le jeton à *MH* via le lien sans fil;
   - Sinon, chercher et délivrer le jeton à *MH* dans sa cellule courante;
   - Attendre le retour du jeton à partir de *MH*.

   Jusqu'à ce que grant soit vide
3. rediriger le jeton vers le successeur de *M* dans l'anneau logique

Les actions exécutées par un MH h

- Quand *h* nécessite l'accès au jeton, il émet une requête à sa *MSS* courante, dite *M*, et
- La MSS où *h* émet sa requête envoiera éventuellement le jeton à *h*. Après que *h* accède à la section critique, il retourne le jeton à la même *MSS*.

Cet algorithme assume qu'un *MH* n'émet une deuxième requête que si sa dernière requête est servie. De plus, quand un *MH* reçoit le jeton, il doit le remettre à la *MSS*

émettrice après avoir accéder à la section critique, i.e. il ne peut pas se déconnecter de manière permanente après la réception du jeton.

Stratégie *Informer*

Une alternative de la stratégie *Chercher* pour localiser un *MH* migrant est d'exiger au site mobile de signaler à la *MSS* (d'où il émet sa requête) chaque changement dans sa localisation jusqu'à la réception du jeton.

Si un site mobile *h* change de cellules moins souvent après l'émission de sa requête, alors il est préférable pour *h* d'informer *S* de chaque changement dans sa localisation au lieu que *S* recherche *h*.

Les actions exécutées par une MSS M

- A la réception d'une requête pour lobtention du jeton à partir d'un *MH* local *h*, *M* insère la requête <*h*, *M*> à la queue de sa file *request*.
- A la réception d'un message *inform*(*h*, *M'*), la valeur courante de *locn*(*h*) est remplacée par *M'* dans l'entrée <*h*, *locn*(*h*)> dans la file *request* de *M*.
- Quand *M* reçoit le jeton à partir de son prédécesseur dans l'anneau logique, elle exécute les étapes suivantes:

1. fait un transfert des requêtes de la file *request* vers la file *grant*.
2. *Répéter*
   - Défiler la requête <*h*, *locn*(*h*)> de la tête de file *grant*;
   - Si *locn*(*h*) == *M*, alors délivrer le jeton à *MH* via le lien sans fil;
   - Sinon, rediriger le jeton à *locn*(*h*), i.e. à la *MSS* courante de *h* qui transmettra le jeton à *h*;

   Jusqu'à ce que grant soit vide
3. Rediriger le jeton vers le successeur de *M* dans l'anneau logique

Les actions exécutées par un MH h

- Quand *h* nécessite l'accès au jeton
  - il émet une requête à sa *MSS* courante, dite *M,* et
  - sauvegarde l'identité de *M* dans la variable local *req_locn.*
- Quand *h* reçoit le jeton à partir de la *MSS req_locn*, il accède à la section critique, retourne le jeton à la même *MSS* et remet alors *req_locn* à ⊥.
- Après chaque déplacement, *h* émet le message *join*(*h, req_locn*) à sa nouvelle *MSS M'.*
  - Si *req_locn* reçu avec le message *join*() est différent de ⊥, alors *M'* envoie un message *inform*(*h, M'*) à la *MSS req_locn*.

1.8 Conclusion

Dans ce chapitre, nous avons présenté plus en détails les environnements mobiles, leurs caractéristiques et leurs problèmes. Tout d'abord, nous avons pu constater que, du fait des communications sans fil, les sites mobiles diffèrent de ceux des stations fixes. Un site mobile pourra non seulement être connecté ou déconnecté du réseau, mais il pourra également économiser de l'énergie en passant en mode veille, ou encore subir une chute de la capacité de la bande passante et ne plus opérer qu'en étant partiellement connecté.

En suite, nous avons pris connaissance de la multitude de problèmes causés par les nouvelles caractéristiques de l'environnement mobile. Principalement les problèmes liés à la connexion sans fil et à la mobilité. Nous avons aussi présenté l'impact de ces caractéristiques sur les mécanismes systèmes existants : les implications sur les algorithmes répartis qui nécessitent une restructuration pour tenir compte des caractéristiques de l'environnement mobile.

Enfin, ce chapitre donne donc un aperçu des problèmes des environnements mobiles et des communications sans fil, problèmes qui se répercutent bien évidemment sur tout développement au sein de ces environnements, aussi bien sur les applications que sur les différents mécanismes système.

Chapitre 2

Une unité mobile comme étant un message

2.1 Introduction

Le modèle typique du calcul distribué traite un réseau comme étant un graphe dans lequel les cercles représentent les processus nœuds et les arêtes représentent les canaux de communication. Les défaillances peuvent rendre des parties du réseau non-opérationnelles de façon soit temporaire ou permanente. Malgré les défaillances, la structure dans son ensemble est considérée être *statique*.

Une manière d'introduire la mobilité dans un modèle similaire est de traiter les nœuds comme des *centres support mobiles* (Mobile Support Centers **MSC**) qui coordonnent entre diverses stations de base radio. Les unités mobiles sont autorisées seulement de se connecter et de se déconnecter des MSCs à travers les communications avec les stations de base. Le modèle résultant est un noyau fixe de nœuds statiques qui représentent les cellules individuelles et de canaux qui représentent l'aptitude des unités mobiles de se déplacent d'une cellule à une autre.

Le fait qu'aujourd'hui le *calcul nomade* devient dominant en environnement mobile, ceci permet l'application de ce modèle à cause de l'évolution rapide des technologies courantes des réseaux **[38, 39]**. Ce modèle s'applique aussi à la *mobilité logique* où les nœuds représentent les services disposés aux agents hôtes mobiles et les arêtes représentent les canaux de communication sur lesquels les agents mobiles peuvent migrer.

Tant que la mobilité demande une nouvelle perspective du calcul distribué, il est également impératif d'investir n'importe quelle caractéristique essentielle de la

mobilité. En effet, *le terme **mobilité** a été utilisé toujours pour faire référence aux processus qui migrent à travers le réseau* **[40, 41]**. Amy Murphy suggère encore une autre manière de penser à la mobilité dans le contexte de la structure d'un graphe fixe traditionnel.

L'idée de base est de traiter les unités mobiles comme des messages nomades qui préservent leurs identités lorsqu'ils traversent le réseau. Plusieurs applications pratiques peuvent être adaptées à ce genre de modélisation. Un téléphone cellulaire, par exemple, peut passer d'une cellule à la prochaine. Donc, tant qu'il opère à l'intérieur d'une cellule, le téléphone peut être vu comme un résidant à un nœud dans le réseau support ; de même, le protocole de *handover* (provoquer par la détection d'une dégradation du signal) peut être modéliser comme la traversé d'un canal entre deux nœuds représentant des cellules individuelles. Les voix de transmissions entre deux téléphones sont aussi modélisées par des massages. La mobilité logique est naturellement représentée par ce modèle et ce avec des agents prenant le rôle des messages persistants qui se déplacent entre les hôtes (hosts).

L'intérêt dans ce modèle est qu'il assure *sa capacité de faciliter l'application des algorithmes distribués déjà établis aux problèmes du calcul mobile* **[43]**. Les algorithmes résultants peuvent alors être utilisés pour supporter la conception des abstractions de haut niveau, qui peuvent être fournies aux programmeurs d'application afin de les aider dans le développement de leurs applications.

En observant l'infrastructure de la mobilité comme un graphe de nœuds et de canaux, nous avons un modèle similaire au modèle du système distribué filaire où les nœuds sont les processus et les liens de communication entre eux sont les canaux. Cependant, une observation immédiate est que plusieurs algorithmes distribués classiques peuvent être exécuter dans le cadre mobile avec certaines petites modifications. Il est à noter qu'à cause de certaines propriétés uniques à la mobilité telles que la déconnexion et la largeur de la bande limitée, il n'est pas pratique de faire une telle translation directe. Amy Murphy et al proposent une direction fondamentalement différente.

Les travaux précédents se sont intéressés à la transformation des algorithmes distribués pour résoudre les mêmes types de problèmes dans le domaine mobile, tandis que le travail de Amy Murphy et al adapte les algorithmes distribués pour résoudre des problèmes liés (propres) à l'environnement mobile. L'exécution de l'algorithme ne change pas mais nécessairement sa sémantique change.

Dans ce chapitre, nous allons présenter une nouvelle manière d'introduire la mobilité et ce en traitant les unités mobiles comme étant des messages nomades qui préservent leurs identités lorsqu'ils traversent le réseau. Nous allons voir tout d'abord, comment cette notion peut être appliquer aux deux types de la mobilité : physique et logique. Par la suite, le chapitre met l'accent sur un problème majore qui s'impose avec ce genre de modèle qui est la livraison de messages entre les couples des unités mobiles (physique ou logique). Pour cela, et afin de mieux voir le problème fondamental dont les schémas typiques de livraison de messages souffre, nous nous présenterons deux approches de livraison de messages.

2.2 Le modèle

Dans cette partie, nous allons voir comment ce nouveau modèle peut être appliquer aux deux types de la mobilité : physique et logique.

a) La mobilité nomade

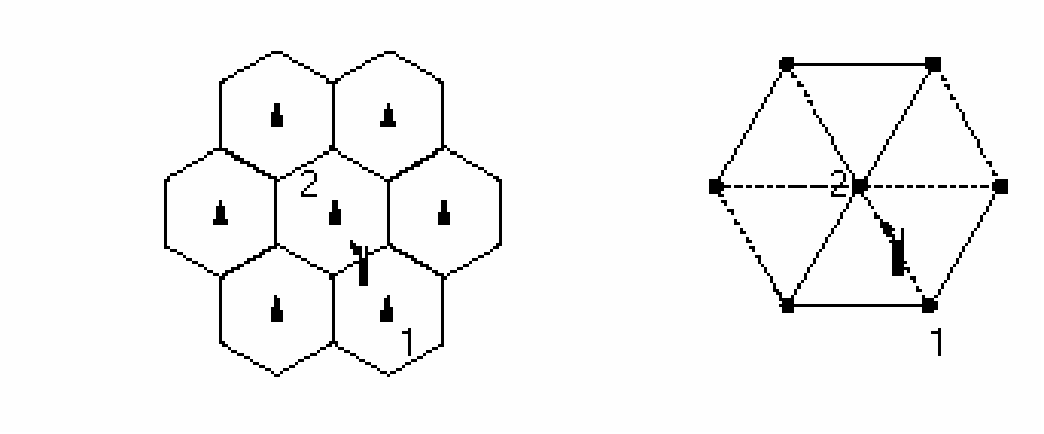
La conception du téléphone cellulaire fournie la fondation pour le modèle de la mobilité nomade que Amy Murphy et al ont adopté dans leur modèle. La figure 2.1(a) montre un modèle de téléphone cellulaire typique avec un seul centre support mobile (MSC) dans chaque cellule. Le MSC est responsable des communications avec les unités mobiles à l'intérieur de leurs régions et il sert comme un manager pour les requêtes de handover lorsqu'un mobile se déplace entre les MSCs. La figure 2.1(b) montre comment le modèle de téléphone cellulaire est transformé en un graphe de nœuds et de canaux où les nœuds représentent les cellules individuelles et les canaux représentent l'aptitude d'une unité mobile de se déplacer d'une cellule à une autre.

Pour simplifier, Amy Murphy et al supposent que :

- Le réseau résultant est connecté, en d'autres termes, un chemin existe entre chaque couple de nœuds.

- Une unité mobile déplaçant entre deux MSCs peut être modéliser comme étant un message en transite sur un canal identique. De cette façon, on ne différencie pas largement entre les mouvements physiques et les communications filaires.

- Il est raisonnable de dire : Que se passe-t-il lorsque les messages et les unités mobiles se trouvent sur le même canal ? ⇒ La réponse sera : Tous les canaux préservent l'ordre des messages, c.-à-d., ce sont des canaux FIFO.



(a) **(b)**

Figure 2.1 – *(a) Système cellulaire avec un MSC par cellule. Tous les MSCs sont supposés être connectés par un réseau filaire.*
(b) Modèle abstrait d'un système cellulaire comme un graphe de nœuds et de canaux. Les lignes en gras forment un arbre de recouvrement.

b) La mobilité logique

Le modèle de la mobilité logique peut être vu comme un graphe de réseau typique où les nœuds représentent les serveurs disposés aux agents hôtes et les arêtes représentent les canaux FIFO sur lesquels les agents peuvent migrer et les messages peuvent être passer.

Le graphe de réseau est supposé être connecté (c.-à-d., un chemin existe entre chaque couple de nœuds), mais pas nécessairement complètement connecté (c.-à-d., un canal n'existe pas nécessairement entre chaque couple de nœuds). La communication entre chaque couple de nœuds est supposée être standard, le passage des messages se fait de manière asynchrone et bornée.

Dans un réseau IP typique, tous les nœuds sont logiquement connectés. Cependant, ce n'est pas toujours le cas au niveau de l'application, comme le montre la figure 2.2. Dans ce cas, un ensemble de sous- réseaux est connecté à un autre à travers un réseau IP, mais un agent peut entrer ou quitter un sous réseau seulement en passant à travers un routeur, par exemple à cause des raisons de sécurité.

En plus, l'agent serveur mobile est supposé avoir le rôle de garder trace des agents traversant le réseau à un moment donné et qu'il fournit un certain mécanisme de base pour livrer un message à un agent, par exemple, en invoquant une méthode de l'objet agent. Finalement, chaque agent a un seul identifiant unique qui peut être utiliser pour diriger un message à un autre agent. Ces dernières suppositions sont toujours satisfaites par la majorité des plat-formes d'agents mobiles.

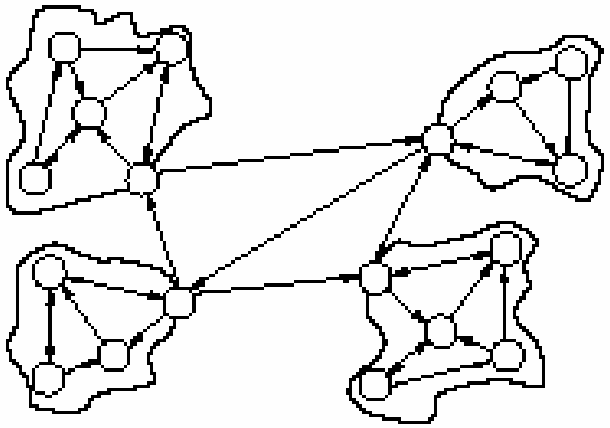


Figure 2.2 – *Un réseau connecté avec des sous- réseaux connectés. Les agents peuvent entrer et quitter les sous- réseaux seulement en passant à travers les routeurs.*

2.3 La livraison de messages

Le problème majore qui s'impose avec ce genre de modèle est la livraison de messages entre les couples des unités mobiles (physique ou logique). Une unité mobile peut envoyer et recevoir des messages seulement quand elle est présente à certain nœud dans le réseau fixe, cette situation modélise l'existence d'une connexion établie entre une unité mobile et un centre support ou un agent mobile s'exécutant sur un hôte. Lorsqu'une unité mobile est sur un canal, elle peut être vue comme étant une unité déconnectée temporairement du réseau et par conséquent incapable de communiquer.

Dans les systèmes distribués typiques, la communication passant un message est manipulé par les routeurs dans le réseau fixe qui capture l'emplacement relatif de tous les services de calcul attachés en se basant sur les adresses IP. Ce modèle ne peut pas être étendu au calcul mobile car l'emplacement des unités mobiles change tout en respectant les routeurs fixes. Cependant, la capacité d'envoyer et de recevoir des messages est critique pour plusieurs applications.

L'utilisation typique du paradigme d'agent logiquement mobile est pour dériver un lien de communication et exploiter l'accès local aux ressources sur un serveur éloigné **[42]**. Ainsi, on pourra argumenter que, la communication avec un agent éloigné n'est pas importante et une plate-forme d'agents mobiles devra se baser plutôt sur les mécanismes de communication qui sont exploités localement. Néanmoins, il existe divers scénarios connus qui exploitent la communication avec ou entre les agents éloignés, quelques-uns entre eux sont liés à un gestionnaire d'agents mobile.

Imaginant un agent *maître* (master) manipuler un nombre d'agents mobiles *esclaves* (slave) qui sont injectés par la suite dans le réseau pour représenter un calcul coopératif, par exemple, trouver une certaine information. A certain point, l'agent maître peut demander d'arrêter le calcul active des agents esclaves, par exemple, car l'information demandée a été trouvée par un d'eux et ainsi il est désirable de terminer l'agent dans le but d'empêcher une consommation inutile des ressources. Or, il peut demander de changer quelques paramètres gouvernant le comportement des agents en réponse à un changement dans le contexte qui détermine sa création. Alternativement, les agents esclaves peuvent demander de détecter si l'agent maître est encore en vie en respectant une sorte de détection orpheline, qui demande de localiser l'agent maître s'il est lui-même autorisé à être mobile.

