N° D'ORDRE : 02/2009-M/IN

REBUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE D'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE DES SCIENCES ET DE LA TECHNOLOGIE

« HOUARI BOUMEDIENE »

FACULTE D'ELECTRONIQUE ET INFORMATIQUE

MEMOIRE

Présenté pour l'obtention du diplôme de MAGISTER

En : INFORMATIQUE

Spécialité : Programmation et Systèmes

Par : **ABDERAZEK SEBA**

Titre

Gestion de Groupe dans les Réseaux Mobiles Ad hoc

Soutenu le 14/10/2009, devant le jury composé de :

M. BELKHIR A.	Maître de Conférences, USTHB	Président
M. BADACHE N.	Professeur, USTHB	Directeur de thèse
Mme. MOUASSAOUI S.	Maître de Conférences, USTHB	Examinatrice
Mme. NOUALI N.	Maitre de Recherche A, CERIST	Examinatrice

Remerciements

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

J'exprime ma profonde reconnaissance et mes vifs remerciements à mon directeur de thèse le Pr Nadjib BADACHE, de m'avoir fait confiance en me proposant ce sujet. Je le remercie également pour ses lectures attentives et pour ses critiques et suggestions qui ont été d'un grand apport pour la finalité de ce travail.
Je remercie Mr Belkhir., Mme MOUSSAOUI et Mme NOUALI d'avoir accepté de juger ce travail.
J'adresse également mes sincères remerciements à Madame NADIA NOUALI, responsable du laboratoire des Systèmes Informatiques (C.E.R.I.S.T), pour sa compréhension, et pour tous les moyens qu'elle a mis à ma disposition.
Un grand MERCI, aux membres de ma famille et à mes très chers amis pour leurs encouragements, leur patience et leur amour.

Table des matières

Résumé

Assurer la gestion de groupes dans un réseau mobile ad hoc est un problème très complexe vu l'aspect dynamique et l'évolution rapide de la topologie de ces réseaux. En effet, les unités mobiles sont dynamiquement et arbitrairement éparpillées d'une manière où l'interconnexion entre les nœuds du réseau peut changer à tout moment. Le but du service de gestion de groupe est de masquer aux applications utilisant les groupes les problèmes de changement de configuration du groupe.

Nous avons commencé par faire un état de l'art des solutions existantes dans la littérature. Nous avons remarqué qu'aucune solution existante ne prend en compte la contrainte d'énergie des nœuds du groupe dans un réseau mobile ad hoc. De plus, les approches existantes génèrent un nombre excessif de messages de contrôle ce qui induit une consommation importante d'énergie.

Nous avons ensuite étudié les méthodes de préservation d'énergie dans les réseaux mobiles ad hoc et nous avons déduit que la préservation de l'énergie doit être présente dans toutes les applications et les services implémentés dans ces réseaux. Ceci nous a permis d'élaborer un service de gestion de groupe plus adapté aux réseaux mobiles ad hoc. En effet, notre approche inclut l'état énergétique des nœuds dans toutes les actions du service de gestion de groupe. Ceci permet en particulier de gérer les déconnexions des nœuds et mieux répartir les tâches afin de prolonger la durée de vie du groupe. Nous avons surtout géré les déconnexions dues aux pannes sèches et qui entrainent inévitablement des pertes importantes des données et dans les cas extrêmes la dissociation des groupes et la perte totale de l'effort collectif effectué. En plus, notre solution permet de répartir les tâches des membres du groupe selon leurs ressources effectives.

Mots clés :
Réseau mobile ad hoc, gestion de groupe, énergie

Introduction générale

Le développement d'applications sur les réseaux mobiles ad hoc est confronté aux différents défis induits par le changement fréquent de l'environnement de ces réseaux. Une des approches pour maîtriser cette complexité repose sur l'utilisation des groupes de nœuds sur le réseau, les applications s'exécutent au dessus du service de gestion de groupes qui gère l'aspect dynamique du réseau.

Dans ce travail, nous nous sommes intéressés aux différents problèmes liés au service de gestion de groupes dans les réseaux mobiles ad hoc. En effet, assurer la gestion de groupe dans un réseau mobile ad hoc est un problème très complexe vu l'aspect dynamique et l'évolution rapide de la topologie. En effet, les unités mobiles se déplacent, elles sont dynamiquement et arbitrairement éparpillées, par conséquent, l'interconnexion peut changer à tout moment. Le but du service de gestion de groupe est de masquer aux applications utilisant les groupes les problèmes de changement de configuration du groupe. Nous avons commencé par étudier les trois principales approches de gestion de groupes pour les réseaux mobiles ad hoc qui existent dans la littérature. Ces approches permettent d'identifier la position des nœuds et de palier aux problèmes des inexactitudes dans la détermination des membres d'un groupe. La première approche est une architecture multi-niveaux qui contient un protocole de couche de proximité synchrone pour déterminer l'ensemble des nœuds voisins d'un nœud donné dans le réseau. Cette information est utilisée par le protocole de gestion de groupe pour la construction des groupes utilisés par les applications distribuées. La seconde approche utilise une vue logique de la connectivité des nœuds pour masquer les déconnexions. Son but est de maintenir une vue consistante de l'état global du système. Dans la troisième approche une manière générique de construction du groupe est présentée.

L'étude de ces approches a révélé qu'elles nécessitent la génération d'un nombre excessif de messages, consommant trop d'énergie. Alors, nous nous sommes donné comme but principal la conception d'une solution de gestion de groupe qui prend en compte les informations sur l'état des énergies des nœuds et de l'inclure dans le service de gestion de groupe. En effet, une telle solution permet d'augmenter la pertinence du service de gestion de groupe en prévenant plus de déconnexions comme celles dues à la panne sèche qui entraine inévitablement des pertes importantes des données et dans les cas extrêmes la dissociations du groupe et la perte totale de l'effort collectif effectué. En plus, ça permet de répartir les tâches des membres du groupe selon leurs ressources effectives.

Ce document est organisé en cinq chapitres comme suit :

Le premier chapitre consiste en une présentation générale des Réseaux Mobiles Ad hoc où nous avons décrit les différents types de réseaux et leurs applications ainsi que les déférentes caractéristiques liées aux contraintes rencontrées dans ce type de réseaux. Nous décrivons également le problème de routage qui caractérise spécifiquement ces réseaux.

Le deuxième chapitre présente la notion de groupe de manière générale dans les réseaux conventionnels en présentant les différents axes de recherche dans le domaine et les différents problèmes liés au concept de groupe. Nous présentons aussi le modèle de groupe, ces caractéristiques et les conditions d'intégrité qui lui sont liées. Ce chapitre présente aussi la notion de la gestion de groupe qui a sollicité maints travaux ainsi que le service de communication de groupe et le problème de consensus.