D'autres exemples s'imposent car les agents mobiles sont juste un des paradigmes disponibles pour concevoir une application distribuée. Dans le contexte de telle application, un mixage entre les agents mobiles et le transfert des messages peut être utiliser pour accomplir des fonctionnalités différentes. Par exemple, un agent mobile peut visiter un site et exécuter une interruption sur une condition donnée. Si la condition n'est pas satisfaite, l'agent peut enregistrer un auditoire d'événements avec le site. De cette manière, quand l'agent mobile visite d'autres sites et avant de donner ses résultats, il peut recevoir les notifications des changements d'états dans les sites déjà visités et décide si une seconde visite est justifiée.

Dans les deux types de mobilité logique et physique, une condition nécessaire pour n'importe quelle communication est ***la fiabilité***. Les primitives de programmation qui garanties que les données envoyées atteins effectivement la cible de communication, sans demander plus d'actions par le programmeur, simplifies beaucoup la tâche du développement et conduit aux applications qui sont plus robustes. Dans les systèmes distribués conventionnels, la fiabilité est accomplie typiquement en fournissant un

certain degré de tolérance aux défaillances dans les liens de communication correspondants ou au niveau des nœuds de communication.

Cependant, les techniques de tolérances aux défaillances ne sont pas suffisantes pour assurer la fiabilité dans les systèmes qui exhibent (exposent) la mobilité. Le fait que les unités mobiles se déplacent librement d'un nœud à un autre selon un certain modèle de migration inconnu, la livraison des données est compliquée. Il est difficile à la fois de déterminer où se trouve l'unité mobile et d'assurer que les données atteins effectivement l'unité mobile avant qu'elle se déplace encore une fois. Si cette dernière condition n'est pas garantie, la perte des données peut arriver. Ainsi, le défi pour les communications sûres persistent même sous les suppositions d'un mécanisme de transport idéal qui lui-même garantit seulement la livraison correcte des données d'un nœud à un nœud malgré la présence des défaillances. *C'est la présence absolue de la mobilité et non pas la possibilité des défaillances qui mine ou attaque la fiabilité* **[43]**.

Définition du problème :

Le problème de livraison fiable des messages peut être maintenant formulé comme suit : *Soit un graphe complètement connecté avec des canaux de communication FIFO et garantis la livraison des messages entre les nœuds, un message localisé à un nœud et une unité mobile pour laquelle le message est destiné. Développer un algorithme distribué qui garanti une livraison unique du message et ne laisse aucune trace du message soit à un nœud ou au niveau de l'unité mobile avec un temps limité après la livraison.* La solution va avoir une limite juste sur le temps de stockage de n'importe quel message au niveau du nœud. **[43]**

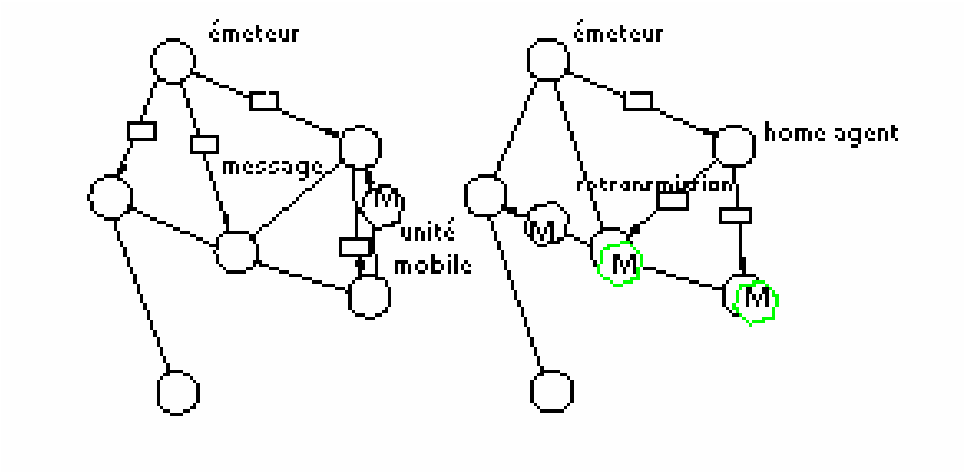
Le fait que les unités mobiles ne puissent pas communiquer directement avec d'autres unités mobiles, le réseau doit fournir un mécanisme pour la transmission des messages. Le message original est supposé être dans la mémoire locale d'un certain nœud, probablement supposé être laissé à ce niveau par une unité mobile qui est la source du message. Puisqu'une unité mobile n'est pas obligée de visiter tous les nœuds pour ramasser ses messages, le message ne peut pas rester isolé à un nœud sur lequel il est arrivé, mais plutôt il doit être distribué à travers le réseau. Les spécifications de ce mécanisme de distribution sont laissées à l'algorithme proposé par Amy Murphy et qu'on va présenter dans le prochain chapitre.

2.4 Les schémas de livraison de messages

Les schémas typiques de livraison de messages souffrent du problème fondamental qu'une unité mobile se trouve en transite durant la livraison et ainsi elle peut être facilement louper. Pour illustrer ce problème, on discute deux approches de livraison de messages : *broadcasting* et *forwarding*.

Un schéma simple de broadcast suppose l'existence d'un arbre de recouvrement de nœuds interconnectés et que n'importe quel nœud peut l'utiliser pour envoyer un message. Un nœud source diffuse une copie du message à ses voisins, et ainsi de suite jusqu'à ce que les nœuds feuilles soient atteins. Cela, cependant, ne garanti pas la livraison du message.

Dans le cas où une unité mobile se déplacerait sur un canal dans le sens inverse de la propagation du message, comme le montre la figure 2.3(a) ou d'une façon plus générale lorsque l'agent quitte sa région en suivant le sens inverse de la propagation du message (du bas vers le haut de l'arbre), dans ce cas l'agent et le message vont se croiser sur le canal par conséquent, la livraison ne va jamais se faire.



a) Spanning tree broadcasting **b) Forwarding**

Figure 2.3 – *Le problème : livraison loupée dans des schémas simples de broadcast et de forwarding.*

Un schéma simple d'envoi (forwarding) maintient un pointeur vers l'unité mobile à un emplacement bien connu, appelé « home agent » dans le protocole mobile IP **[44]** où cette idée permet la mobilité physique des hôtes. A la migration, l'unité mobile doit informer l'agent mère de son nouveau emplacement dans le but de permettre plus de communications. Cependant, n'importe quel message envoyé entre la migration et la modification est perdu, le fait que l'unité mobile quitte son emplacement avant que le message n'atteigne sa destination.

Même si la retransmission du message au nouveau emplacement est arrivée, l'unité mobile peut se déplacer à nouveau en éloignant du message et effectivement empêcher de garantir la livraison, comme le décrit la figure 2.3(b).

En plus, le forwarding a l'inconvénient supplémentaire qui exige la communication avec l'agent mère chaque fois que l'unité mobile se déplace. Dans certaines situations de la mobilité logique, cela fait échouer le but d'utiliser les agents mobiles en réintroduisant la centralisation. Dans l'environnement de la mobilité physique, le mouvement rapide nécessite d'éviter un message envoyé semble improbable.

Cependant, une des tendances dans la mobilité est de réduire la taille de la cellule (exp. : nanocellules) et d'augmenter la réutilisation fréquente. Tant que les tailles des cellules sont réduites, le temps nécessaire pour traverser une cellule se diminué en conséquence.

2.5 Conclusion

Dans ce chapitre, nous avons présenter une nouvelle stratégie de voir la mobilité proposée par Amy Murphy et qui traite une unité mobile comme étant un message nomade qui préserve son identité lorsqu'il traverse le réseau.

Dans ce nouveau modèle, comme tout autre modèle, le problème de la livraison de messages aux unités mobiles paraît important. En effet, les schémas typiques de livraison de messages tels que : le *broadcasting* et *forwarding*, souffrent du problème fondamental qu'une unité mobile se trouve en transite durant la livraison et ainsi elle peut être facilement louper.

Pour cela, nous allons présenter dans le chapitre suivant un algorithme proposé par Amy Murphy pour la livraison de messages en utilisant cette nouvelle notion de la mobilité et qui considère les unités mobiles comme des messages spéciaux. Il est à noter que nous allons restreindre notre intérêt qu'à la mobilité physique. Néanmoins, la mobilité logique est aussi un autre axe de recherche qui est très important.

Chapitre 3

La livraison de messages aux unités mobiles

3.1 Introduction

Avec l'évolution rapide des réseaux et de la technologie sans fil, la mobilité est devenue un axe important de recherche. Un des problèmes majeurs dans la mobilité est la livraison d'un message à une unité mobile tout en se déplaçant.

La plupart des applications, aujourd'hui, se basent sur l'envoi des messages à des emplacements particuliers. Par exemple, l'e-mail suppose que l'émetteur connaît une adresse particulière du récepteur. Dans un réseau fixe, les protocoles de routages tel que dans **[45]**, travaillent sur une topologie fixe. Lorsqu'un message arrive, le routeur exécute une table *Lookup* pour décider quel est le port ou la direction qui va emprunter le message. Malgré que tous les protocoles de routage qui permettent le changement de la topologie, soit dû aux pannes de liens de communication ou dû aux changements de politique, le temps nécessaire pour propager de tels changements n'est pas adéquat pour supporter la mobilité.

Un des modèles de la mobilité couramment implémentés se base sur une architecture d'un réseau fixe, fournissant effectivement des points spécifiques à la connexion des mobiles **[46]**.

Dans le Mobile IP **[44]**, chaque mobile est associé avec un agent mère (home agent) qui fournit un service d'envoi (forwarding) au mobile. Lorsque les unités mobiles se

déplacent vers une nouvelle localisation, elle contacte son agent mère avec les nouvelles informations. Les messages adressés au mobile sont envoyés à l'adresse fixe de l'agent mère.

Dans les téléphones cellulaires, un système similaire au Mobile IP est utilisé lorsque les utilisateurs restent or leurs régions mères (home region). Lorsque le téléphone est activé, l'utilisateur va s'enregistrer au niveau de sa maison, en indiquant essentiellement un nouveau code région pour lui rediriger les appels.

Les solutions du téléphone cellulaire et le Mobile IP travaillent bien avec des systèmes où les unités mobiles ne se déplacent pas fréquemment. Cependant, si le mobile se déplace rapidement entre les différentes cellules, l'information au niveau de l'agent mère va refléter une ancienne position et les messages envoyés à cette localisation vont être perdus. Donc, la question qui se pose : *Peut-on concevoir des protocoles efficaces qui peuvent garantir la livraison à une unité mobile (ou un ensemble d'unités mobiles) et qui peut se déplacer à des vitesses arbitraires à travers le réseau ?*. **[46]**

Clairement, les solutions triviales existent et qui diffusent le message à tous les nœuds jusqu'à l'arrivé du mobile. A l'inverse, les solutions les plus efficaces devraient limiter la liste de diffusion et / ou les conditions de stockage **[46]**.

Dans ce chapitre, on décrit l'application de la nouvelle stratégie de conception, qui considère les unités mobiles comme des messages, pour résoudre le problème de livraison de messages. Tout d'abord, nous parlerons du problème de livraison de messages aux unités mobiles en se basant sur le modèle cellulaire. Par la suite, nous présenterons, en bref, la livraison snapshot du moment où l'algorithme proposé par Amy Murphy pour résoudre le problème de livraison de messages, qu'on va présenter, est une adaptation de l'algorithme classique de Chandy-Lamport. Enfin, nous allons donner quelques limitations de ces algorithmes.

3.2 Motivation du problème

On s'intéresse ici au problème de la livraison des messages à une unité mobile dans un réseau de centres supports mobiles (MSCs) et de stations de base radio (RBSs). Pour simplifier, A. Murphy et al ont supposé que chaque MSC contrôle une seule RBS et tous les MSCs à qui les RBSs sont voisines ont un canal de communication fixe entre eux. Par exemple, dans la figure 3.1, chaque cellule représente un couple unique (MSC, RBS) et les MSCs de toutes les cellules voisines sont connectés par un réseau fixe. Une unité mobile peut envoyer et recevoir des messages seulement d'une seule station de base à la fois et seulement lorsqu'elle est dans la cellule associée à cette RBS.

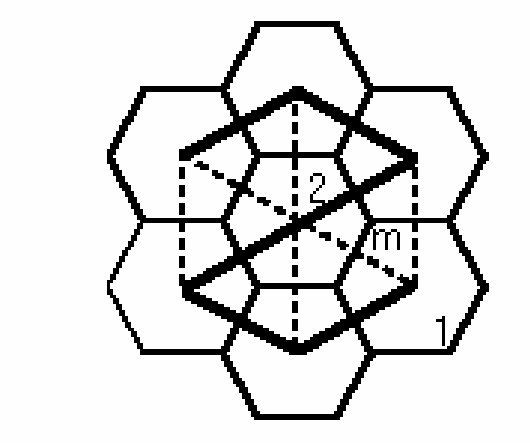


Figure 3.1 – *Diffusion basée cellule.*

Un schéma de diffusion connu dans ce cadre est la construction d'un arbre de recouvrement (spanning tree) sur les MSCs et envoyer les messages sur cet arbre **[48, 49]**. Dans la figure 3.1, un arbre de diffusion est indiqué par les lignes solides (en gras).

Pour cet exemple, on suppose qu'un nœud mobile *(m)* est localisé à la cellule 1 à la frontière de la cellule 2. Supposons maintenant que la cellule 2 prend l'initiative de délivrer un message à l'unité mobile. En suivant l'arbre de recouvrement proposé, MSC2 qui contrôle RBS2 diffuse le message localement, en suite envoi le message sur les deux liens sortant de l'arbre. Après y avoir envoyer le message avec succès, MSC2 supprime sa copie du message. Les MSCs du niveau plus bas se comportent de la même manière, diffusant localement le message ensuite l'envoyé à leurs fils et finalement, suppriment leurs copies du message. Si l'unité mobile ne quitte pas la cellule 1, elle va recevoir une copie du message lorsqu'il est diffusé par RBS1.

Cependant, le fait que l'unité mobile soit sur la frontière qui sépare la cellule 1 de la cellule 2, il est possible pour un handover d'être initié et l'unité mobile de perdre le contact avec RBS1et d'établir une communication avec RBS2. Si ce handover a lieu après que RBS2 n'a supprimé sa copie du message et avant que RBS1 ne diffuse sa copie de ce même message, l'unité mobile ne recevra jamais le message même si elle a été connectée au réseau pendant la durée entière de la diffusion du message.

Bien que, en réalité, les messages traversent le réseau plus vite qu'une unité mobile ne peut traverser une région et le fait qu'un handover de base demande très peu de temps pour se terminer et la longueur du chemin sur l'arbre que *m* peut emprunter entre les deux cellules peut être plus longue, il est raisonnable pour un simple mécanisme de diffusion comme celui ci d'être défaillant dans ce genre de circonstances.

Par conséquent, Amy Murphy et al ont proposé une alternative d'algorithme de diffusion qui garantit que l'agent mobile, dans de telles circonstances, va recevoir une copie du message diffusé. Leur solution est basée sur la notion classique d'un snapshot dans le calcul distribué, en particulier, cet algorithme est basé sur l'algorithme de snapshot original de Chandy et Lamport **[47]**.

3.3 La livraison snapshot

Dans cette section, nous présentons les détails d'application de l'algorithme de snapshot pour le problème de livraison de messages. Dans le but d'éviter la confusion dans la terminologie entre le trafic de contrôle généré par les algorithmes de snapshot et le trafic de données contenant les informations étant transmissent à l'unité mobile, et le fait que les messages puissent jouer un double rôle dans ce nouveau modèle (une unité mobile est un message), nous allons utiliser :

- ***Réseau (network) :*** Pour indiquer une organisation statique de processus nœuds et de canaux.
- ***Unité mobile (mobile unit) :*** Pour indiquer les messages qui modélisent les unités mobiles.
- ***Message de données (***announcement***):*** Il est utiliser pour tout message transportant une information d'une unité mobile vers une autre.

Les messages de données comme ils sont définis sont crées avec une unité mobile et ils sont passés entre les nœuds dans le but d'accomplir une livraison éventuelle à d'autres unités mobiles. En plus, tous les autres genres de messages vont être simplement référencés par *messages*. Cependant, une unité mobile est un message spécial qui conserve son identité lorsqu'elle se déplace à travers le réseau.

3.3.1 Le passage des algorithmes de snapshot distribués vers la livraison de messages de données

Pour garantir la livraison dans toutes les circonstances, une alternative d'algorithme de diffusion basée sur la notion classique de snapshots distribués peut être utiliser. Avant d'adresser la livraison d'un message de données, on note en premier lieu les propriétés générales des algorithmes de snapshot et celles qui sont importantes dans la livraison des messages. *Le but d'un algorithme de snapshot est de fournir une vue consistante sur l'état des nœuds et des canaux d'un réseau.* L'état consiste-en : l'ensemble des processus et de n'importe quel message en transite entre les nœuds.

Un simple algorithme de snapshot bloquera le calcul jusqu'à ce que tous les messages soient or les canaux, enregistrer l'état des processus (en prenant en compte les files contenant les messages sortants), en suite il recommence le calcul. Bien qu'elle soit une solution non- pratique dans la plupart des domaines distribués, elle fournisse une intuition derrière l'algorithme de snapshot, en particulier que l'état global consistant est l'union des snapshots locaux des différents processus. En général, un snapshot est initié par un processus unique et des messages de contrôle sont envoyés aux nœuds voisins en les informant qu'un snapshot est en progression et ainsi les snapshots locaux débutent.

La propriété principale des snapshots que Amy Murphy et al ont exploité est que chaque message va apparaître exactement une fois dans l'état enregistré.

Bien que les algorithmes de snapshot soient développés pour détecter les propriétés stables telles que : la terminaison ou l'inter- blocage, et ce en créant et analysant une vue consistante de l'état distribué, les mises aux points mineurs qu'on décrit dans ce chapitre les adaptent pour représenter la livraison d'un message de données dans l'environnement mobile.

Pour passer d'un réseau de nœuds et de canaux vers l'environnement du calcul mobile, on revient à la structure cellulaire des centres supports mobiles (MSCs). Ces

composants et les liens qui les relient tracent directement le graphe interconnecté du calcul distribué standard. *Les unités mobiles sont simplement représentées comme des messages persistants dans l'environnement distribué,* c.-à-d. elles sont toujours quelque part dans le système soit à un nœud (lorsqu'une unité mobile est en communication avec sa station de base) ou sur un canal (durant un handover) **[43]**.

A ce point, nous avons une structure sur laquelle le snapshot s'exécute et on note que le fait qu'une unité mobile soit un message et le snapshot enregistre l'emplacement des messages, le snapshot global du système mobile va indiquer l'emplacement de l'unité mobile.

Cependant, une option est de simplement délivrer le message directement à cet emplacement ; ainsi, il est possible (dans les systèmes où les unités mobiles changent leurs positions de manière fréquente et rapide) qu'après l'enregistrement de la position de l'unité mobile, elle se déplace au moment où le message de données arrive à la position enregistrée.

Par conséquent, la modification du snapshot enregistré est nécessaire afin de délivrer le message et ce en augmentant les messages de contrôle et l'enregistrement des messages de données s se fait après leurs livraisons. En plus, l'état global du système n'a aucune importance majeure sur la livraison, ainsi, aucune information sur l'état du système n'est demandée d'être rassembler.

3.3.2 L'adaptation de l'algorithme de Chandy-Lamport à la livraison de messages

Partout dans cette section, l'algorithme de snapshot utilisé est celui de Chandy-Lamport **[47]**, et nous allons monter son adaptation pour la livraison d'un message de données. Pour faire la transition à l'environnement mobile, certaines restrictions sont portées sur l'algorithme distribué original et quelques caractéristiques du modèle mobile doivent être claires et ce en passant de la structure cellulaire vers le modèle graphique.

Premièrement, l'algorithme de Chandy-Lamport repose sur des canaux FIFO unidirectionnels. Pour modéliser des canaux bidirectionnels, deux canaux unidirectionnels sont placés dans des directions opposées ; cependant, pour manipuler les suppositions FIFO, des extensions sont nécessaires concernant le protocole de

handover. Dans le modèle cellulaire, une unité mobile se déplace directement entre les cellules ; cependant, dans la représentation de graphe, le mobile doit se déplacer sur un canal avant d'arriver à la nouvelle cellule. C'est une supposition naturelle quand les détails du handover sont considérés.

Dans l'algorithme de Chandy-lamport, il est possible pour le snapshot d'être initialiser au niveau de plus qu'un emplacement dans le réseau ; cependant, initialement, le message est supposé localisé à un certain point dans le réseau ; ainsi, le snapshot va être à l'origine d'un unique MSC.

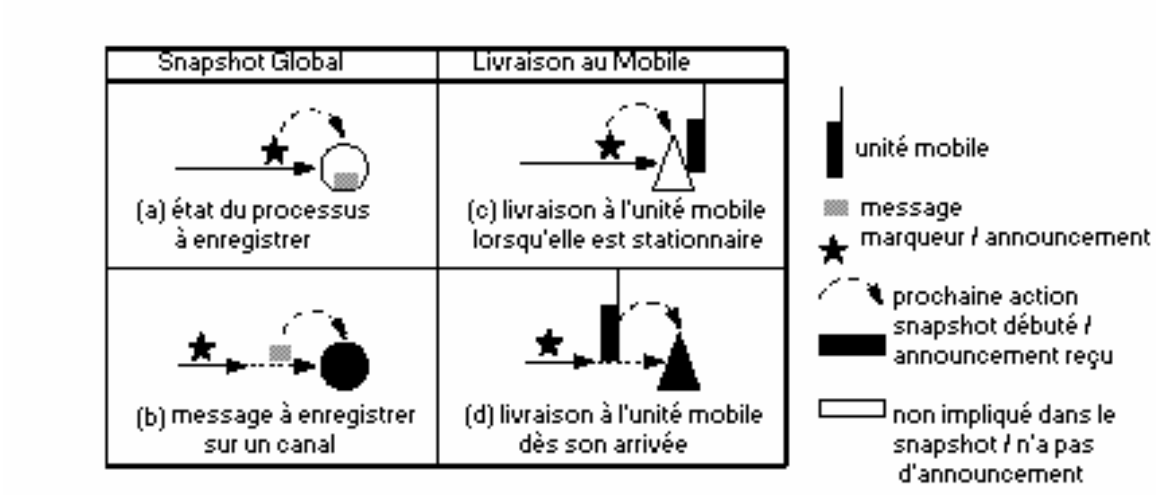


Figure 3.2 – *Translation de concepts de snapshots globaux vers la livraison de messages aux mobiles. Les flèches courbées montrent le traitement d'un élément d'un* canal tandis *que le texte décrit l'action provoquée par un tel mouvement.*

L'algorithme de Chandy-lamport consiste en deux actions principales localisées pour collecter le snapshot local : Le traitement des messages de contrôle (marqueurs) et l'arrivée des messages qui doivent être enregistrés.

La règle d'arrivée d'un marqueur indique que lorsqu'un marqueur arrive à un nœud non- impliqué dans un snapshot, le nœud commence son sanpshot local en enregistrant l'état du processus, ensuite envoyer le marqueur sur tous les canaux sortants (figure 3.2(a)). Dans l'environnement mobile, ceci est analogue à l'arrivée d'un message de données à un nœud. Si l'unité mobile est présente, elle va recevoir sa copie du message, sinon le nœud va rester dans un état de snapshot local et stocker la copie d'un message de données jusqu'à ce que le snapshot local soit terminé. Le snapshot

local est complété lorsque le marqueur / le message de données (noté aussi announcement) arrive de tous les canaux entrants à ce nœud.

La règle d'arrivée d'un message indique que : si le message arrive à un nœud d'un canal *C* avant que le marqueur n'arrive sur ce canal et le nœud est au milieu du sanpshot local, le message est enregistré d'être sur le canal durant le snapshot (figure 3.2(b)). Dans le cadre mobile, cette condition est l'arrivée d'une unité mobile à un MSC qui a stocké une copie du message de données. Cependant, l'arrivée du mobile provoque la transmission du message à ce mobile.

3.4 L'algorithme de livraison de messages basé snapshot

3.4.1 Présentation de l'algorithme

Nous capturons l'essentiel de la solution de Amy Murphy et al sous forme d'un pseudo code montré dans la figure 3.3.

Dans ce programme, l'intérêt vise seulement une unité mobile particulière et un seul message supposé être destiné à cette unité, ce sont les seuls types de messages qui peuvent apparaître n'importe ou dans le système. En plus de l'arrivé du message de données et l'arrivé d'une unité mobile, d'autres cas sont incluent pour la terminaison du snapshot local (*cleaningup*) et afin de permettre à l'unité mobile de se déplacer d'un nœud vers un canal. Les canaux sont supposés être FIFO et supportent à la fois et les unités mobiles et tous les messages.

Le système est supposé être initialiser avec une localisation d'une unité mobile (*MobileAt, MSC A* dans la figure 3.4), et une unique copie du message de données placée à un certain nœud (*AnnAt*). Les canaux sont supposés être vides, et afin d'identifier la terminaison du snapshot local, une variable d'état est introduite et qui évalue sur les canaux (*flushed*). Principalement, un canal dit *flushed* s'il a reçu un marqueur, et lorsque tous les canaux entrants ont reçu le marqueur, le snapshot local est complété.

Déclarations

flushed$_{A,B}$ booleen, il est à vrai si l'announcement traverse le lien de A à B ; initialement est à faux.

AnnouncementAt$_A$ booleen, il est à vrai si l'announcement est stocké à A ; initialement est à vrai seulement où ann débute.

MobileAt$_A$ booleen, il est à vrai si l'unité mobile se trouve à A ; initialement est à vrai seulement où le mobile est localisé.

Actions

ANNOUNCEMENTARRIVES$_A$ (B) ; l'arrivée à A de B

Faire :
flushed$_{B,A}$:= TRUE
si ¬ AnnouncementAt$_A$
– Envoyer l'announcement sur tous les canaux sortants
– AnnouncementAt$_A$:= TRUE
si MobileAt$_A$
– Délivrer l'announcement
Fait ;

MOBILEARRIVES$_A$ (B) ; l'arrivée à A de B

Faire :
MobileAt$_A$:= TRUE
si ¬ flushed$_{B,A}$ **et** AnnouncementAt$_A$
– Délivrer l'announcement
Fait ;

MOBILELEAVES$_A$ (B) ; l'émigration de A à B

Préconditions :
MobileAt$_A$ **et** canal (A, B) existe
Faire :
MobileAt$_A$:= FALSE
L'unité mobile se déplace sur (A, B)
Fait ;

CLEANUP$_A$; A termine son snapshot local

Préconditions :
Pour tous les voisins X, flushed$_{X, A}$ = TRUE
Faire :
AnnouncementAt$_A$:= FALSE ; supprimer ann
Pour tous les voisins X, flushed$_{X, A}$ = FALSE
Fait ;

Figure 3.3 – *Algorithme de livraison basé snapshot*.

Ces actions décrites les transitions d'états d'un nœud local et qui sont suffisantes pour la livraison des messages. Aucune information globale ne doit être maintenue. Un nœud pourra être dans un des trois états suivants :

- *Unnotified :* il n'est pas encore informé du début de snapshot,
- *Notified :* il est entrain de prendre le sanpshot local,
- *Finished :* il a fini le snapshot local.

Dans la figure 3.4, ces états sont représentés par ***blanc***, ***gris*** et ***noir*** respectivement. Tous les nœuds (à l'exception du nœud qui est à l'origine de l'announement) débutent *unnotified*. Un nœud *unnotified* tel que *E* va sûrement recevoir un message de données sur un de ses canaux entrants (*AnnArrives*) (tel que *(B, E)*). Cette action lui cause une transition vers l'état *notified*, en délivrant le message de données si c'est possible, il garde une copie du message, il marque le canal sur lequel le message de données s'est arrivé comme *flushed* et il envoi des copies du message de données sur tous les canaux sortants.

Une fois le canal est marqué *flushed*, si l'unité mobile arrive sur ce nœud alors il est sûr d'avoir reçu le message de données par un certain autre nœud (il a été enregistré dans un autre snapshot local). Cependant, pour éviter la livraison multiple, si le mobile arrive sur un canal *flushed*, la livraison ne se répète pas (*MobileArrives*). Si le message de données arrive à un nœud *notified* tel que *A*, le canal sur lequel il est arrivé va être marqué comme *flushed*, mais le fait que le message de données soit toujours sauvegardé, aucune copie supplémentaire n'est faite. Lorsque tous les canaux entrants ont été marqués *flushed* (comme dans *B*), le snapshot local du nœud est terminé et l'état local (en inclut le statut *flushed* des canaux et les messages de données sauvegardés) peut être supprimé (*Cleanup*).

L'action finale, *MobileArrives*, modélise les mouvements d'un mobile loin d'un nœud. Le mobile est simplement placé sur le canal et les variables d'états sont modifiées pour refléter ce changement. Cela accomplit effectivement les mouvements au hasard d'une unité mobile, cependant, si un modèle de mouvement particulier est désiré, il peut être ajouté à cette action.

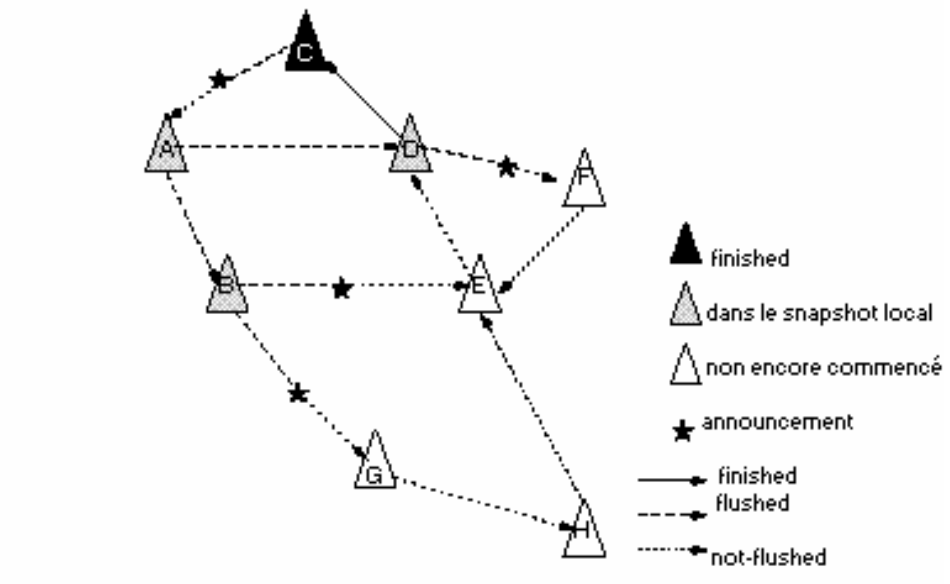


Figure 3.4 – *Les phases de livraison.*

3.4.2 Les propriétés

Nous avons présenté dans la section précédente un algorithme de snapshot global modifié pour représenter la livraison de messages dans un système mobile. Le fait que l'approche soit basée sur un algorithme bien connu pour le calcul distribué, on peut adapter les propriétés déjà prouvées dans l'environnement de calcul distribué à l'environnement mobile.

Les trois propriétés primaires prouvées par le snapshot distribué de Chandy-Lamport sont :

(a) Il n'y a aucune mémoire résiduelle dans le système à un certain point après que l'algorithme ne débute son exécution.
(b) Chaque message est enregistré une fois.
(c) Aucun message n'est enregistré plus d'une fois.

La transformation directe de ces propriétés en environnement mobile donne **[43]** :

(a) Eventuellement, il n'y a aucune mémoire résiduelle dans le système à un certain point après que le processus de livraison ne débute.
(b) Le message de données est délivré au récepteur destiné.

(c) Le message de données est délivré seulement une fois.

Eventuellement, pour montrer que toutes les informations concernant le message de données sont supprimées du système, il faut montrer que le programme atteint un état où : il n'y a aucun message de données à n'importe quel nœud, il n'y a aucun annoncement dans n'importe quel canal et tous les canaux sont vides (*unflushed*). Cela peut être vu en observant que par la connectivité du graphe et les règles de dispersion de l'algorithme, chaque nœud reçoit une copie du message sur chacun de ses canaux. Dans la figure3.4, cela est analogue à chaque nœud devient *notified*. Une fois cela est arrivé, chacun des canaux est *flushed* et l'action de nettoyage (*cleaunup*) est permise. Dans la figure, le nœud va transiter vers l'état *finished*. Donc, il n'y aura aucun message de données sur les canaux car une fois qu'un message est passé à travers un canal, il n'est pas possible pour un autre message de données d'être envoyer sur ce canal. Par conséquent, la première propriété est prouver.

En suite, il faut montre que le message de données est éventuellement délivré. Premièrement, nous notons que tous les nœuds reçoivent le message de données. Lorsque cela arrive, si l'unité mobile lui a été délivrée le message alors la preuve est terminée. Sinon, si l'unité mobile n'a pas été retrouvée alors c'est le cas où l'unité mobile est localisée sur un canal *unflushed* et elle doit se trouver avant le message de données sur le canal. Dans la figure 3.4, une unité mobile *unnotified* va être localiser sur un nœud blanc, ou sur un canal en pointillé, mais dans ce cas, nous avons décrit un graphe dans lequel tous les nœuds sont déjà *notified* ; cependant, l'unité mobile doit être localiser sur un segment de canal *unflushed* en avant d'un message de données. Par conséquent, le fait que l'unité mobile arrive sur un canal unflushed, et le fait que tous les nœuds aient reçu une copie du message, le nœud vers lequel elle se dirige doit avoir une copie du message de données. La livraison sera faite lorsque l'unité mobile arrive sur le nœud.