Dans le troisième chapitre nous présentons l'état de l'art sur la gestion de groupe dans les réseaux mobiles ad hoc. Nous avons étudié trois approches pivot dans le domaine :

- Approche de Prakash et Baldoni [79] intitulée « Architecture pour la communication de groupe dans les réseaux ad hoc » où les auteurs proposent d'organiser le service de gestion de groupe sous forme d'une pile de protocoles. Dans cette architecture, le concept de gestion de groupe repose sur une couche de proximité dans le rôle est de contrôler le changement de configuration du réseau en s'appuyant sur les primitives de la sous couche MAC de la couche de base du réseau mobile
- Approche de roman et al. [86] intitulée « Un service consistant de gestion de groupe dans les réseaux ad hoc » où les auteurs proposent d'utiliser :
- un protocole de découverte de groupe pour déterminer qui est aux environs d'un nœud donné.
- un protocole de reconfiguration pour renforcer l'atomicité des changements de configuration qui comprend la fusion de groupes voisins et la scission d'un groupe menacé par la possibilité d'une déconnexion.
- Les concepts clés de ce protocole est la distance de sécurité entre les nœuds où les groupes qui déterminent si un nœud a ou non le temps d'exécuter le changement de configuration avant que la déconnexion ait lieu
- Approche de Sailhan et Liu [62] intitulée « Management de groupe pour les réseaux ad hoc » où les auteurs donnent une caractérisation des attributs du service de gestion de groupes dans les MANET, et proposent une conception du service de groupe en respect des attributs en questions.

Cette étude nous a permis de mettre en évidence les avantages et les manques de ces approches. En particulier, l'énergie qui est un élément clé dans les réseaux mobiles ad hoc n'est pas prise en compte. Nous proposons donc une solution qui tient compte de la consommation d'énergie.

Le quatrième chapitre fait la synthèse des mécanismes de réduction de la consommation d'énergie dans les réseaux mobiles ad hoc.

Le cinquième chapitre présente notre approche concernant le service de gestion de groupe dans les réseaux mobiles ad hoc, notre solution est une solution énergie aware et consiste à utiliser l'information sur l'énergie pour les différents modules du service et en particulier le module d'attribution de tâches. Notre approche permet aussi de prédire toute éventuelle panne ou déconnexion des nœuds et cela en utilisant une marge de sécurité basée sur le concept de distance et d'énergie de sécurité qui prévoit le temps pour récupérer toute donnée présente dans un nœud à risque.

Le document se termine par une conclusion qui trace le bilan de notre travail et dessine ses perspectives.

Chapitre 1

Les Réseaux Mobiles Ad hoc

1. Introduction

Les systèmes de communication cellulaire sont basés essentiellement sur l'utilisation des réseaux filaires (tel qu'Internet ou ATM) et la présence des stations de base qui couvrent les différentes unités mobiles du système. Les réseaux mobiles "ad hoc" sont à l'inverse, des réseaux qui s'organisent automatiquement de façon à être déployable rapidement, sans infrastructure fixe, et qui doivent pouvoir s'adapter aux conditions de propagation, aux trafics et aux différents mouvements des nœuds mobiles.

Les réseaux mobiles présentent une architecture originale. En effet, l'atténuation des signaux avec la distance, fait que le médium peut être réutilisé simultanément en plusieurs endroits différents sans pour autant provoquer de collisions. Ce phénomène est appelé *la réutilisation spatiale* (Spatial Reuse) [09, 03] et il sert de base au concept de la communication cellulaire.

La contrepartie de la réutilisation spatiale est que certaines paires de nœuds peuvent alors être hors de portée mutuelle. Ceci nécessite l'emploi d'un routage interne par des nœuds intermédiaires. La gestion de ce routage consiste à établir une sorte d'architecture globale où l'on doit tenir compte de la mobilité des nœuds et de la versatilité du médium physique. Pour parvenir à des protocoles efficaces d'accès, de communication, d'allocation de ressources et de routage, ces nouvelles approches doivent faire appel à de l'algorithmique de pointe.

Le problème du routage est loin d'être évident dans les réseaux mobiles ad hoc, où il est difficile de localiser la destination à un instant donné. La conception des stratégies de routage doit tenir compte de tous les facteurs et limitations physiques imposés par l'environnement afin que les protocoles de routage résultant ne dégradent pas les performances du système.

Ce chapitre présente les réseaux mobiles ad hoc et leurs caractéristiques. Il met en relief la problématique clé de ces réseaux : le routage.

2. Les réseaux mobiles Ad Hoc

Les évolutions technologiques récentes dans les domaines de la communication sans fil et des unités de calcules portables (LAPTOP, NETBOOK, etc.) ainsi que les programmes et algorithmes sous jacents, rapprochent de plus en plus les horizons de « l'accès à l'information m'importe ou et n'importe quand ».

Ainsi, le concept des réseaux mobiles ad hoc essaie d'étendre les notions de la mobilité à toutes les composantes de l'environnement. Ici, contrairement aux réseaux basés sur la communication cellulaire, aucune administration centralisée n'est disponible, ce sont les hôtes mobiles eux-mêmes qui forment, d'une

Figure 1-1. Réseau Ad hoc

manière ad hoc, une infrastructure du réseau. Aucune supposition ou limitation n'est faite sur la taille du réseau ad hoc, le réseau peut contenir des centaines ou des milliers d'unités mobiles.

Les réseaux mobiles ad hoc sont idéals pour les applications caractérisées par une absence (ou la non-fiabilité) d'une infrastructure préexistante, tel que les applications militaires et les autres applications de tactique comme les opérations de secours (incendies, tremblement de terre..) et les missions d'exploration.

2.1. Définition

Un réseau mobile ad hoc, appelé généralement MANET (Mobile Ad hoc Network), consiste en une grande population, relativement dense, d'unités mobiles qui se déplacent dans un territoire quelconque et dont le seul moyen de communication est l'utilisation des interfaces sans fil, sans l'aide d'une infrastructure préexistante ou administration centralisée. Un réseau ad hoc peut être modélisée par un graphe $G_t = (V_t , E_t)$ où V_t représente l'ensemble des nœuds (i.e. les unités ou les hôtes mobiles) du réseau et E_t modélise l'ensemble les connexions qui existent entre ces nœuds (voir la figure 1.2). Si $e = (u, v)$ E_t, cela veut dire que les nœuds u et v sont en mesure de communiquer directement à l'instant t.

La topologie du réseau peut changer à tout moment (voir la figure 1.3). Elle est donc dynamique et imprévisible ce qui fait que la déconnexion des unités est très fréquente.