Ayant montré que le message va être sûrement délivrer, il reste à montrer que le message de données est délivré seulement une fois. Pour le faire, il est suffisant de montrer qu'après la livraison, une unité mobile ne peut pas être dans une position de lui délivrer un message de données. La livraison peut se faire dans deux situations :

1) *L'unité mobile étant sur un nœud unnotified lorsque le message de données arrive.*

2) *L'unité mobile qui arrive (sur un canal unnotified) à un nœud notified.*

Pour montrer que chacun de ces cas ne vont pas se présenter après la livraison, Amy Murphy caractérise une région du graphe appelée ***will-be-notified***. Cette région est définie comme étant un ensemble de tous les nœuds *unnotified* et tous les canaux ou les segments de canaux sur lesquels une copie du message de données n'a pas été passé à travers. Dans la figure 3.4, cette région correspond aux nœuds blancs (*unnotified*) et les canaux en pointillé (*unflushed*). Ayant définit cette région et noter que les deux cas de livraison cités précédemment arrivent lorsque l'unité mobile est dans cette région, il est suffisant de montrer qu'après la livraison, l'unité mobile ne va pas être dans la région ***will-be-notified***. Par conséquent, afin que l'unité mobile se déplace or cette région, elle doit dépasser le message de données et le fait que les canaux soient FIFO et en plus les règles de dispersion du message de données, ceci n'est pas possible. Donc, le message de données peut seulement être délivré une fois et une seule.

Evidemment, ces trois propriétés sont prouvées par l'algorithme de Amy Murphy. Pour ceux qui veulent voir la preuve complète et détaillé de ces propriétés, ils peuvent se référencer aux articles suivants : **[48, 49]**.

3.4.3 L'application des hypothèses FIFO

Lorsqu'on passe de l'environnement distribué à l'environnement mobile, diverses suppositions doivent être faites sur la nature du réseau et le comportement des composants dans le réseau entre autres les canaux FIFO.

Un inconvénient majeur d'utiliser l'algorithme de Chandy-Lamport est sa dépendance au comportement FIFO des canaux. Plus précisément, le fait que les unités mobiles et les messages soient modélisés à traverser le même canal, cela semble être une supposition incorrecte car les unités mobiles émigrent beaucoup plus lentement à travers un espace que les messages transitent à travers un réseau fixe.

Pour exploiter plus ce problème, les mouvements des messages et les unités mobiles doivent être vues en détails, spécialement, il faut revoir le handover des unités mobiles entre les cellules. Nous allons montrer comment l'hypothèse FIFO peut être violer et présenter, par la suite, un simple mécanisme proposé par A. Murphy **[43]** afin de le restaurer.

Un des U. S. Standards pour la communication cellulaire analogique est le AMPS **[51]**, dans lequel les téléphones cellulaires sont réglés sur une seule fréquence à la fois. Lorsque le signal entre le MSC et une unité mobile commence à se dégrader, le MSC cherche un MSC voisin avec un potentiel de signal de communication plus fort indiquant ainsi l'unité mobile d'émigrer vers cette cellule particulière et une fréquence disponible dans cette cellule. Lorsque la fréquence est demandée, un handover se déclenche.

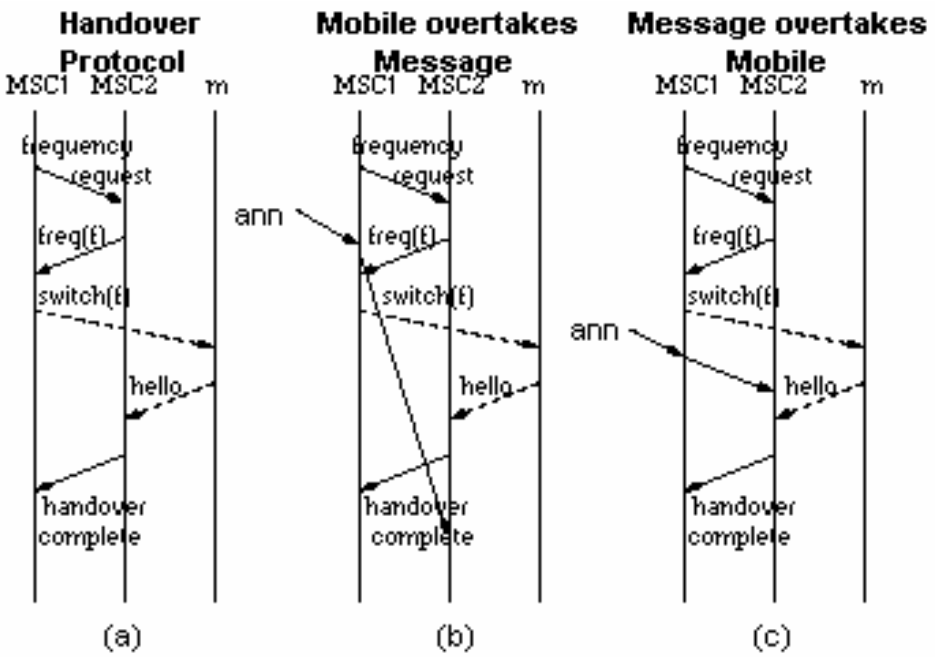


Figure 3.5 – *AMPS handover protocol.* ***(a)*** *Si les messages sont traités (c-à-d., diffusés au mobile) immédiatement dès leur réception, il est possible* ***(b)*** *Pour le mobile de se déplacer plus rapidement que le message sur le canal, ou* ***(c)*** *Pour le message de transiter plus vite que le mobile, ainsi vider la propriété FIFO des canaux.*

La figure 3.5 (a), montre les messages de contrôle échangés lorsqu'une unité mobile, *m*, émigre de la cellule *A* vers la cellule *B*. Premièrement, la requête de fréquence (*frequency request*) est échangée entre MSCs. A ce moment, l'unité mobile est informée du handover par la réception d'une nouvelle fréquence de son MSC courant, *A*. Après, le transfert de la nouvelle fréquence, le mobile envoi un *hello* sur la nouvelle fréquence, alertant *B* que le mobile est maintenant à l'écoute sur la nouvelle

fréquence. Finalement, *B* envoi un *handover complete* à *A*, pour qu'il libère l'ancienne fréquence.

Le premier souci maintenant, est de faire les canaux FIFO tout en respectant les unités et les messages (et les messages de contrôle et les messages de données). Même si on suppose que les canaux entre les MSCs sont FIFO, une réorganisation est possible car une partie du handover prend place sur les canaux sans-fil qui ne sont pas synchronisés avec les canaux filaires **[48, 49, 50, 43, 46]**. Spécialement, les deux cas du comportement non-FIFO sont adressés :

1) *Le mobile dépasse le message, et*

2) *Le message dépasse le mobile.*

Il est important de définir à quel point le mobile se déplace logiquement sur le canal. Ceci peut être défini comme étant le moment où la communication avec *A* est terminée ou lorsque le message *switch* est transmis. De même, le mobile sort du canal lorsque la transmission sans-fil du message *hello* est acceptée par le destinataire. Comme on peut le voir dans la figure 3.5(b), il est possible pour un message envoyé sur le canal filaire avant le message *switch*, d'arriver à la destination après l'arrivée de l'unité mobile, violant ainsi l'ordre FIFO. De manière similaire, un message envoyé après le message *switch* peut transiter rapidement à travers le canal et arriver à la destination avant le mobile (figure 3.5 (c)).

Afin de surmonter ce problème, A. Murphy et al **[48, 49, 50, 43, 46]** propose un changement mineur dans le protocole pour impliquer et les canaux filaires et les canaux sans-fil dans le handover. Le seul changement du côté de la source (*A* dans ce cas) est la transmission filaire d'un message de manière atomique avec la transmission sans-fil du *switch*. Ce message est appelé *unité mobile virtuelle (vmu)* car il identifie le point sur le canal filaire au moment où le mobile quitte la source. Tous les messages envoyés avant le *vmu* sont envoyés avant que l'unité mobile ne quitte la source et tous les messages envoyés après le *vmu* sont envoyés après que l'unité mobile ne quitte la source.

Cependant, le comportement du destinataire va changer (*B* dans ce cas) pour avoir ce nouveau comportement. Le but est d'avoir l'unité mobile virtuelle et l'unité mobile physique arrivent à la destination en même temps. Cependant, si le *hello* arrive avant le *vmu*, tous les messages arrivant sur le canal filaire sont traités comme si l'unité mobile

n'était pas présente même si la communication est possible. Inversement, si le *vmu* arrive avant le *hello*, tous les messages envoyés sur le canal filaire sont enregistrés jusqu'à l'arrivée du *hello*. Lorsque les deux messages sont arrivé et ont été traités, B continue le traitement de tous les messages dans l'ordre de leurs arrivés. En forçant le récepteur d'attendre les deux messages, les canaux filaires et non-filaires sont synchronisés, produisant effectivement un unique canal FIFO pour à la fois les unités mobiles et les messages.

3.4.4 Contrôle d'exactitude

Afin de passer au domaine mobile à partir du modèle cellulaire, plusieurs suppositions ont été faites.

Premièrement, A. Murphy et al (track) a proposé que les canaux soient FIFO et que et les messages et les unités mobiles traversent ces canaux. Comme nous l'avons vu dans la section précédente, il est possible d'étendre le protocole du handover pour appliquer les canaux FIFO.

Deuxièmement, tous les MSCs voisins sont directement connectés par un canal filaire. Cependant, pour des raisons de coût et de faisabilité, ce n'est pas souvent le cas. Par conséquent, il suffit d'ajouter des canaux virtuels entre chaque couple MSCs et probablement imposer les mêmes contraintes FIFO sur ces canaux.

Troisièmement, la livraison snapshot suppose la fiabilité des liens de livraison. Cependant, la plupart des Internet utilisent des liens non-fiables comme *Ethernets*, *frame relay* et *ATM* **[46]**. Malgré que la probabilité d'erreur soit petite, les paquets sont vraiment perdus. Une solution possible est d'ajouter des ACKs aux messages du message de données s (comme il est utilisé, par exemple, dans l'algorithme d'inondation intelligent (*intelligent flooding algorithm*) utilisé par Link State Routing dans OSI **[45]** et OSPF **[52]**). Une autre solution est seulement de fournir un meilleur service **[46]**. Le fait que la perte d'annoncements puisse produire un inter-blocage, la suppression du message de données est nécessaire après un time-out même s'il est supposé être toujours sur un canal.

Finalement, l'hypothèse que tous les MSCs dans le réseau sont impliqués dans chaque livraison paraît impossible car il n'est pas pratique de supposer que tous les

nœuds dans Internet vont participer dans un snapshot pour localiser une seule unité mobile.

3.5 Conclusion

Dans ce chapitre, nous avons vu comment peut-on transférer des résultats faites et prouvés dans le calcul distribué vers l'environnement mobile.

En effet, nous avons présenté un algorithme de A. Murphy qui est adresser à la livraison fiable des messages aux unités mobiles en appliquant la nouvelle stratégie décrite dans le chapitre 2 et en utilisant un mécanisme du calcul distribué qui est le snapshot. Cet algorithme permet de localisé une unité mobile et lui délivré le message durant le snapshot. Il est à noter que cet algorithme est proposé pour la livraison *unicast* des messages du moment où il spécifie un seul message adressé à une seule unité mobile.

Finalement, nous avons vu que le fait que les unités mobiles sont traitées comme des messages, cela fourni une manière efficace de transférer les résultats prouvés par des algorithmes distribués classiques pour émerger un champ du calcul mobile.

Chapitre 4

La livraison multicast de messages aux unités mobiles

4.1 Introduction

La mobilité des unités mobiles introduis un nouveau ensemble de problèmes non constaté dans les réseaux comprenant seulement des unités statiques.

Premièrement, le fait que la localisation d'une unité mobile change à chaque mouvement, sa localisation courante doit être déterminée en premier avant qu'un message ne peut être routé à cette unité mobile.

Deuxièmement, la bande passante du lien sans fil connectant une unité mobile à une station de base est significativement moins que les liens filaires entre les unités statiques **[53,12]**.

Troisièmement, les unités mobiles ont des contraintes limités sur la consommation d'énergie relative aux stations fixes **[53,54,55]**.

Par conséquent, les algorithmes réseaux pour faciliter la mobilité des unités dans un réseau statique ont besoin d'être spécialement adapté pour saisir ces contraintes.

Dans ce chapitre, nous allons présenter des protocoles **[30]** qui traitent le problème du *multicast* en environnement mobile. Ces protocoles représentent, en effet, un schéma de base pour la conception de plusieurs protocoles qui sont venus par la suite.

Ce chapitre est structuré comme suit : tout d'abord, nous allons définir le problème du multicast en présentant les effets de la **mobilité** des unités mobiles sur la livraison **fiable** des messages **multicast**. En suite, nous allons présenter les algorithmes proposés par A. Acharya et B. Badrinath dans **[30]** suivis d'une discussion. Finalement, nous présentons un aperçu sur la notion du groupe multicast et nous terminons par une conclusion.

4.2 Le problème du multicasting

Le multicasting nécessite d'envoyer des copies d'un même message à des stations de base différentes (cellules). Dû à la capacité de déplacement des unités mobiles entre les différentes cellules, deux cas peuvent se présenter : **(1)** un message multicast peut être délivrer à un récepteur à des localisations multiples (dans plusieurs cellules), **(2)** un message multicast peut ne pas être délivrer du tout à un récepteur à cause de la mobilité des récepteurs **[56]**.

Il existe un certain nombre de protocoles de routages multicast pour les unités statiques ou fixes, par exemple **[57, 58, 59]** ; mais les effets de la mobilité sur le routage multicast **[68]** ont commencé à recevoir l'attention seulement dernièrement. De même, il existe encore un corps de travail sur les protocoles multicast fiables **[60, 61, 62]** et ce pour les unités statiques. Donc, théoriquement, de tels protocoles peuvent supporter la mobilité des unités si le réseau correspondant est capable de router les paquets multicast au / des récepteurs / émetteurs mobiles.

Cependant, plusieurs protocoles ont été proposés dans le cadre d'une livraison fiable des messages multicast en tenant compte des problèmes posés par la mobilité entre autre les protocoles proposés par Acharya-Badrinath **[30]** et qui sont basés sur une approche de ***two-tier***, où les conditions de communication et de calcul du protocole entier sont déplacées au niveau de l'infrastructure filaire/statique.

4.3 Les effets de la mobilité sur la livraison multicast

Une livraison multicast signifie l'envoi de copies d'un même message à des destinations multiples dans le réseau. Afin d'assurer une livraison fiable des messages,

ces derniers peuvent être sauvegarder au niveau de plusieurs stations de bases. Le fait qu'une unité mobile peut se connecter au réseau statique à partir de différentes MSSs et à des moments différents, elle peut recevoir plus d'une copie du message. De même, elle peut ne pas recevoir de copie du message du tout. Par conséquent, les facteurs suivants contribuent au problème :

- Deux MSSs peuvent recevoir des copies du même message multicast à des moments différents (pas en même temps),
- Les MSSs peuvent scheduler (faire envoyer le message sur les liens de communication sans fil) des copies du message pour une transmission sans fil à des temps différents dans leurs cellules respectives,
- La latence du réseau sans fil à l'intérieur d'une cellule, et
- Une MH peut perdre la connexion réseau lors d'un changement de cellules, spécialement lorsque les cellules ne sont pas chevauchées.

La figure 4.1 représente les effets de la mobilité des hôtes sur la livraison des messages multicast aux unités mobiles : un message multicast peut être délivré à un récepteur à des localisations multiples (cellules) ou peut ne pas être délivré à aucun et ce dépend de la mobilité des récepteurs.

Soit un message multicast *m* envoyé par MSS_2 aux unités attendues h_1, h_2, h_3 et h_4. Une copie de *m* est envoyée à toutes les cellules dont un récepteur est localisé, c.-à-d. MSS_3 , MSS_4 et MSS_5. Chaque MSS réceptrice est maintenant responsable d'assurer la livraison de *m* aux destinataires mobiles localisées dans sa cellule. Concédèrent maintenant les cas suivants :

- h_3 n'a pas reçu *m* de MSS_4 ni de MSS_5 malgré le fait que les deux stations de base ont reçu des copies de *m*.
- h_2 a reçu une copie de *m* de chacune des stations de base MSS_4 et MSS_3.
- h_1 ne reçoit jamais *m* car elle s'est déplacée vers la cellule correspondante à MSS_1 qui ne possède pas une copie de *m*.

- h_4 n'a pas changé de cellule lorsque le message m est en progression et donc elle a reçu m exactement une fois de MSS_5. Contrairement à h_1 et h_2 qui violent la contrainte unique d'une livraison fiable à cause de leur mobilité (ou déplacement).

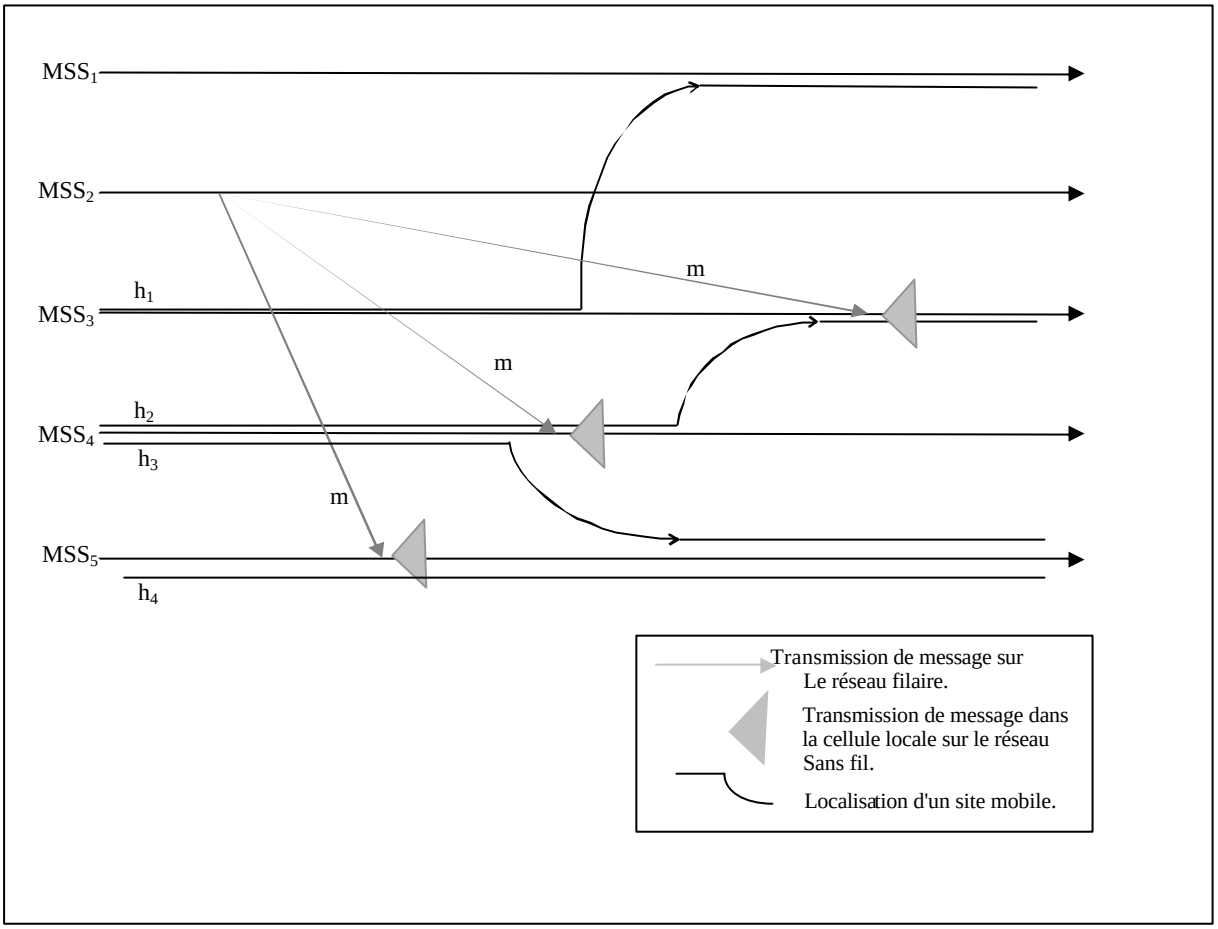


Figure 4.1– *Les effets de la mobilité des hôtes sur la livraison multicast des messages.*

Cet exemple, soulève aussi les deux questions importantes : *Quelles sont les MSSs aux quelles il faut envoyer une copie du message m ? et jusqu'à quand la copie du message doit être sauvegarder au niveau d'une MSS avant qu'elle ne peut être sûrement supprimer ?*.

Par exemple, h_1 se déplace vers la cellule contrôlée par MSS_1 sans avoir reçu m de MSS_3. Afin d'assurer que h_1 reçoit m d'au moins une cellule, il devient nécessaire d'envoyer une copie de m à MSS_1. En suite, considérant le mouvement de h_3. Si h_3 va recevoir m d'au moins un emplacement (dans ce cas par MSS_5), alors la copie au niveau

de MSS_5 ne peut pas être supprimer jusqu'à ce que h_3 reçoive m. Si les MSSs ne gardent pas trace des localisations courantes des unités mobiles, alors pour assurer au moins une livraison, il est nécessaire d'envoyer une copie du message à toutes les MSSs du système. En plus, les stations de base ne peuvent pas supprimer leurs copies locales seulement après que le message sera délivrer à tous les récepteurs voulus.

4.4 Schémas de protocoles multicast

Dans cette section, nous allons présenter des protocoles qui sont explicitement conçus pour satisfaire les deux principales contraintes physiques d'un environnement de calcul mobile : La faible bande passante des connexions sans fil entre une station de base et une unité mobile locale et Les contraintes limités sur la consommation d'énergie des unités mobiles. En plus, le fait que la capacité de communication et de calcul qui est plus grande pour les unités statiques, les protocoles présentés sont structurés de telle façon que les conditions de calcul et de communication d'une exécution d'un tel protocole sont réalisées dans le segment statique du réseau à une extension possible **[30]**.

Cela est accompli en partitionnant les protocoles multicast en deux interactions : une, qui se réalise entièrement dans le réseau statique entre les stations de base et la deuxième, qui prend place seulement dans une seule cellule entre une station de base et une unité mobile locale. Cette approche a été aussi utilisée pour structurer les algorithmes distribués **[63, 3, 2, 29]** et pour augmenter les performances TCP **[64]** pour les unités mobiles **[30].**

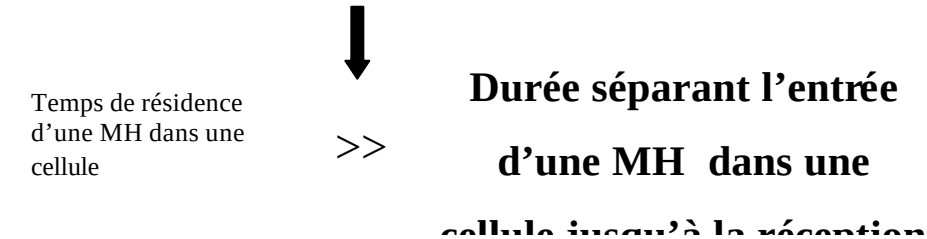
Donc, nous allons présenter des algorithmes pour la livraison multicast des messages à des destinations mobiles pour les cas suivants : d'au moins une localisation, d'au plus une localisation et pour exactement une localisation.

4.4.1 Modèle et hypothèses

Le modèle du système est le modèle cellulaire. Le système est formé de N_s stations de base notées MSSs et de N_h unités mobiles notées MH qui communiquent directement avec les MSSs. Une cellule est contrôlée par une et une seule station de base.

Les protocoles proposés par Acharya-Badrinath sont basés sur les hypothèses suivantes :

- ✓ La livraison des messages entre chaque couple d'MSSs est supposée être *fiable*.
- ✓ Le réseau sans fil dans la cellule locale d'une MSS assure *la livraison FIFO* des messages entre une MSS et une MH locale.
- ✓ Chaque MH lui est associée une adresse unique ***id*** qui est indépendante de sa localisation dans le réseau.
- ✓ Tous les composants du système (et les unités mobiles et les liens de communication) sont supposés être fiables et les MSSs ont suffisamment de mémoire pour bufférisers (enregistrer) les messages multicast.
- ✓ Le modèle du système exige que le segment statique du réseau a une capacité de traitement des hôtes et une vitesse de communication suffisante, de telle façon qu'après qu'une MH entre dans sa nouvelle cellule et avant qu'elle ne quitte, par la suite, la cellule : La procédure *handoff* est complétée et s'il y a des messages multicast pendant cette durée au niveau du MSS locale qui peuvent être délivrés au MH alors l'MSS doit être capable de livrer au moins une fois un tel message.



Il est à noter qu'une exécution d'un protocole peut être demandée soit par une unité mobile ou par une station fixe. Lorsque la demande est invoquée par une MH, l'MSS locale est responsable de l'exécution du protocole au nom de l'unité mobile. De même, une unité fixe non-MSS peut demander aussi une exécution du protocole ; et dans ce cas la requête va être envoyer à une station de base qui va alors exécuter le protocole souhaité. Ainsi, le terme *initiateur* va faire référence au MSS exécutant un protocole au nom d'une MH ou une autre station fixe.

En outre, la livraison fiable d'un message multicast aux unités mobiles demande que chaque récepteur doit répondre par un ACK à la réception du message. Cependant, envoyer chaque ACK séparément à l'initiateur du multicast va mener à un problème *d'implosion d'ACK*. Ceci est traité par Acharya-Badrinath dans ses schémas de protocoles multicast en exigeant que chaque station de base doit collectée les ACKs de tous les récepteurs dans sa cellule avant d'envoyer une liste de ces ACKs à l'initiateur.

Les protocoles que nous allons présenter partagent les trois caractéristiques suivantes :

- Un ensemble arbitraire d'unités mobiles peut être spécifié à être l'ensemble des récepteurs d'un message multicast.
- Chaque MSS dans le système reçoit une copie du message multicast en prévenance de l'initiateur.
- Chaque MSS est considérée à être un initiateur possible d'un multicasting.

4.4.2 Handoff

Un *handoff* est déclenché lorsqu'une MH «découvert » qu'elle est entrée dans une nouvelle cellule. Les schémas utilisent une représentation abstraite de la procédure handoff de la couche réseau présentée dans **[6]**.

Considérant une MH *h* qui se déplace physiquement de la cellule couverte par *M* vers la cellule couverte par *N* (étape (1) de la figure 4.2). En découvrant qu'elle est entrée dans une nouvelle cellule, *h* envoi un message *greeting(h, M)* à *N* en précisant son propre identité *h* et l'identificateur de son ancienne MSS. *N* répond à *h* par un message d'acquittement positif *greetingACK*. Si le message d'accueil (*greeting*) est perdu, l'MH va simplement le ré-envoyer. Si le message d'acquittement (ACK) est perdu, l'MH va envoyer un autre *greeting* jusqu'à l'obtention d'un acquittement positif. Ainsi, l'hypothèse suivante est nécessaire : *lorsqu'une MH entre dans une nouvelle cellule, un message greeting est reçu par succès par l'MSS locale comme il est indiqué dans l'étape (2) de la figure 4.2.*

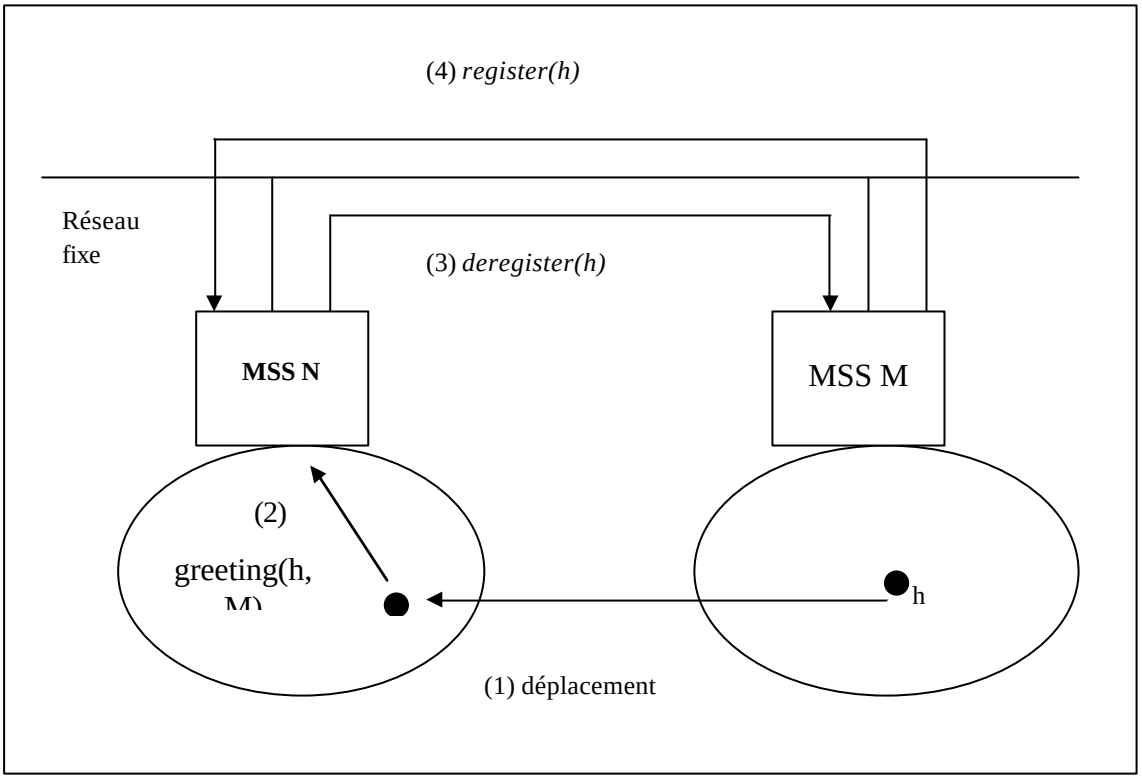


Figure 4.2– *Handoff*.

Les listes, *M- local* et *N- local*, contiennent les identificateurs des MHs locaux des cellules de *M* et *N* respectivement. A la réception du *greeting(h, M), N* envoi un message *deregister(h)* à *M* (étape (3) de la figure 4.2). *M*, alors, supprime *h* de *M- local* et transmis les structures de données liées à *h* à *N* via un message *register(h, <data>)* (étape (4) de la figure 4.2). En réponse, *N* ajoute *h* à sa liste locale *N- local.*

Le fait que les canaux de communications entre une MSS et une MH locale sont supposés être bidirectionnels et FIFO, il est nécessaire d'assurer que ces canaux FIFO sont vides lors de la migration du MH vers la nouvelle cellule. Ceci est assurer de la manière suivante :

Soit une MH *h* qui change de cellule de *M* vers *N*. *h* inclut maintenant un paramètre en plus $seq_{M\text{-}to\text{-}h}$ dans le message *greeting* ; c'est le numéro de séquence du dernier message reçu sur le canal FIFO liant *M* à *h*. En plus, *h* va se débarrasser (supprimer) de tous les messages reçus à partir de *M* et dont le numéro de séquence est supérieur à $seq_{M\text{-}to\text{-}h}$ le fait que leurs livraisons va violer l'ordre FIFO des canaux. Ce numéro est envoyé à *M* comme un paramètre supplémentaire au message *deregister()* de la part de

N. Au niveau de *M*, ceci représente un ACK à la livraison de tous les messages envoyés à *h* et dont les numéros de séquences sont inférieurs ou égaux à $seq_{M\text{-}to\text{-}h}$. De même, *M* inclus $seq_{h\text{-}to\text{-}M}$ dans le message *register()* envoyé à *N*.

4.4.3 La livraison de message au moins une fois

Le schéma de livraison d'un message *m* à un ensemble d'unités mobiles *dest(m)* est comme suit :

1. L'initiateur *M* envoi une copie de *m*, accompagnée de *dest(m),* à tous les MSSs du système.

2. A la réception de *m*, un MSS *N* sauvegarde le message et crée une liste vide *ack-list(m)* pour garder trace de ses MHs locales et pour celles *m* doit être délivrer.

   a) Pour chaque MH *h* locale, qui est incluse dans *dest(m)*, *N* envoi *m* à *h*.

   b) Lorsque *h* répond par un ACK à la réception de *m*, *N* insert *h* dans *ack-list(m)* et supprime h de *dest(m).*

   c) Lorsque *dest(m)* ne contient plus une MH locale, *N* envoi *ack-list(m)* à *M* (seulement si elle n'est pas vide). *N* alors remet *ack-list(m)* à vide.

   d) Si une MH, qui est incluse dans *dest(m),* entre dans la cellule locale, *N* va alors ré-exécuter les étapes a), b) et c) afin de délivrer *m* à MH.

3. Lorsque *M* reçoit un message *ack-list(m),* elle supprime les MHs de *dest(m)* qui sont listées dans *ack-list(m).* Lorsque *dest(m)* devient vide (chaque récepteur désiré a répondu par un ACK à la réception de *m*), *M* envoi un message *delete(m)* à toutes les MSSs.

4. La réception de *delete(m)* par une MSS implique la suppression de *m* et la liste associée *ack-list(m)* du buffer local.

Malgré que la livraison des messages est fiable dans le réseau fixe et localement dans les cellules, l'initiateur a besoin d'attendre un acquittement positif explicite à la réception du message multicast de chaque destination mobile dû à la mobilité des

récepteurs et non pas à la fiabilité des liens de communication. En raison de la mobilité des MHs, une unité mobile destinataire peut recevoir plusieurs copies de *m* de différentes cellules.

4.4.4 La livraison de message au plus une fois

Un schéma pour la livraison d'au plus une fois doit assurer qu'un récepteur ne doit pas recevoir le message plus qu'une fois d'une station de base. Il est comme suit :

1. Chaque MSS maintient un compteur qui est incrémenté si elle a initié une nouvelle exécution du protocole. Le numéro de séquence du message multicast est affecté à ce compteur. Pour envoyer un message multicast *m*, l'initiateur M envoi une copie ainsi que la listes des destinataires *dest(m)* à toutes les MSSs ; soit *m_seq* le numéro de séquence attribué à *m*.