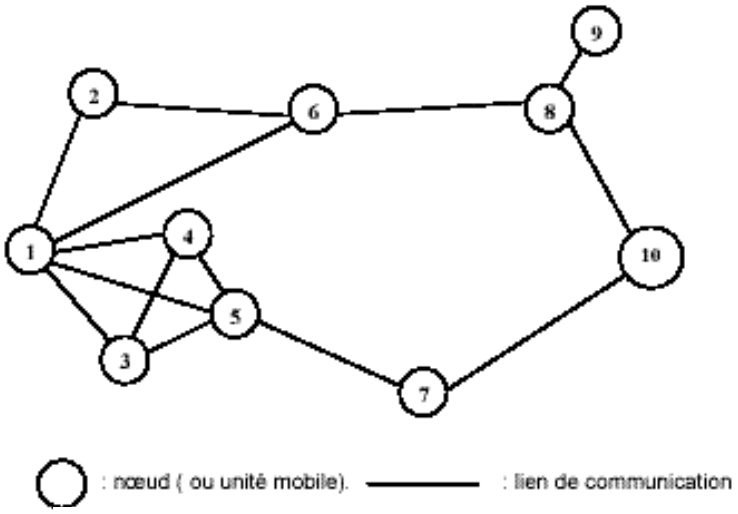

Figure 1.2 : La modélisation d'un réseau ad hoc.

Un exemple d'un réseau ad hoc : un groupe d'unités portables reliées par des cartes HIPERLAN (ou HIgh PERformance radio LAN). Les réseaux appelés GSM ne représentent pas les réseaux mobiles ad hoc, car la communication entre les unités passe obligatoirement par des stations de base du réseau filaire.

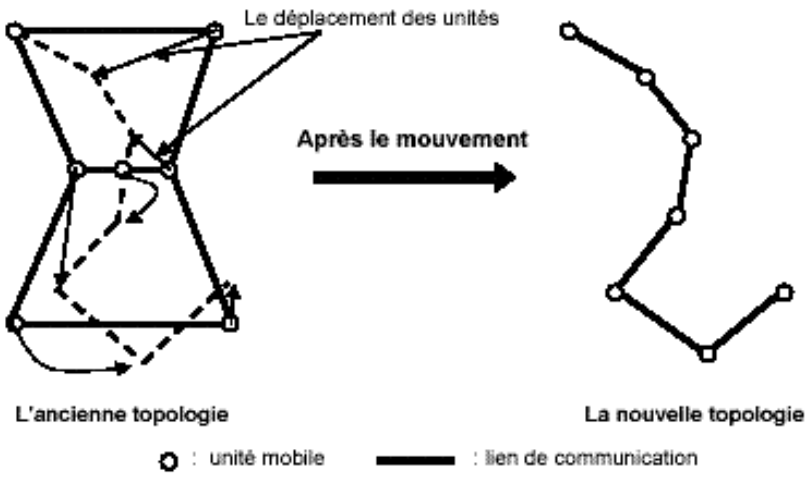

Figure 1.3 : Le changement de la topologie des réseaux ad hoc.

2.2. Les applications des réseaux mobiles ad hoc

Les applications ayant recours aux réseaux mobiles ad hoc couvrent un très large spectre, incluant les applications militaires et de tactique, les bases de données parallèles, l'enseignement à distance, les systèmes de fichiers répartis, la simulation distribuée interactive et plus simplement les applications de calcul distribué ou méta computing.

D'une façon générale, les réseaux mobiles ad hoc sont utilisés dans toute application où le déploiement d'une infrastructure réseau filaire est trop contraignant, soit à cause de la difficulté de la mise en place, soit parce que la durée d'installation du réseau ne justifie pas de câblage.

2.3. Les caractéristiques des réseaux ad hoc

Les réseaux mobiles ad hoc sont caractérisés par ce qui suit :

Une topologie dynamique : Les unités mobiles du réseau, se déplacent d'une façon libre et arbitraire. Par conséquent la topologie du réseau peut changer, à des instants imprévisibles, d'une manière rapide et aléatoire. Les liens de la topologie peuvent être unis ou bidirectionnels.

Une bande passante limitée : Une des caractéristiques primordiales des réseaux basés sur la communication sans fil est l'utilisation d'un médium de communication partagé. Ce partage fait que la bande passante réservée à un hôte soit modeste.

Des contraintes d'énergie : Les hôtes mobiles sont alimentés par des sources d'énergie autonomes comme les batteries ou les autres sources consommables. Le paramètre d'énergie doit être pris en considération dans tout contrôle fait par le système.

Une sécurité physique limitée : Les réseaux mobiles ad hoc sont plus touchés par le paramètre de sécurité, que les réseaux filaires classiques. Cela se justifie par les contraintes et limitations physiques qui font que le contrôle des données transférées doit être minimisé.

Une absence d'infrastructure : Les réseaux ad hoc se distinguent des autres réseaux mobiles par la propriété d'absence d'infrastructures préexistantes et de tout genre d'administration centralisée. Les hôtes mobiles sont responsables d'établir et de maintenir la connectivité du réseau d'une manière continue.

3. Le problème du routage dans les réseaux ad hoc

3.1. Définition

Généralement, le routage est une méthode d'acheminement des informations à la bonne destination à travers un réseau de connexion donné. Le problème du routage consiste, pour un réseau dont les arcs, les nœuds et les capacités sur les arcs sont fixées, à déterminer un acheminement optimal des paquets (de messages, de produits …etc.) à travers le réseau au sens d'un certain critère de performance. Le problème consiste à trouver l'investissement de moindre coût en capacités nominales et de réserves qui assure le routage du trafic nominal et garantit sa survie en cas de n'importe quelle panne d'arc ou de nœud.

Par exemple si on suppose que les coûts des liens sont identiques, le chemin indiqué dans la figure suivante est le chemin optimal reliant la station source et la station destination. Une bonne stratégie de routage utilise ce chemin dans le transfert des données entre les deux stations.

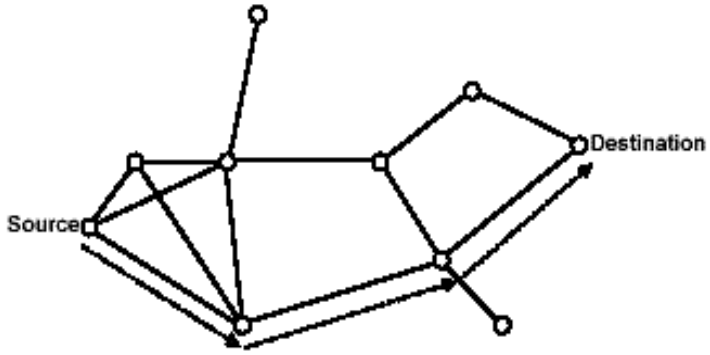

Figure 1.4 : Le chemin utilisé dan le routage entre la source et la destination.