2. Chaque MH *h* lui est associé un tableau *h_RECD*[1..N_s]. Ce tableau est stocké au niveau d'une MSS locale.

3. Pour chaque MH *h*, une MSS vérifie si h appartienne *dest(m) et si h_RECD*[*M*] < *m_seq*. Si ces conditions sont vérifiées alors *m* est transmit sur un lien sans fil à *h*. Lorsque *h* reçoit *m*, elle répond par un ACK à MSS, et dans cas *h_RECD*[*M*] = *m_seq*.

4. Une MSS peut supprimer *m* à n'importe quel moment après que l'étape (3) ne soit terminée.

5. Lorsqu'une MH *h* quitte sa cellule, le tableau *h_RECD*[ ] est inclut dans le message *register()* durant l'exécution du protocole *handoff*.

L'idée clé de ce protocole est de stocker le tableau *h_RECD*[ ]au niveau des MSSs et non pas au niveau de l'MH *h* elle même. En plus, la taille du tableau est proportionnelle à Ns et non pas au nombre des MHs ; ceci est un avantage du moment où nous avons $N_h >> N_s$. Mais, avec ce protocole il y a la possibilité qu'une unité mobile peut ne pas recevoir le message multicast du tout.

4.4.5 La livraison de message exactement une fois

La livraison exactement une fois est accomplie en assurant les deux types de livraison : au plus une fois et au moins une fois. Le protocole de livraison exactement une fois est ainsi basé sur les observations suivantes :

- ✓ Les copies d'un message multicast doivent être disponibles au niveau de chaque MSS.
- ✓ Un acquittement positif explicite de la livraison de message est exigé pour chaque récepteur mobile.
- ✓ L'association d'un tableau *RECD[ ]* à chaque MH assure la livraison du message au plus une fois à l'unité mobile.
- ✓ Due à la faible bande passante des liens de communication sans fil et les contraintes liées à la puissance limité des MHs, le tableau *RECD[ ]* est stocké au niveau d'MSS locale et il est transféré sur le réseau fixe durant le handoff.

Le protocole est constitué de trois modules. Le module fixe (Wired module) est responsable de toutes les communications entre un initiateur et les autres stations de bases. Le module sans fil (Wirless module) contrôle les communications entre une MSS et les MHs locales. Le module handoff transfert les informations d'états utiles entre les MSSs lorsqu'une MH change de cellule.

4.5 Discussion

Le travail proposé par Acharya-Badrinath est un travail qui est différent des autres recherches en trois points clés :

Premièrement, le but de conception pour les travaux référencés **[44 ,65, 6, 66, 67]** est de garder la mobilité des hôtes transparentes à la couche transport et aux couches situant au dessus de cette dernière ; et cependant, saisir la mobilité des hôtes à travers de nouveaux protocoles et mécanismes d'adressages au niveau des couches réseau et communication.

Deuxièmement, ils sont essentiellement concernés par le routage des messages de / à une unité mobile sans prendre en considération sa localisation dans le réseau.

Troisièmement, leur but a été limité aux messages *unicast* (point- à-point).

Concernant la complexité de communication des protocoles proposés par Acharya-Badrinath, le coût (au pire des cas) de la livraison d'un message *m* à un nombre de récepteurs mobiles $N_{dest(m)}$ est :

$$2 \times N_s \times C_f + N_{dest(m)} \times C_{sf} + N_{ack\text{-}list} \times C_f$$

Où: N_s : nombre de station fixe dans le système.

$N_{dest(m)}$: nombre de destinataires mobiles du message m.

$N_{ack\text{-}list}$: nombre de messages ack-list() envoyés par les MSSs à l'initiateur du multicast.

C_f : coût de communication filaire.

C_{sf} : coût de communication sans fil.

4.6 Notion de groupe multicast

Lorsqu'une source multicast envoi une unique copie d'un message de données aux récepteurs désignés, un groupe de ces récepteurs va être former pour la mise en place d'une session multicast.

Un groupe multicast est un ensemble de composants réseau (network devices) identifiés par une adresse multicast commune. D'un côté, les émetteurs n'ont pas besoin d'être des membres du groupe et n'ont pas une connaissance à priori sur les membres de ce groupe.

D'un autre côté, les récepteurs vont rejoindre le groupe ; un récepteur initié sa requête d'appartenance (membership request) lorsqu'il apprend qu'un groupe est formé. Un routour multicast local envoi périodiquement les messages *membership query*.

Chaque hôte désirant rejoindre le groupe répond avec un message *membership report* **[71]**.

Il est à noter, qu'un membre d'un groupe multicast peut être une station de base ou une unité mobile.

Dans **[30]**, Acharya-Badrinath ont introduit les groupes multicast des unités mobiles et ils ont associé une *vue des hôtes* (*hote view*) à chaque groupe. Les vue des hôtes d'un groupe multicast représente l'information de localisation du groupe ; chaque membre d'un groupe multicast est localisé dans la cellule de certaine MSS dans la vue des hôtes.

Définition : **[30]**

Un groupe multicast G est définit par le tuple {E_G , H_G}. Il est associé aux attributs suivants :

Members E_G : L'ensemble des MHs qui forment le groupe multicast G.

Host-view H_G : L'ensemble des MSSs tel que au moins une MH appartient à E_G est locale à chaque MSS dans cet ensemble.

Cependant, pour délivrer un message multicast à un groupe d'MHs, il suffit d'envoyer une copie seulement aux MSSs figurant dans *host- view* du groupe. Le fait que la vue des hôtes change dynamiquement dû à la mobilité des membres du groupe, la vue est traitée (dans les protocoles proposés dans **[30]**) comme un élément de données répliqué dont sa cohérence est gérée par un protocole « *host- view membership* ».

Il existe plusieurs travaux, par exemple **[69, 70]**, qui sont basés sur l'approche décrite dans **[30]** pour traiter le problème de la livraison fiable des messages multicast en utilisant la notion du groupe multicast ou bien pour proposer de nouveaux protocoles dans ce cadre.

4.7 Conclusion

Ce chapitre a été consacré à la livraison multicast des messages aux unités mobiles. Le multicasting dans un environnement de calcul mobile est un problème crucial et important du moment où la mobilité impose d'autres challenges non traités auparavant.

Nous avons, tout d'abord, définit le problème du multicasting en présentant les effets de la mobilité des unités mobiles sur la livraison fiable des messages multicast. Nous avons suggérer nécessaire de présenter les protocoles proposés par A. Acharya et B. Badrinath dans **[30]** du moment où ils représentent des schémas de base pour la conception de plusieurs protocoles. En suite, nous avons donner un aperçu sur la notion du groupe multicast qui représente une nouvelle approche de traitement du problème de la livraison fiable des messages multicast aux unités mobiles.

Chapitre 5

Un algorithme de livraison multicast de messages en environnement mobile

5.1 Introduction

Si l'envoi de données à un unique récepteur (*unicasting*) ou à tous les récepteurs sur un réseau donné est un problème important, le *multicasting* représente un problème crucial et qui vise à délivrer des données à un ensemble de récepteurs sélectionnés.

Dans ce chapitre, nous présentons un nouveau algorithme pour la livraison multicast des messages à des unités mobiles dans un réseau mobile. Le protocole est basé sur une nouvelle stratégie (déjà décrite dans le chapitre 2) qui traite les unités mobiles comme étant des messages circulant sur le réseau. Cet algorithme exploite les avantages offerts par l'algorithme du snapshot de Chandy-Lamport ainsi que l'algorithme d'*unicasting* proposé par Amy Murphy. Donc, l'objectif de notre contribution est d'étudier l'applicabilité du mécanisme du snapshot et la nouvelle stratégie de Amy Murphy pour l'implémentation d'un nouveau protocole qui résout le problème de la livraison multicast des messages en environnement mobile.

Ce chapitre est structuré comme suit : nous présentons en premier lieu les hypothèses sur lesquelles se base notre protocole. Par la suite, nous citons les différentes structures de données utilisées par l'algorithme. avant de donner les détails sur notre algorithme, nous expliquons tout d'abord son fonctionnement. Le paragraphe suivant sera consacré à la preuve de correction de notre solution proposée. Finalement, nous présentons les critiques de l'algorithme et nous terminons par une conclusion.

5.2 Les hypothèses

- Le modèle du système est le modèle *cellulaire*.
- Le système est composé de N_{MSC} stations de base et de N_h unités mobiles avec $N_{MSC} << N_h$.
- Chaque cellule est identifiée par le couple (MSC, RBS).
- Chaque station de base RBS est contrôlée par un seul centre support mobile MSC.
- Il existe un canal *filaire* entre chaque MSC.
- Les liens de communication utilisés sont supposés être *fiables*.
- Les canaux sans fil sont FIFO et bidirectionnels.
- Les canaux entre les MSCs sont FIFO et unidirectionnels (pour modéliser un canal bidirectionnel, on utilise deux canaux unidirectionnels dans des sens opposés).
- Lorsqu'une unité mobile h_i quitte sa cellule contrôlée par MSC_i en rejoignant une autre cellule contrôlée par MSC_j , elle doit se déplacer sur le canal qui relie MSC_i à MSC_j.

5.3 Les structures de données

Chaque MSC (appelé aussi nœud) possède :

Etat $_{MSCi}$: Indique l'état d'un MSC. La variable prend les trois valeurs suivantes : non- impliqué dans le snapshot local,
impliqué dans le snapshot local et
terminé le snapshot local.

Msg $_{MSCi}$: Passe à vrai si le MSC reçoit une copie du message multicast, et passe à faux sinon.

MSC_i *-Buf* : Représente la liste des messages sauvegardés et non encore délivrés.

MSC_i *-local* : Représente la liste de toutes les unités mobiles localisées dans la cellule couverte par MSC_i .

CE $_{MSCi}$: Indique la liste de tous les canaux entrants à MSC_i .

CS $_{MSCi}$: Indique la liste de tous les canaux sortants d'un MSC_i .

Dest(m) : Indique la liste des destinataires du message *m*.

Marqué $_{MSCi}$: Tableau de booléen indexé par les numéros des canaux entrants à MSC_i. Initialement est à faux pour indiquer que les canaux ne sont pas encore

marqués. Si le canal est marqué alors l'entrée du tableau correspondante à ce canal passe à vraie.

Deliver $_{MSCi}$: Tableau de booléen indexé par les identificateurs des unités mobiles. Initialement est à faux. Si une unité mobile destinataire a reçu le message qui lui est délivré, l'entrée correspondante à cette unité mobile passe à vraie.

ArrMobile $_{hi}$: Variable booléenne qui prend la valeur vraie si l'unité mobile h_i arrive à un MSC, faux sinon.

5.4 Fonctionnement de l'algorithme

Initialement, une unité mobile h_i désire envoyer un message m à un ensemble de destinataires d'unités mobiles *dest(m).* L'émission se fait en deux étapes : tout d'abord la transmission du message à partir du site mobile émetteur h_i vers son MSC_i , en suite l'émission du message vers les destinataires sélectionnés dans *dest(m).*

A la réception du message par le MSC_i , l'algorithme débute et le MSC_i commence le snapshot local en exécutant les étapes suivantes :

a) Enregistre son état qui passe de non- impliqué à l'état impliqué.

b) Sauvegarde une copie du message ainsi que la liste des destinataires.

c) Faire envoyer {*m*, *dest(m)*}sur tous les canaux sortants.

A la réception de {*m*, *dest(m)*} par chaque MSC_j dans le système, ils exécutent les mêmes étapes a), b) et c) décrites ci-dessus. Il est à noter qu'avant d'exécuter l'étape c) et afin de diminuer la charge des canaux, chaque MSC va enlever les unités mobiles destinataires qui sont localisées dans sa cellule de la liste *dest(m),* c.-à-d. $dest(m) = dest(m) - h_i$ tel que $h_i \in MSC_i$ *–local*. En suite, il fait envoyé la nouvelle liste de destinataires sur tous ses canaux sortants.

Lorsqu'un *MSC* possède une copie du message, il doit délivrer ce message à leur unités mobiles destinataires. On distingue deux situations : l'arrivée d'un message à un MSC et l'arrivée d'un mobile à un MSC. Plusieurs cas peuvent se présenter :

(1) L'arrivée d'un message à un MSC qui possède déjà une copie du message Þ aucune action n'est prise par cette MSC.

(2) Le message arrive et l'une des unités mobiles destinataires est en communication avec un MSC (elle est localisée) Þ le MSC doit délivré le message à cette unité mobile.

(3) **a)** L'arrivée d'une unité mobile à un MSC qui ne possède pas une copie du message Þ dans ce cas, il n'y a pas de livraison de message.

b) L'arrivée d'une unité mobile à un MSC sur un canal marqué Þ dans ce cas, le MSC ne doit pas délivrer le message car le mobile a déjà reçu le message par un autre MSC.

c) L'arrivée d'une unité mobile $h_i \notin dest\ (m)$ à un MSC Þ pas de livraison.

(4) L'arrivée d'une unité mobile à un MSC sur un canal non marqué Þ dans ce cas, le MSC doit délivré le message à cette unité mobile.

5.5 Code source de l'algorithme

Lorsqu'une unité mobile h_i désire envoyer un message m à un ensemble de destinataires *dest(m),* elle envoi d'abord ce message à son MSC_i pour qu'il s'occupe de l'émission de ce message aux destinataires voulus.

Send (m, dest(m)) de $h_i \rightarrow MSC_i$
/* Lorsque h_i décide d'émettre un message m à un ensemble de destinataires *dest(m).*
Begin
Envoyer (*m*, *dest(m)*) à MSC_i via un canal sans fil.
End.

a) Partie d'initialisation

```
Init ()
Begin
    Pour chaque MSCi
    Do
        Etat MSCi := non- impliqué ;    Msg MSCj := false ;
        Pour chaque canal ce ∈ CE MSCi
        Do
        Marqué MSCi [ce] := false ;
        Endo ;
        Pour chaque hi ∈ Dest(m)
        Do
        Deliver [hi] := false ; ArrMobile hi := false ;
        Endo ;
    Enddo ;
End.
```

b) Partie de traitement

A la réception du message par MSC_i qui va prendre le rôle d'initiateur, le processus d'envoi débute. Le MSC_i lance le calcul du snapshot local, il enregistre son état, il sauvegarde une copie du message, en suite il fait une propagation du message sur tous les canaux de sortie. Après y avoir exécuter ces étapes, il passe à la livraison du message aux destinataires sélectionnés dans *dest(m).*

L'initiateur MSC_i exécute :

```
Initiator ()
/* A la réception de (m, dest(m)) de h_i.
Begin
    Etat_MSCi := impliqué ;                    /* L'enregistrement d'état.
    Msg_MSCi := true ;
    Insérer (m, dest(m)) dans MSC_i-Buf ;      /* Sauvegarder le message.
    If dest(m) ∩ MSC_i-local ≠ ∅
       then
            Pour chaque h_k Î dest(m) ∧ h_k Î MSC_i-local
               Do
                  dest(m) = dest(m) – h_k ;
               Enddo ;
    Endif ;
    Pour chaque canal cs ∈ CS_MSCi
       Do
            Envoyer (m, dest(m)) sur cs ;
       Endo ;
    /* Délivrer m aux destinataires locaux.
     Déliver (m) ;
     If ∀ ce ∈ CE_MSCi ∧ Marqué_MSCi [ce] = true
       then
            /* Supprimer m et les destinataires de m.
           Delete(m, dest(m)) ;
        else
           /* Attendre l'arrivée d'une unité mobile sur un canal entrant.
           ArriveMobile() ;
    Endif ;
End.
```

Lorsque les autres MSCs reçoivent *(m, dest(m))* envoyé par l'initiateur MSC_i, ils vont procédés de la même façon que l'initiateur à une exception prés et ce en enregistrant leurs états locaux en suite faire propager le message sur tous les canaux de sortie après avoir sauvegarder une copie du message.

Chaque MSC_j (j ¹ i) exécute :

```
Non- initiator ()
/* A la réception de (m, dest(m)) par sur un canal entrant ce.
Begin
    Marqué MSCj [ce] := true ;                    /* Le marquage du canal ce.
    If Msg MSCj = false
      then
          Etat MSCj := impliqué ;                 /* L'enregistrement d'état.
          Msg MSCj := true ;
          Insérer (m, dest(m)) dans MSCj -Buf ;   /* Sauvegarder le message.
          If dest(m) ∩ MSCj -local ≠ ∅
             then
                 Pour chaque hk Î dest(m) ∧ hk Î MSCj -local
                     Do
                        dest(m) = dest(m) – hk ;
                     Enddo ;
          Endif ;
          Pour chaque canal cs ∈ CS MSCj
          Do
          Envoyer (m, dest(m)) sur cs ;
          Endo ;
          /* Délivrer m aux destinataires locaux.
           Déliver (m) ;
          If ∀ ce ∈ CE MSCi ∧ Marqué MSCi [ce] = true
             then
                  /* Supprimer m et les destinataires de m.
                 Delete(m, dest(m)) ;
              else
                 /* Attendre l'arrivée d'une unité mobile sur un canal entrant.
                 ArriveMobile() ;
          Endif ;
    Endif ;
End.
```

Si un MSC est impliqué dans le snapshot local, il doit délivrer le message *m* aux destinataires de *m* c.-à-d. *dest(m)*. Dans ce cas, il est responsable de la livraison du message aux unités mobiles destinataires et qui sont localisées dans sa cellule (elles sont en communication avec leur MSC) ou bien qui se trouvent sur des canaux entrants à cet MSC.