3.2. La difficulté du routage dans les réseaux mobiles ad hoc

Nous avons vu que, l'architecture d'un réseau mobile ad hoc est caractérisée par une absence d'infrastructure fixe préexistante, à l'inverse des réseaux de télécommunication classiques. Un réseau ad hoc doit s'organiser automatiquement de façon à être déployable rapidement et pouvoir s'adapter aux conditions de propagation, au trafic et aux différents mouvements des unités mobiles.

Dans le but d'assurer la connectivité du réseau, malgré l'absence d'infrastructure fixe et la mobilité des stations, chaque nœud est susceptible d'être mis à contribution pour participer au routage et pour retransmettre les paquets d'un nœud qui n'est pas en mesure d'atteindre sa destination; tout nœud joue ainsi le rôle de station et de routeur.

Chaque nœud participe donc à un protocole de routage qui lui permet de découvrir les chemins existants, afin d'atteindre les autres nœuds du réseau. Le fait que la taille d'un réseau ad hoc peut être énorme, souligne que les techniques du routage utilisées doivent être complètement différentes des approches utilisées dans le routage classique. Le problème qui se pose dans le contexte des réseaux mobiles ad hoc est l'adaptation de la méthode d'acheminement utilisée avec le grand nombre d'unités existants dans un environnement caractérisé par de modestes capacités de calcul et de sauvegarde.

Dans la pratique, il est impossible qu'un hôte puisse garder les informations de routage concernant tous les autres nœuds, dans le cas où le réseau serait volumineux. Certains protocoles, comme le DSR et le AODV [80, 83, 19], utilisent la sauvegarde des données de routage concernant une destination donnée (dans le cas ou la source ne possède pas déjà de telles informations). Le problème ne se pose pas dans le cas de réseaux de petites tailles, car l'inondation (la diffusion pure) faite dans ces réseaux n'est pas coûteuse. Par contre dans un réseau volumineux, le manque de données de routage concernant les destinations peut impliquer une diffusion énorme dans le réseau, et cela si on considère seulement la phase de découverte de routes. Le trafic causé par la diffusion, dans ce cas, est rajouté au trafic déjà existant dans le réseau ce qui peut dégrader considérablement les performances de transmission du système caractérisé principalement par une faible bande passante.

Dans le cas où le nœud destination se trouve dans la portée de communication du nœud source le routage devient évident et aucun protocole de routage n'est initié. Malheureusement, ce cas est généralement rare dans les réseaux mobiles ad hoc. Une station source peut avoir besoin de transférer des

données à une autre station qui ne se trouve pas dans sa portée de communication. Par exemple, dans le réseau illustré par la figure 1.5 l'unité mobile W n'est pas dans la portée de communication de l'unité U (indiquée par le cercle d'origine U) et vice versa. Dans le cas où l'unité U voudrait transférer des paquets à W, elle doit utiliser les services de l'unité V dans l'envoi des paquets, puisque l'unité V contient dans sa portée de communication les unités U et W.

Dans la pratique, le problème du routage est plus compliqué à cause de la non uniformité de la transmission sans fil et de la possibilité du déplacement imprévisible de tous les nœuds concernés par le routage.

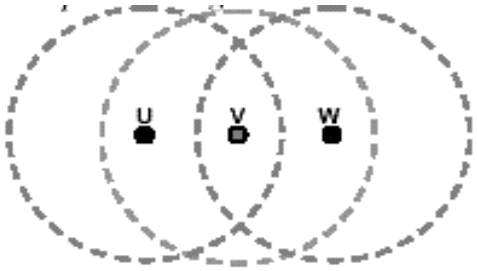

Figure 1.5: Un simple réseau ad hoc constitué de trois unités mobiles.

3.3. Notions de multihoping et d'inondation

3.3.1. La notion de "multihoping"

Les stratégies de routage utilisées dans les réseaux ad hoc sont caractérisées par le fait de pouvoir acheminer les paquets de données sans l'aide des stations de base utilisées dans la communication cellulaire.

Dans le modèle cellulaire, la communication entre deux nœuds est faite en utilisant les stations de base et le réseau filaire. Par conséquent, aucune unité mobile n'est utilisée comme routeur intermédiaire. Le modèle cellulaire est dit alors "single hop" (i.e. le nombre de routeurs mobiles intermédiaires est nul). La contrepartie de ce modèle est le modèle de communication sans infrastructure. Dans ce modèle plusieurs nœuds peuvent participer au routage c'est pour cela que l'environnement des réseaux ad hoc est dit "multihop" (i.e. le nombre de stations mobiles qui peuvent être utilisées comme routeurs intermédiaires peut dépasser le un).

3.3.2. L'inondation

L'inondation ou la diffusion pure, consiste à faire propager un paquet (de données ou de contrôle) dans le réseau entier. Un nœud qui initie l'inondation envoie le paquet à tous ses voisins directs. De même, si un nœud quelconque du réseau reçoit le paquet, il le rediffuse à tous ses voisins. Ce comportement se répète jusqu'à ce que le paquet atteigne tous les nœuds du réseau (voir la figure 1.6). Notons que les nœuds peuvent être amenés à appliquer - durant l'inondation - certains traitements de contrôle, dans le but d'éviter certains problèmes, tel que le bouclage et la duplication des messages. Le mécanisme d'inondation est utilisé généralement dans la première phase du routage plus exactement dans la procédure de découverte des routes, et cela dans le cas où le nœud source ne connaît pas la localisation exacte de la destination. Un paquet de requête de route est inondé par la source afin qu'il atteigne la station destination. Il faut noter que l'inondation est très coûteuse surtout dans le cas où le réseau est volumineux (latence, surcharge des

messages…etc.), c'est pour cela que les protocoles de routage essaient de minimiser au maximum la propagation des paquets inondés en rajoutant d'autres paramètres de diffusion.

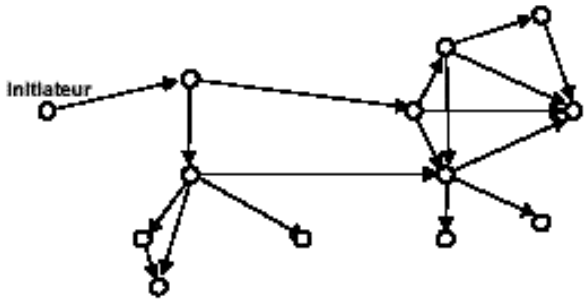

Figure 1.6 : Le mécanisme d'inondation.

4. Conclusion

Dans ce chapitre nous avons présenté le concept de réseau mobile ad hoc et le problème de routage dans cet environnement. Dans la pratique, les réseaux mobiles ad hoc connaissent aujourd'hui plusieurs applications telles que les applications militaires et les applications de secours et de façon générale, toutes les applications caractérisées par une absence d'infrastructure préexistante. Après avoir défini l'environnement mobile ad hoc et décrit ses principales applications et caractéristiques, nous avons présenté le problème d'acheminement des paquets dans les réseaux mobiles ad hoc, c'est à dire le problème du routage. Le routage est un service très important dans les environnements mobiles, surtout quand il n'y a pas d'infrastructure qui s'occupe de l'acheminement des données.

Le problème du routage est loin d'être évident dans les réseaux mobiles ad hoc, où l'environnement impose de nouvelles limitations par rapport aux environnements classiques. Les stratégies de routage doivent tenir compte des changements fréquents de la topologie, de la consommation de la bande passante qui est limitée, ainsi que d'autres facteurs.

Dans le chapitre suivant, nous allons présenter les notions de bases du concept de groupe et du service de gestion et de communication de groupe.

Chapitre 2

Service de groupe

1. Introduction

Un groupe est une collection de processus apparentés, considérés comme une entité logique unique, coopérant à la réalisation d'une tâche commune. Ce paradigme recouvre deux services de base, à savoir, le service de gestion de la composition du groupe (de l'anglais, *group membership*) ou simplement, service de gestion de groupe et le service de communication de groupe. Le *service de gestion de groupe* est chargé de fournir, aux processus membres, la composition actuelle du groupe. Ceci est un attribut principal de l'état du groupe (les autres attributs de l'état du groupe sont liés au service ou au calcul que le groupe est censé fournir). La composition du groupe évolue selon la volonté des processus de le joindre ou de le quitter, et également selon l'occurrence de défaillances de ses membres ou des canaux de communication. L'ensemble des processus qui constituent le groupe à un instant donné est appelé la vue courante du groupe. Le but du *service de communication* est de donner aux entités extérieures et intérieures au groupe des primitives de communication bien adaptées au paradigme de calcul de groupe. La communication se fait de manière individualisée (point à point) ou collective (*broadcast*) tout en suivant des restrictions d'ordre (FIFO, causal, total) et de fiabilité (terminaison, atomicité, ponctualité, etc.).

Du fait de l'importance du modèle de groupe dans la réalisation d'applications fiables, de nombreux efforts ont été réalisés durant cette dernière décennie pour comprendre la problématique liée à la mise en œuvre d'un tel paradigme dans le cadre des systèmes repartis classiques. Ces efforts ont abouti à l'obtention de résultats théoriques et au développement de nombreux protocoles. Les travaux réalisés à l'université de Cornell – ISIS [17], HORUS [96], ENSEMBLE [47] et SPINGLASS [20] – en sont les exemples les plus significatifs.

Il est important de remarquer que l'établissement de spécifications appropriées, ainsi que la conception et la réalisation d'un tel service de groupe sont des loin d'être considérées comme triviales [01, 25]. Ces difficultés sont dues principalement aux résultats d'impossibilité concernant les problèmes d'accord auxquels on se trouve confronté lors de la mise en œuvre des fonctionnalités du paradigme dans des environnements asynchrones. Les premiers systèmes conçus: ISIS [17], HORUS [96], TRANSIS [38], TOTEM [73], Relacs [13], Consul [75], NewTop [41] ont fait preuve de spécifications incomplètes et ont fait l'objet de mises en œuvre insatisfaisantes (dans lesquelles par exemple le groupe peut disparaître) [01]. Les nouvelles générations: ENSEMBLE [47], Relacs [14], Phoenix [69], Bast [45] essayent d'apporter des spécifications plus rigoureuses avec des caractérisations plus précises du comportement du système dans des périodes d'exécution considérées "instables" (avec peu de synchronisation). On remarque toutefois une diversité importante dans les terminologies et les propriétés utilisées dans la définition de tels systèmes. L'étude réalisée par [25] synthétise de manière très rigoureuse les diverses sémantiques associées à ces environnements, fournissant un cadre unique pour leur classification, appréciation et comparaison.

Plusieurs applications principalement des applications de coopérations : calcul distribué, télé enseignement etc. sont basées sur le concept de groupe et de communication de groupe. Ce chapitre introduit ces deux concepts et leurs caractéristiques

Dans ce chapitre, nous nous concentrons sur la spécification de primitives d'accord liées à la sémantique du paradigme de groupe. Nous présentons par la suite le modèle de groupe avec la spécification des services essentiels à la réalisation d'une telle abstraction.

1.1. Le modèle de groupe

De manière informelle, un *groupe* est un ensemble de processus coopérant pour effectuer une tâche commune (ex., des copies d'un serveur répliqué, des participants à une transaction distribuée, ou des utilisateurs d'une application coopérative). Le groupe est *dynamique* si sa composition peut changer (suite à l'intégration ou l'exclusion de processus) selon les impératifs de l'application et les conditions d'exécution (existence de défaillances, retards, etc.). D'autre part, si sa composition ne peut pas changer, il est dit *statique*.

Dans les groupes *ouverts*, les entités externes au groupe sont habilitées à lui envoyer des messages sans se soucier de sa composition. Bien évidemment, ces entités ne peuvent pas recevoir les messages échangés au sein du groupe. Si la communication se restreint seulement aux processus appartenant au groupe, on parle alors de groupes *fermés*.

Le groupe : Le groupe possède un *état*: cela signifie que si tous les processus le représentant à un instant donné, sont défaillants, alors le groupe cesse d'exister et on dit qu'il défaille [88]. A tout instant, la composition courante du groupe, perçue par un de ses membres, constitue sa *vue courante*. Une vue *v* est composée de deux champs: *v.id* (représentant son identité) et *v.membres* (représentant sa composition); sachant que |*v*| représente sa cardinalité. En cas d'existence de plusieurs groupes, chacun peut être géré de manière autonome

Les processus : L'ensemble $\Pi = \{p1 ; p2; ....\}$ représente la totalité des processus pouvant joindre le groupe. A tout instant, seul un sous-ensemble de Π participe au calcul du groupe. Les processus peuvent invoquer deux opérations, désignées par joindre et quitter, représentant l'entrée et la sortie du processus du groupe. On suppose qu'un processus n'est membre du groupe qu'une seule fois, donc, l'action de le quitter est définitive. En pratique, un processus qui quitte un groupe (volontairement ou pas) peut le réintégrer sous une nouvelle identité. Après que le processus *pi* ait invoqué l'opération joindre, il sait qu'il appartient au groupe lorsque un message installe(*v*) lui est délivré, où *v* est une vue telle que $i \in v.membres$. Quand *pi* exécute le code correspondant à la livraison du message *installe*(*v*), nous disons que "*pi* installe la vue *v*".