A la réception de *{m, dest(m)}* par l'initiateur en prévenance des autres MSCs, il doit marquer le canal sur lequel cette information arrive.

```
Iinitiator ()
/* A la réception de (m, dest(m)) par sur un canal entrant ce.
Begin
Marqué MSCi [ce] := true ;                    /* Le marquage du canal ce.
End.
```

Avant de passer à la phase de livraison, on doit distinguer les deux situations suivantes :

- Certaines unités mobiles destinataires appartiennent à la liste locale d'un MSC et donc sont localisées par cet MSC. Par conséquent, cet MSC va délivrer localement le message à ces unités mobiles.

- D'autres unités mobiles destinataires appartiennent à d'autres cellules couvertes par d'autres MSCs. Dans ce cas, si ces unités mobiles ont quitté leurs cellules en se dirigeant vers d'autres, elles doivent se déplacer sur des canaux entrants aux MSCs vers lesquels elles se dirigent. Par conséquent, les nouveaux MSCs vont être les responsables de la livraison du message à ces unités mobiles, bien sûr si elles n'ont pas déjà reçu une copie du message par leurs anciens MSCs. Par exemple, si une unité mobile h_i quitte sa cellule couverte par MSC_i en rejoignant la cellule couverte par MSC_j , elle doit se déplacer sur un canal entrant à MSC_j . Si le canal sur lequel h_i arrive à MSC_j est marqué par ce dernier, alors MSC_j ne doit pas délivrer le message à h_i car elle a déjà reçu une copie du message par son ancien MSC. Dans le cas où le canal ne serait pas marqué et si $h_i \in dest(m)$, alors MSC_j doit délivré le message à h_i .

```
Déliver (m) to dest(m)
Begin
    Pour chaque h_i ∈ dest(m) ∧ h_i ∈ MSC_i-local
    Do
        /* Délivrer le message à h_i
        HDéliver(m) ;
    Enddo ;
End.
```

```
HDéliver (m)
Begin
    If Deliver[h_i ] = false   then  Délivrer m  à h_i ;
                                     Deliver[h_i ] := true ;
    Endif ;
End.
```

```
ArriveMobile ()
Begin
    While ArrMobile_hj = false
       Do
          Attendre l'arrivée d'une unité mobile ;
       Enddo ;
    /* A l'arrivé d'une unité mobile h_j à un  MSC_j sur un canal entrant  ce. MSC_j
    marque son arrivé dans la procédure handoff' en mettant
    ArrMobile_hj := true.

    If Msg_MSCj = true
       then  If h_j  Î  dest(m)
                then If Marqué_MSCj [ce] = false
                        then  Hdéliver (m) ;
             Endif ;
    Endif ;
End.
```

Si le snapshot local arrive à sa fin (terminé) sans l'enregistrement de certaines unités mobiles destinataires comme une partie de cet état, le MSC supprime sa copie du message, assurant que ces unités mobiles vont apparaître dans d'autres snapsots locaux d'autres MSCs.

Un MSC prend la décision de supprimer sa copie du message lorsque tous les canaux entrants à cet MSC sont marqués et lorsque toutes les unités mobiles destinataires localisées dans la cellule couverte par cet MSC ont reçu une copie du message.

Remarque : Le fait que nous ayons supposé que les liens de communication sans fil soient fiables, nous n'avons pas besoin de vérifier si une unité mobile a reçu sa copie du

message envoyé ou pas. Dans le cas où nous n'aurions pas cette hypothèse (c.-à-d. la perte des messages est possible du moment où les liens ne sont pas fiables), il faut s'assurer que le message envoyé est reçu par toutes les unités mobiles avec succès et ce en utilisant un simple mécanisme d'envoi d'ACKs. Avec ce mécanisme, lorsqu'une unité mobile reçoit un message, elle doit répondre par un ACK envoyé à son MSC.

Delete (m, dest(m))
/* La suppression de (*m, dest(m)*) par un MSC.
Begin
Etat $_{MSCi}$:= terminé ;
Msg $_{MSCj}$:= false ;
Enlever (*m, dest(m)*) dans MSC_i *-Buf* ;
Pour chaque canal $ce \in CE_{MSCi}$
Do
Marqué $_{MSCi}$ *[ce]* := false ;
Endo ;
End.

5.6 Module handoff

Un *handoff* est déclenché lorsqu'une unité mobile quitte sa cellule en entrant dans une autre cellule.

Supposons qu'une unité mobile h_i résidante dans la cellule couverte par MSC_i se déplace vers une autre cellule couverte par un autre MSC_j. Lorsque h_i découvert qu'elle a entré dans une autre cellule, elle envoi un message *greeting*(h_i , MSC_i) à MSC_j en présentant dans ce message son identité ainsi que l'identificateur de son ancien MSC.

A la réception de ce message par MSC_j , il envoi à un message *deregister*(h_i) à MSC_i afin que ce dernier supprime cette unité mobile de sa liste locale MSC_i *-local*. Donc, Le module *Handoff* débute son exécution en recevant le message *deregister*(h_i) *de* MSC_j.

Le module *Handoff* au niveau de MSC_i traite le message *deregister*(h_i) comme suit : (a) il supprime h_i de sa liste locale MSC_i *–local*, (b) il répond à son tour par un envoi du message *register*(h_i,< *data*>) à MSC_i contenant les structures de données concernant cette unité mobile, c.-à-d. *data = {deliver*$_{hi}$*}*.

A la réception de ce message par MSC_j, le module *Handoff* au niveau de MSC_j traite le message *register(h_i,< data>)* comme suit : (a) il ajoute h_i à sa liste locale MSC_j*-local*, (b) il marque l'arrivée de h_i en mettant $ArrMobile_{hi}$ à true. A ce moment la procédure *Handoff* arrive à sa fin et à partir de là MSC_j se charge de la livraison des messages destinés à h_i. L'information $ArrMobile_{hi}$ = true fournie par le module *Handoff* déclenche la fonction *ArriveMobile()*.

5.7 Preuve de correction

Définition

On dit qu'un algorithme assure la livraison multicast des messages aux unités mobiles si et seulement s'il vérifie les deux propriétés suivantes :

1. Toutes les unités mobiles destinataires ont reçu le message à délivrer.

2. Chaque unité mobile destinataire a reçu une et une seule copie du message à délivrer.

En plus de ces deux propriétés, l'algorithme doit vérifié une troisième propriété le fait qu'il soit basé sur la notion du snapshot et qui est la suivante :

3. Après que le snapshot arrive à sa fin, toute information concernant le message à délivrer doit être supprimée du système, c.-à-d. le système doit retrouvé son état initial.

La première propriété

Pour montrer que le message a été délivré et par conséquent reçu par toutes les unités mobiles destinataires, il faut monter que :

$$\forall h_i \in dest(m) \Rightarrow deliver[h_i] = true \wedge \forall ce \in CE_{MSCi} : Marqué[ce] = true \quad \ldots\ldots(1)$$

Il faut noter, tout d'abord, que tous les MSCs du système sont impliqués dans le sanpshot. Donc tous les MSCs vont recevoir une copie du message à délivrer le fait que chaque MSC aille faire une propagation du message sur tous ses canaux de sortie et à

cause de la connectivité du graphe (c.-à-d. les liens de communications entre les MSCs forme un graphe complet).

A la réception du message, chaque MSC délivre une copie du message aux unités mobiles destinataires qui sont localisées par cet MSC c.-à-d. elles appartiennent à sa liste locale. Donc, nous aurons :

$$\forall h_i \in dest(m) \wedge h_i \in MSC_i\text{-}local \Rightarrow deliver[h_i] = true \quad \ldots\ldots(a)$$

Nous avons, donc, pour chaque MSC_i la propriété (a) est vérifiée. A un instant donné *t* du snapshot, tous les canaux entrants aux MSCs vont être marqués du moment où il y a une propagation du message sur tous les canaux de sorties de ces MSCs ; par conséquent, nous aurons la propriété suivante est vérifiée pour chaque MSCs du système :

$$\exists t : \forall ce \in CE_{MSCi} \Rightarrow Marqué[ce] = true \quad \ldots\ldots(b)$$

En combinant les deux propriétés (a) et (b), nous obtiendrons la propriété (1) qui sera vérifiée et donc, la preuve est terminée.

Supposons maintenant, que certaines unités mobiles destinataires ne sont pas localisées, c.-à-d. : $\exists h_j \in dest(m) \wedge h_j \notin MSC_j\text{–}local \Rightarrow deliver[h_j] = false$ ……(c)

Alors, nous sommes dans le cas où ces unités mobiles se trouveraient en déplacement sur des canaux non encore marqués et sur lesquels ces unités mobiles se déplacent, c.-à-d. : $\exists ce' \in CE_{MSCj} : Marqué[ce'] = false$ ……(d)

Nous sommes dans la situation suivante : une unité mobile se trouve sur un canal non marqué et elle est en avant du message.

Lorsque cette unité mobile arrive au MSC vers lequel elle se dirige, elle va sûrement recevoir le message du moment où tous les MSCs du système possèdent des

copies du message à délivrer et le fait que cette unité mobile n'a pas encore reçu le message. Donc, à l'arrivée de h_j à MSC_j , elle reçoit le message et donc *deliver[h_j]* passe à vrai. En suite, le message va arrivé après l'arrivée de h_j (car les canaux sont FIFO) et dans ce cas MSC_j marque le canal sur lequel arrive le message en mettant *Marqué[ce¢]* à vrai. Par conséquent, nous avons la propriété (1) qui est vérifiée et donc la preuve est terminée.

La deuxième propriété

Après avoir montrer que chaque unité mobile destinataire a reçu le message, il reste à montrer que chaque unité mobile destinataire va recevoir une seule fois le message. Pour le faire, nous utilisons l'invariant suivant $nb\ –livraison_{hi} £ 1$, c.-à-d. il faut vérifier : $" h_i Î dest(m) Þ nb –livraison_{hi} £ 1$……(2)

Initialement, la propriété (2) est vérifiée c.-à-d. $" h_i Î dest(m) Þ nb –livraison_{hi} = 0$.

Lorsque le message est délivré à une unité mobile destinataire, le nombre de livraison de cette unité mobile s'incrémente de 1, c.-à-d. $nb –livraison_{hi} = 1$. Nous avons deux cas possible où le nombre de livraison d'une unité mobile peut passer à 1.

- Le message *m* arrive à un MSC sur un canal non marqué et cet MSC ne possède pas une copie du message et certaines unités mobiles destinataires sont localisées.

- Une unité mobile destinataire arrive sur un canal non marqué à un MSC possédant une copie du message.

Dans les deux cas, une première livraison est permise et d'autres livraisons ne doivent pas se faire. Donc, il faut montrer qu'après la première livraison, les unités mobiles destinataires ne doivent pas recevoir une deuxième copie du même message.

1er cas :

Supposons qu'une unité mobile destinataire h_i a reçu une copie du message *m* par MSC_i , c.-à-d. $\$ h_i Î dest(m) Ù deliver[h_i] = true Þ nb –livraison_{hi} = 1$.

Supposons maintenant que le message *m* arrive sur un canal non marqué à MSC_i (cet MSC possède déjà une copie du message), MSC_i va marqué le canal sur lequel arrive ce message. Et en va supposer que MSC_i va délivrer encore ce message aux unités mobiles destinataires qui sont localisées dans sa cellule entre autre h_i . Dans ce cas le nombre de livraison de h_i va passer à 2, mais cette livraison n'est permise que si *deliver* $[h_i]$ = *false*, mais se n'est pas le cas car h_i a déjà reçu une copie du message c.-à-d. *deliver* $[h_i]$ = *true*. Ce qui mène à une contradiction avec la condition de livraison.

Donc, le fait que h_i a reçu une copie du message ce qui implique *deliver*$[h_i]$ = *true*, la deuxième livraison n'est pas possible car la condition de livraison (*deliver*$[h_i]$ = *false*) n'est pas vérifiée. Alors h_i ne peut recevoir qu'une seule fois le message, c.-à-d. nous avons toujours la propriété (2) qui est vérifiée et par conséquent la preuve est terminée.

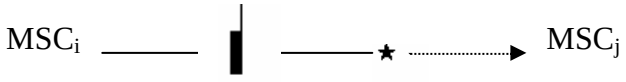
2^{eme} *cas :*

Supposons qu'une unité mobile destinataire h_i arrive sur un canal marqué à MSC_j possédant une copie du message *m*. Donc, la propriété suivante est vérifiée :

$\$ h_i \hat{I} dest(m) \grave{U} deliver [h_i] = true \grave{U} ArrMobile_{hi} = true \eth nb\text{–}livraison_{hi} = 1.$

Supposons maintenant que h_i va recevoir une copie du message par MSC_j dès son arrivée, donc, le nombre de livraison de cette unité mobile va passé à 2 ce qui viole la condition de livraison. Cette livraison n'est permise que si nous avons la condition de livraison est vérifiée c.-à-d. *deliver* $[h_i]$ = *false*, mais ce n'est pas le cas ici. Donc, une deuxième livraison n'est pas possible et le nombre de livraison ne passe jamais à 2.

On note que si h_i arrive à MSC_j sur un canal marqué, alors nous sommes dans la situation suivante : h_i se trouve sur un canal marqué et le message est en avant de h_i.



Pour que h_i reçoive une autre copie du message dès son arrivée à MSC_j , il faut qu'elle arrive avant le message. Donc, elle doit dépasser le message *m* ce qui est impossible car les canaux sont supposés être FIFO. Par conséquent, la livraison de *m* aux unités mobiles destinataires est faite une et une seule fois, ce qui mène à la vérification de la deuxième propriété.

La troisième propriété

Pour montrer cette propriété, il faut montrer que l'algorithme atteint un état où il n'y a aucune copie du message à délivrer sauvegardée à n'importe quel MSC et il n'y a aucun message sur n'importe quel canal et tous les canaux sont vides.

Lorsqu'un MSC reçoit une copie du message *m* sur chacun de ses canaux entrants, il va passé à l'état impliqué et il va marquer ces canaux sur lesquels le message arrive. Lorsque cela arrive, l'action de nettoyage ou de suppression est permise du moment où la condition de suppression est vérifiée. Donc, cet MSC va passer à l'état terminé. En plus, une fois qu'un message est passé à travers un canal, il n'y a pas une possibilité d'envoyer un autre message sur ce même canal. Donc, il n'y aura aucun message sur les canaux lorsque le système atteint l'état de terminaison et par conséquent, la propriété est prouvée.

5.8 Les critiques de l'algorithme

a) Avantages

- L'exécution du protocole est à la charge des MSCs, tandis que le rôle des unités mobiles se réduit à l'émission et la réception des messages.

- Les structures de données au niveau des MSCs sont crées lorsqu'il y a une livraison multicast, c.-à-d. lorsque le snapshot débute. Et elles sont supprimées à la fin du snapshot et par conséquent, à la fin de la livraison multicast.

- La décision de suppression du message est prise automatiquement par les MSCs en examinant les canaux entrants s'ils sont marqués ou pas, sans faire recoure à l'initiateur du multicast. Donc, ce n'est pas à l'initiateur de donner l'ordre de suppression du message et mettre fin au snapshot local.

- Si l'algorithme arrive à sa fin, nous somme sûr que toutes les unités mobiles destinataires ont reçu leurs copies du message multicast.

b) Inconvénients

- Avec cet algorithme, tous les MSCs sont impliqués dans le snapshot.
- Avec ce protocole, il n'y a qu'une seule livraison multicast à la fois, c.-à-d. il n'y a pas d'autres envois de messages jusqu'à ce que le snapshot arrive à sa fin.

c) Performance et complexité

Soient C_f et C_{sf} les coûts de l'émission fiable d'un message respectivement sur un lien de communication filaire et un lien de communication sans fil.

Soient N_{MSC} et $N_{dest(m)}$ le nombre d'MSCs et le nombre d'unités mobiles destinataires dans le système respectivement.

Concernant le coût d'exécution du protocole *Handoff* par mouvement est : $2C_f + C_{sf}$ pour les messages *greeting()*, *register()* et *deregister()*.

Avec notre protocole, au pire des cas, le coût de la livraison d'un message multicast à un nombre $N_{dest(m)}$ d'unités mobiles destinataires est :

$$(N_{dest(m)} + 1)C_{sf} + C_f \sum_{i=1}^{NMSC} nbcs\,[i]$$

Où : nb_{cs} [i] représente le nombre de canaux sortants du MSC identifié par *i*.

Il est à noter que le coût de la livraison d'un message multicast à un nombre $N_{dest(m)}$ d'unités mobiles destinataires de notre protocole est inférieur au coût de la livraison d'un message multicast à un nombre $N_{dest(m)}$ d'unités mobiles destinataires de l'algorithme proposé dans **[30]**.