1.2. Caractéristiques de groupe

Les groupes peuvent être classifiés selon plusieurs critères tels que, leur composition, la discipline de communication, le contrôle de concurrence...etc. [54]. Ainsi, nous pouvons distinguer :

- groupe *Statique* vs. *Dynamique* : Un groupe statique se caractérise par une population qui n'évolue pas dans le temps. Un groupe dynamique se caractérise par une population qui peut changer au cours du temps, c.-à-d. que les règles qui définissent l'adhésion au groupe peuvent évoluer dans le temps donc on peut modifier la définition de la structure du groupe;
- groupe *défini* vs. *Indéfini* : Un groupe défini est un groupe au sein duquel il est possible de déterminer l'ensemble des entités qui le compose. Un groupe Indéfini est un groupe au sein duquel il n'est pas possible de déterminer de façon non ambiguë l'ensemble des membres qui le compose. Les entités qui sont connues par toute ou une partie de la population sont appelées Membres Connus indéfini;
- groupe *complètement connu* vs. *partiellement connu :* Un groupe peut être constitué d'une population complètement connue, i.e. toutes les entités sont capables de connaître le nom ou l'adresse individuelle de chacun des membres Population Partiellement connue : population ou seule une partie des entités est capable de connaître le nom où l'adresse individuelle de chacun des membres du groupe;

- groupe mixte vs. envoi seulement vs. reçoit seulement : dans un groupe envoi seulement les membres envoient seulement des messages sans recevoir aucune réponse des récepteurs. Dans un groupe reçoit seulement les membres reçoivent seulement les messages sans avoir l'autorisation d'envoyer des messages. Finalement, dans un groupe mixte les membres peuvent envoyer et recevoir dans un groupe envoi seulement les messages;
- groupe *centralisé* vs. *décentralisé* : dans un groupe centralisé un seul membre est autorisé à envoyer les messages et les autres membres sont autorisés à les recevoir. La désignation de l'émetteur peut être dynamique, et changer d'un membre à un autre. Contrairement à un groupe décentralisé où tous les membres peuvent envoyer ou recevoir les messages ;
- groupe *contrôlé* vs. *Incontrôlé* : dans un groupe contrôlé seulement les membres autorisés, à un instant donné, peuvent transmettre des informations. Contrairement à cela, dans un groupe incontrôlé tous les membres peuvent transmettre des messages simultanément ;

Les membres d'un groupe peuvent avoir différents rôles, associés à des privilèges appropriés, responsabilités ou priorités. Le rôle le plus important est celui du superviseur du groupe, contrôlant la composition du groupe et les conditions d'intégrités.

Une des raisons de création d'un groupe est de pouvoir adresser les membres avec une seule adresse de groupe, et les considérer comme étant une seule entité virtuelle. Cependant, il est considéré qu'être membre d'un groupe ne signifie pas nécessairement être accessible par l'adresse du groupe [70]. Par contre, un membre doit exprimer (explicitement ou implicitement) son intention de participer à la communication. Ainsi, il devient un membre enregistré, les membres enregistrés qui participent dans l'échange de données constituent le groupe actif, ce qui nous amène à un autre critère de classification :

- groupe *ouvert* vs. *fermé* : dans un groupe ouvert un membre non enregistré peut prendre part dans le groupe actif. Par contre, dans un groupe fermé seuls les membres enregistrés peuvent le faire.

Un autre concept important est l'association de groupe, qui est l'association établie entre les membres du groupe actif pour des raisons de transfert de données. Plusieurs associations de groupe peuvent coexister dans un même groupe. Il existe trois importantes associations de groupe de base [54, 70, 04]

- *one-way* (ou simplexe) : c'est une association dont les participants sont clairement partitionnés en deux parties, ceux qui reçoivent et ceux qui envoient les messages.
- *two-way* (ou duplex) : c'est une association dont tous les participants ont la possibilité d'envoyer ou recevoir, mais ils sont partitionnés en deux parties de tel façon qu'un participant d'une partie ne peut échanger des messages qu'avec un participant de la deuxièmes partie
- *N-way* : c'est une association qui comprend les deux associations précédentes

La figure 2.1 montre un exemple des trois types d'associations de groupe de bases. Par leur

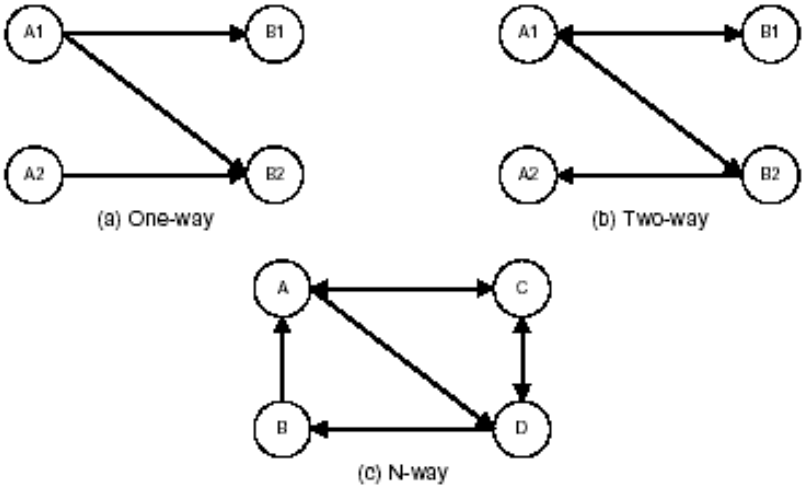

Figure 2.1. Exemple des trois associations de groupe de base

combinaison, toutes autre association peut être obtenue dans un groupe envoi seulement

1.3. Les conditions d'intégrités

Les conditions d'intégrités peuvent être définies comme étant un ensemble de lois concernant la composition et la topologie du groupe [04]. Ces lois décident si l'association du groupe doit être établie, maintenue, suspendue ou détruite.

- Le nombre minimum de membres : représente le nombre minimum de membre que doit contenir un groupe actif ;

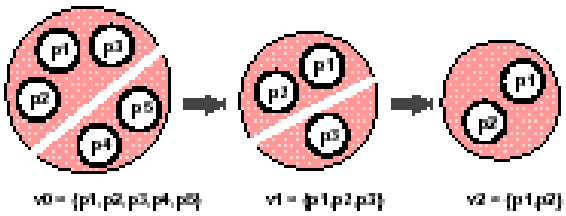

Figure. 2.2 – *Le service de gestion du groupe à composante primaire*

- Le nombre maximum de membres : représente le nombre maximum de membre que peut contenir un groupe actif ;
- Quorum : représente le pourcentage des membres que doit contenir un groupe actif;
- Les membres mandataires : représente la liste des membres qui doivent être obligatoirement présents ;
- L'atomicité : représente la composition exacte du groupe actif ;

2. Le service de gestion de groupe

Informellement, le *problème de gestion de groupe* consiste à fournir aux processus, formant le groupe à un instant donné, une vue, tenant compte de l'histoire antérieure du groupe, qui soit le reflet des opérations joindre et quitter déjà exécutées, et des défaillances des processus et des canaux de communication qui se sont produites.

Le problème de gestion de groupe a été introduit et résolu pour la première fois par Cristian [27] dans le contexte des systèmes distribués synchrones. Dans le domaine des systèmes asynchrones, le système ISIS [17] est le pionnier. Les difficultés existantes pour spécifier et résoudre ce problème sont directement liées au résultat d'impossibilité concernant la détection fiable des défaillances dans le modèle asynchrone [26]. Puisque des défaillances réelles combinées à d'éventuelles fausses suspicions peuvent mener à un fractionnement du groupe en sous-groupes, deux approches pour la gestion de la composition sont apparues: le *service de gestion à composante primaire* (*primary partition systems*) et le *service de gestion à composantes cloisonnées* (*partitionable systems*).

2.1. La gestion à composante primaire et à composantes cloisonnées

La stratégie introduite par ISIS [87] pour dépister les défaillances de processus a été d'éliminer du groupe tout processus soupçonné d'être défaillant. Cette approche, désignée sous le nom de "service de

gestion à composante primaire", assure qu'à tout instant le groupe est représenté par une vue unique. Cette vue primaire rassemble les processus appartenant à la composante majoritaire du groupe. Dans les composantes minoritaires, l'exécution de l'application est bloquée. Au fil du temps, ceci signifie que l'ensemble des vues représentant le groupe est totalement ordonnancée. La figure 2.2 illustre cette abstraction.

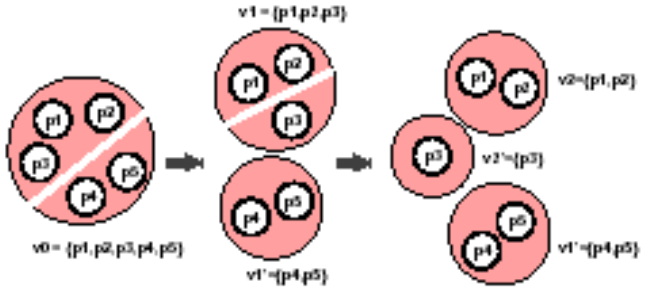

Figure. 2.3 – *Le service de gestion du groupe à composante cloisonnées*

D'un point de vue pratique, l'approche de gestion à une seule composante a été motivée par le contexte dans lequel se plaçait ISIS, celui des réseaux locaux, où il y a très peu de défaillances de canaux. Lorsque l'on considère un contexte où celles-ci sont plus fréquentes (ex., les réseaux à grande échelle, les réseaux de communication sans fil) on doit fournir des abstractions appropriées pour traiter le problème du fractionnement du groupe en de multiples sous-groupes (partitionnement du réseau en plusieurs composantes). Dans chacune de ces composantes, le service (ou au moins une partie de celui-ci) doit continuer à être assuré. Ceci est souhaitable pour certaines applications (ex., réservation de billets d'avion, transactions bancaires). Le but est donc de permettre une coexistence de différentes vues disjointes du groupe (des vues concurrentes) à un instant donné [38, 75, 13]. Ceci signifie que l'ensemble des vues est partiellement ordonnancé. Cette abstraction caractérise le service de gestion à composantes cloisonnées illustré par la figure 2.3. Les processus dans une vue ne peuvent pas communiquer avec des processus appartenant à d'autres vues. Dans chaque vue, les processus se comportent comme s'ils étaient les seuls à mettre en œuvre l'état actuel du groupe. Éventuellement, lorsque la communication est restaurée, le service de gestion exécute un mécanisme de *fusion de vues* afin de remettre le groupe dans un état cohérent (tenant compte des différents états des composantes fusionnées)

La *fusion de vues* est une procédure très difficile à mettre en œuvre. En principe, les processus peuvent enregistrer les messages pendant les périodes de partitionnement du groupe pour une retransmission ultérieure lors de la phase de fusionnement. Toutefois, une capacité de mémoire suffisante doit être prévue pour supporter les longues périodes de rupture de communication. En outre, même s'il y a de la mémoire disponible, le problème du partage de l'état est très dépendant de l'application et présente ses propres inconvénients: il engendre, en particulier, une longue période d'indisponibilité du service jusqu'à la réconciliation complète. Les difficultés rencontrées pour traiter ce problème sont tellement importantes qu'elles peuvent même compromettre l'intérêt de l'emploi du paradigme de groupe. La série de travaux réalisés à l'université de Bologne fournit d'importantes contributions à la compréhension et à la résolution des problèmes posés par la fusion de vues lorsque l'état de l'application est partagé [12, 14, 11].

Pour assurer un comportement cohérent de l'application durant l'occurrence d'un partitionnement (réel ou virtuel) du groupe, la plupart des services de gestion à composantes cloisonnées sont obligés de confiner les fonctionnalités de l'application dans une composante unique, la vue primaire [10]. Les autres vues, considérées comme secondaires, voient leur fonctionnalités restreintes. Par exemple, dans un système de base de données, la vue primaire est autorisée à effectuer des opérations de lecture/écriture,

tandis que dans les vues secondaires seule la lecture est acceptée. Avec une telle stratégie, on élimine ainsi la plupart des problèmes liés à la réconciliation de vues. Notons que dans ce cas, les systèmes à composantes cloisonnées ont une sémantique similaire à celles des systèmes à composante primaire.

Un résultat d'impossibilité les auteurs de [24] ont montré que même une version très faible (mais toutefois réaliste) du problème de la gestion à composante primaire du groupe n'a pas de solution dans le modèle FLP [43]. Ce problème était contraint par le résultat d'impossibilité dit de Fischer-Lynch-Paterson [43] qui stipule qu'il est impossible de résoudre le problème du consensus dans un système asynchrone sujet aux défaillances. Cette version est faible puisqu'elle n'impose pas de retenir dans le groupe les processus opérationnels. Elle autorise l'élimination des processus corrects de la vue du groupe (en suivant une approche similaire a celle employée dans ISIS). La seule réserve faite est la suivante: à partir de la vue initiale (reconnue a priori par l'ensemble des processus qui la forment), si un seul processus *p* demande à quitter le groupe, alors le service de gestion de la composition ne devrait pas exclure la possibilité que *p* soit en effet le seul processus retiré de la vue du groupe à un moment donné. Cette impossibilité est de la même nature que celle du consensus [43].