5.9 Conclusion

Dans ce chapitre, nous avons présenté un algorithme pour la livraison multicast des messages aux unités mobiles dans un modèle du système cellulaire. Cet algorithme garantit les propriétés d'une livraison multicast dans un environnement mobile. Le protocole exploite les avantages de l'algorithme du snapshot de Chandy-Lamport et ainsi que les avantages de la nouvelle stratégie proposée par Amy Murphy et qui traite les unités mobiles comme étant des messages persistants se déplaçant sur des canaux FIFOs. Comme tout algorithme, notre algorithme présente des avantages et souffre de quelques inconvénients. Parmi ses avantages, nous citons la décision de suppression du message multicast qui est prise automatiquement par les MSCs sans faire recoure à l'initiateur de la livraison multicast comme nous le trouvons dans d'autres protocoles, par exemple dans **[30]**.

Conclusion générale

L'apparition d'un nouveau environnement de calcul, appelé ***environnement mobile*** ou ***nomade*** a introduit une gamme de nouveaux problèmes non traités auparavant. Ces problèmes sont induits par les caractéristiques liées au médium de communication sans fil et aux sites mobiles.

Le terme *mobilité* a été utilisé toujours pour faire référence aux processus qui migrent à travers le réseau ; Amy. Murphy suggère encore une autre manière d'introduire la mobilité et ce en traitant les unités mobiles comme étant des messages nomades qui préservent leurs identités lorsqu'ils traversent le réseau. Cette nouvelle stratégie à donner naissance à un nouveau modèle.

Cependant, le problème majore qui s'impose avec ce genre de modélisation est la livraison des messages aux unités mobiles et qui reste un problème crucial à cause des déplacements fréquents des unités mobiles.

Dans une première partie de cette thèse, nous avons étudier ce nouveau modèle proposé par Amy. Murphy dans le but d'adapter les algorithmes distribués déjà établis pour résoudre des problèmes liés au calcul mobile à savoir la livraison des messages. Pour garantir une livraison des messages dans toutes les circonstances, une alternative d'algorithme de diffusion basée sur la notion classique du *snapshot* a été utilisée. Ce travail de Amy. Murphy est un protocole de livraison *unicast* des messages, basé sur l'algorithme de snapshot de Chandy-Lamport.

Dans la seconde étape de notre travail, nous avons proposé un algorithme de livraison *multicast* des messages aux unités mobiles. Le protocole est basé sur la nouveau modèle dont lequel les unités mobiles sont vues comme des messages nomades. Nous avons essayer d'exploiter les avantages offerts par l'algorithme

d'unicasting de Amy. Murphy ainsi que l'algorithme de snapshot sur lequel il se base et ce afin de développer un protocole de livraison multicast des messages en environnement mobile.

Notre protocole assure la livraison exactement une fois du message multicast à toutes les unités mobiles destinataires. En plus, la copie du message sauvegardée est automatiquement supprimer par les MSCs (Mobile Support Center) sans faire recoure à l'initiateur du multicast.

Toutefois, cet algorithme présente deux inconvénients : tous les MSCs sont impliqués dans la livraison et il n'y a qu'une seule livraison multicast à la fois. Ces inconvénients sont dus non pas à l'algorithme proposé mais dus à l'algorithme de snapshot de Candy-Lamport et au modèle sur lequel notre algorithme se base.

Dans le but de surmonter ces inconvénients, une des perspectives de ce travail est d'essayer d'utiliser la notion de *groupe multicast* afin d'avoir que les MSCs concernés par la livraison qui doivent participés à livraison multicast des messages. Cependant, pour réaliser ceci, il faut une redéfinition du modèle initial sur lequel notre algorithme s'est basé.

Bibliographie

[1] M. Raynal. *Distributed Algorithms and Protocols*. Wiley and Sons, 1988.

[2] B. R. Badrinath, A. Acharya et T. Imielinski. *Designing Distributed Algorithms for Mobile Computing NetWorks.* Rapport technique, Departement of Computer Sciences, University of Rutgers, US, 1994.

[3] B. R. Badrinath, A. Acharya et T. Imielinski. *Structuring Distributed Algorithms for Mobile Hosts.* In 14th International Conference on Distributed Computing Systems, Poznan, Poland, pages 21-28, June 1994.

[4] B. R. Badrinath, A. Acharya et T. Imielinski. *Impact of Mobility on Distributed Computations.* ACM Operating Systems Review, vol. 27, n° 2, April 1993.

[5] Baggio. *Environnements Mobiles : Etude et Synthèse Bibliographique.* DEA de Systèmes Informatiques, au Laboratoire MASI, Université Pierre et Marie Curie, 1995.

[6] J. Ioannidis, D. Duchamp, et G. Q. Maguire. *IP-Based Protocols forMobile Internetworking.* In Proc. of ACM SIGCOMM Symposium on Communication, Architectures and Protocols, pages 235–245, September 1991.

[7] N. Badache. *La Mobilité dans les Systèmes Répartis.* Publication Interne IRISA n° 962, Octobre 1995. ftp://ftp.irisa.fr /techreports/1995/PI–962.ps.gz.

[8] N. Aleb. *Spécification des Systèmes en Environnement Mobile à l'aide du Pi-calcul.* Thèse de Magister, USTHB, Octobre 2000.

[9] E. Pitoura, B. Bhargava. *Dealing with Mobility: Issues and Research Challenges*. Technical Report TR-93-070, Department of Computer Science, Purdue University, November 1993.

[10] G. Forman, J. Zahorjan. *The Challenges of Mobile Computing*. Computer Science & Engineering, University of Washington, US, IEEE Computer, pages 39-47, April 1994.

[11] J. J. Kistler, M. Satyanarayanan. *Disconnected Operation in the Coda File System*. Thirteenth ACM Symposium on Operating Systems Principles, Asilomar Conference Center, Pacific Grove, US, vol. 25, pages 213-225, 1991.

[12] B. Marsh, F. Douglis, R. Cáceres. *System Issues in Mobile Computing*. Matsushita Information Technology Laboratory, Technical Report MITL-TR-50-93, February 1993.

[13] T. Imielinski et B. R. Badrinath. *Mobile Wireless Computing: Solutions and Challenges in Data Management*. CACM, vol. 37, n° 10, pages 18-28, October 1994.

[14] E. Pitoura, B. Bhargava. *Building Information Systems for Mobile Environnements*. Department of Computer Sciences, Purdue University, Third International Conference on Information and Knowledge Management, pages 371-378, November 1994.

[15] M. Weiser. *Les Réseaux Informatiques de l'An 2000*. Pour la Science, n° 169, pages 72-84, novembre 1991.

[16] F. Bennett, T. Richardson, A. Harter. *Teleporting Applications Mobile*. Olivetti Research Laboratory, IEEE Workshop on Mobile Computing Systems and Applications, Santa Cruz, CA, US, December 1994.

[17] T. Imielinski, B. R. Badrinath. *Data Management for Mobile Computing*. Department of Computer Science, Rutgers University 2, SIGMOD RECORD, vol. 22, n° 1, pages 34-39, March 1993.

[18] T. Imielinski, B. R. Badrinath. *Mobile Wireless Computing: Challenges in Data Management*. Department of Computer Science, Rutgers University.

[19] B. R. Badrinath, T. Imilelinski, A. Virmani. *Locating Strategies for Personal Communication Networks*. Department of Computer Science, Rutgers University, IEEE

Globecom 92 Workshop on Networking of Personal Communications Applications, Orlando, US, December 1992.

[20] T. Imielinski, B. R. Badrinath. *Querying in Highly Mobile Distributed Environments.* Department of Computer Science, Rutgers University, 18th Very Large Data Bases, pages 41-52, August 1992.

[21] K. S. Meier-Hellstern, E. Alonso, D. O'Neil. *The Use of the SS7 and GSM to Support High Density Personal Communications*. Third WinLab Workshop on Third Generation Wireless Information Networks, pages 49-57, April 1992.

[22] L. J. Ng et al. *Distributed Architectures and Databases for Intelligent Personal Communication Networks.* ICWC, pages 300-304, June 1992.

[23] D. J. Goodman. *Trends in Cellular and Cordless Communications*. IEEE Communication Magazine, June 1991.

[24] J. Ioannidis, G. Q. Maguire Jr. *The Design and Implementation of Mobile Internetworking Architecture*. Columbia University, Usenix Winter 1993 Conference, January 1993.

[25] B. R. Badrinath, A. Acharya, T. Imielinski. *Impact of Mobility on Distributed Computations*. Department of Computer Science, Rutgers University, ACM Operating System Review, vol. 27, n° 2, pages 15-20, April 1993.

[26] N. Badache. *Ordre Causal et Tolérance aux Défaillances en Environnement Mobile.* Thèse de Doctorat, Département: Architecture et Systèmes, Laboratoire: Systèmes d'Exploitation et Réseaux. USTHB, Octobre 1998.

[27] G. Le lann. *Distributed Systems: Towards a Formal Approach*. In: IFIP Congress, pages 155-160, Toronto, August 1977.

[28] B. R. Badrinath, A. Acharya, T. Imielinski. *Structuring Distributed Algorithms for Mobile Hosts.* Department of Computer Science, Rutgers University, 14th International Conference on Distributed Computing Systems, Poznan, Poland, may 1994.

[29] B. R. Badrinath, A. Acharya, T. Imilelinski. *Designing Distributed Algorithms for Mobile Computing Networks.* Department of Computer Science, Rutgers University, 1994.

[30] Acharya et B. R. Badrinath. *A Framework for Delivering Multicast Messages in Networks with Mobile Hosts.* Rapport Technique n° DCS-TR-310, Department of Computer Science, University of Rutgers, 1994.

[31] C.Benzaid. *Estampillage et Ordre Causal dans les Environnements Mobiles*. Thèse de Magister, USTHB, Juillet 2003.

[32] L. Lamport. *Time, Clocks and the Ordering of Events in Distributed Systems.* CACM, vol. 21, n° 7, pages 558-565, 1978.

[33] S. Alagar et S. Venkatesan. *Causally ordered Message Delivery in Mobile Systems.* IEEE Transactions on Computers, vol. 46, n° 3, pages 353-361, March 1997.

[34] M. Raynal, A. Schiper et S. Toueg. *The Causal Ordering and a Simple Way to Implement it.* Information Processing Letters, 1991.

[35] E. W. Dijkstra et al. *Derivation of a Termination Detection Algorithm for Distributed Computation*. In Information Processing Letters, June 1983.

[36] Y. Amir et al. *Fast Message Ordering and Membership Using a Logical Token-Passing Ring.* In Proc. Of the 13th Intl. Conf. on Distributed Computing Systems, May 1993.

[37] K. Raymond. *A Tree-based Algorithm for Distributed Mutual Exclusion*. ACM Transactions on Computer Systems, 7(1), February 1989.

[38] B. R. Badrinath, A. Acharya et T. Imielinsky. *Structuring Distributed Algorithms for Mobile Hosts.* In Proceedings of Fourteenth International Conference on Distributed Computing Systems, pages 21-28, Poznan, Poland, 1994.

[39] D. B. Johnson. *Scalable Support for Transparent Mobile Host Internetworking.* In H. Korth and T. Imielinski, editors, Mobile Computing, pages 103-128. Kluwer Academic Publishers, 1996.

[40] R. Gray, D. Kotz, S. Nog, D. Rus et G. George. *Mobile Agents for Mobile Cmoputing*. Technical Report PCS0TR96-285, Dartmouth College, May 1996.

[41] M. Ranganathan, A. Acharya, S. Sharma et J. Saltz. *Network-aware Mobile Programs.*Technical Report CS-TR-3659, University of Maryland, College Park, 1997.

[42] A. Fuggetta, G. P. Picco et G. Vigna. *Understanding Code Mobility.* IEEE Transactions on Software Engineering, 24(5):342-361, 1998.

[43] A. L. Murphy. *Enabling the rapid Development of Dependable Applications in the Mobile Environment.* Thèse de Doctorat, Washington University, St. Louis, MO (USA), Aout 2000.

[44] C. E. Perkins. *IP Mobility Support.* Technical Report RFC 2002, IETF Networt Working Group, October 1996.

[45] M. Steenstrup. *Routing in Communication Networks, Chapiter 5.* Prentice-Hall, 1995.

[46] A. L. Murphy, G. C. Roman et G. Varghese. *Search and Tracking Algorithms for Rapidly Moving Mobiles.* Department of Computer Science, Washington University, June 1998.

[47] K. M. Chandy et L. Lamport. *Distributed Snapshots : Determining Global States of Distributed Systems.* ACM Transactions on Computing Systems, 3(1):63-75, 1685.

[48] A. L. Murphy, G. C. Roman et G. Varghese. *An Algorithm for Message Delivery in a Micromobility Environment.* Department of Computer Science, Washington University, St Louis, MO, April 1997.

[49] A. L. Murphy, G. C. Roman et G. Varghese. *Algorithms for Message Delivery in a Micromobility Environment.* Department of Computer Science, Washington University, St Louis, MO, February 1998.

[50] A. L. Murphy, G. C. Roman et G. Varghese. *An Algorithm for Message Delivery to Mobile Units.* In Proceedings of 16th Annual ACM Symposium on Principles of Distributed Computing (PODC ‘79), Santa Barbara, CA, USA, November 1997.

[51] M. Steenstrup. *Routing in Communication Networks, Chapiter 10.* Prentice-Hall, 1995.

[52] J. Moy. *OSPF Version 2*. Internet Engineering Task Force, 1994.

[53] A. Athas et D. Duchamp. *Agent-Mediated Message Passing for Constrained Environments*. In USENIX Synposium on Mobile and location- Independent Computing, August 1993.

[54] M. Bender et al. *Unix for Nomads : Making Unix Support Mobile Computing*. In USENIX Synposium on Mobile and location- Independent Computing, August 1993.

[55] T. Imielinski, S. Viswanathan et B. R. Badrinath. *Power Efficient Filtering of Data on the Air*. In EDBT'94, 1994.

[56] A. Acharya et B. R. Badrinath. *Delivering Multicast Messages in Networks with Mobile Hosts*. In Processing of 13th International Conference on Distributed Computing Systems, May 1993.

[57] T. Ballardie, P. Francis et J. Croweroft. *Core Based Trees (CBT) and Architecture for Scalable Inter-Domain Multicast Routing*. In Proc. of ACM SIGCOMM'93, 1993.

[58] S. E. Deering et D. R. Cheriton. *Multicast Routing in Datagram Internetworks and Extended Lans*. ACM Transactions on Computers Systems, May 1990.

[59] B. Rajagopalan. *Reliability and Scaling Issues in Multicast Communication*. In Proc. of ACM SIGCOMM Symposium on Communication, Architectures and Protocols, pages 188–197, 1992.

[60] S. Floyd, V. Jacobson, C. Liu, S. McCanne et L. Zhang. *A Reliable Multicast Framework for Light-Weight Sessions and Application Level Framing*. In Proc. of ACM SIGCOMM' 95, 1995.

[61] H. W. Holbrook, S. K. Singhal et D. R. Cheriton. Log-based Receiver-Reliable Multicast for Distributed Interactive Simulation. In Proc. of ACM SIGCOMM '95, 1995.

[62] J. C. Lin et P. Sanjoy. *RMTP: A Reliable Multicast Transport Protocol*. In INFOCOM '96, 1996.

[63] A. Achaya et B. R. Badrinath. *Checkpointing Distributed Applications on Mobile Computers*. In Processing of 3th International Conference on Parallel Distributed Information Systems, October 1994.

[64] A. Bakre et B. R. Badrinath. *I-TCP : Indirect TCP for Mobile Hosts.* In Processing of 15th International Conference on Distributed Computing Systems, May 1995.

[65] P. Bhagwat et C. E. Perkins. *A Mobile Networking System Based on Internet Protocol (IP).* In USENIX Symposium on Mobile and Location-Independent Computing, August 1993.

[66] F. Teraoka, Y. Yokote et M. Tokoro. *A Network Architecture Providing Host Migration Transparency*. In Proc. of ACM SIGCOMM '91, September 1991.

[67] H. Wada, T. Yozawa, T. Ohnishi et Y. Tanaka. *Mobile Computing Environment Based on Internet Packet Forwarding*. In 1992 Winter Usenix, January 1993.

[68] A. Achaya, A. Bakre et B. R. Badrinath. *IP Multicast Extensions for Mobile Internetworking*. In IEEE INFOCOM '96, 1996.

[69] R. Prakash, A. Schiper et M. Mohsin. *Reliable Multicast in Mobile Networks.*

[70] K. Brown et S. Singh. *The Problem of Multicast in Mobile Networks.* In Processing of 5th International Conference on Computer Communications and Networks (IC3N96), Washington DC, October 1996.

[71] I. Romdhani, M. Kellil, H-Y. Lach, A. Bouabdallah et H. Bettahar. *IP Mobile Multicast : Challenges and Solutions*. Va apparaître dans IEEE Communications Surveys and Tutorials, 2004.