Ce résultat négatif est à l'origine des problèmes identifiés dans la spécification et l'exactitude du protocole donné par ISIS [87]. Ce protocole peut mener à un suicide collectif du groupe (suppression de tous les processus) avant même d'installer de nouvelles vues [01]. Les services de gestion à composantes cloisonnées échappent à cette impossibilité, car en tolérant la coexistence de multiples vues, ils admettent en effet une "discordance de vues". Toutefois, [01] constate également des problèmes concernant la spécification [39, 13] et la mise en œuvre de quelques uns de ces protocoles. Ils peuvent morceler le groupe en plusieurs composantes minoritaires, composées à la limite d'un seul processus [40, 73]. Ce qui remet en cause l'existence même du groupe. Ils permettent également des changements de vues capricieux – effectués sans que des motifs justifiés existent (l'occurrence de défaillances réelles et/ou d'opérations joindre ou quitter) [39]. Finalement, le danger de l'utilisation de tels protocoles est la perte potentielle de la composante primaire, représentant l'état de l'application.

Selon [01], la spécification d'un service de gestion de la composition du groupe "devrait être assez forte pour garantir que n'importe quel protocole la satisfaisant soit réellement utile, et elle devrait être assez faible pour être mise en œuvre dans un système où des défaillances sont possibles". D'une manière générale, les protocoles à composante primaire assurent la première condition, mais pas la seconde. Ils font objet de spécifications plus rigoureuses, et leur conception favorise la cohérence et la simplicité. Les protocoles à composantes cloisonnées satisfont le second critère, mais pas le premier. Ils favorisent le développement flexible, mais leurs protocoles gagnent en complexité, notamment lors de la phase de fusion. L'obtention de spécifications remplissant les critères établis par [01] est encore un domaine d'investigation [39, 14, 50, 56, 99].

2.2. Spécification de la gestion de la composition du groupe à composante primaire

Une spécification récente à ce problème satisfaisant les critères établis par [01] a été donnée par [99]; malheureusement, elle peut bloquer l'évolution du système dans le cas d'une unique défaillance de processus. Le système Phoenix [76] présente un protocole basé sur les détecteurs de défaillances, toutefois sans qu'une description précise du problème ne soit donné. [56] propose l'utilisation de l'abstraction d'*oracles*, dont la sémantique généralise celle des détecteurs de défaillances. Ces oracles, outre les défaillances, sont censés capturer des informations spécifiques à l'application (en ce qui concerne les besoins d'entrer et de sortir du groupe). Avec une telle approche, les auteurs proposent une spécification rigoureuse suivie d'un protocole résolvant le problème à condition que les informations fournies par les oracles soient de bonne qualité.

Le *groupe est défaillant* quand tous les processus qui le composent subissent une défaillance. Cette propriété est nécessaire quand le groupe contient un état. Elle vise à permettre aux processus de la vue

suivante de reconstruire un état cohérent du service implanté par le groupe. Ceci vient du fait que les processus appartenant à deux vues consécutives peuvent transmettre l'information d'état d'une vue à une autre.

2.3. Le service de gestion de la composition du groupe à composante primaire faible

À cause du résultat d'impossibilité mentionné à la section 2.1, il se trouve qu'il n'y a pas de solution au problème tel que nous le définissons dans la section précédente. Pour contourner cette impossibilité, On adopte l'approche utilisée avec succès pour résoudre d'autres problèmes d'accord: on renforce le modèle asynchrone en ajoutant des hypothèses supplémentaires de synchronie et/ou l'on affaiblit la définition du problème de façon à le résoudre dans le modèle asynchrone.

Politique best effort : pour être utilisé, un protocole résolvant le problème doit faire de son mieux pour que la vue courante reflète la composition réelle du groupe. Ceci signifie qu'il ne doit pas définir systématiquement de nouvelles vues à chaque fois qu'un événement non fondé (surtout une suspicion erronée) se produit. Cette politique du *best effort* doit être considérée comme une propriété essentielle pour tout protocole de gestion de la composition du groupe [25]. Cependant, elle n'a pas de définition générique précise. Pour chaque mise en œuvre d'un service, il convient de préciser quel niveau de qualité "peut être" offert.

3. Le service de communication dans le groupe

Le service de communication comprend les protocoles pour la diffusion de messages au sein d'un groupe (ou de plusieurs groupes). Les propriétés spécifiées pour ces protocoles regroupent aussi bien la fiabilité que l'ordre de livraison. Les propriétés de fiabilité assurent une vue cohérente de l'ensemble des messages délivrés par les processus et celles d'ordonnancement établissent un ordre commun (FIFO, causal [57], total) dans la livraison de celles-ci. La combinaison de ces propriétés engendre un grand nombre de primitives de communication (diffusion fiable, atomique, causale, causale atomique, etc.) [51, 35].

3.1. La diffusion fiable

Un service de *diffusion fiable* garantit une livraison atomique des messages aux membres du groupe (un message envoyé est reçu par tous les processus corrects ou par aucun d'entre eux) [51]. Dans de nombreuses applications, une sémantique forte du type "tout ou rien" est nécessaire lors de la diffusion d'information du groupe.

Les propriétés définissant la *diffusion fiable* sont les suivantes: Il n'y a aucune duplication de message. Seuls les messages émis sont délivrés. Un message envoyé par un processus correct finira par être délivré par tous les processus corrects. Tous les processus corrects finiront par délivrer le même ensemble de messages.

3.2. La diffusion atomique

Un service de *diffusion* assure que tous les processus dans le groupe délivreront le même ensemble de messages dans le même ordre [51]. C'est la primitive clé pour la mise en œuvre d'un serveur répliqué de manière active, car elle contribue à préserver le déterminisme des copies. La figure 2.4 illustre une telle abstraction.

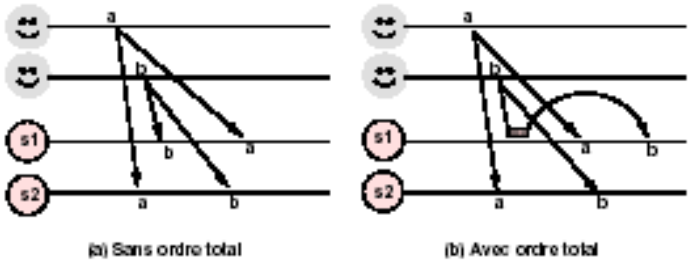

Figure.2.4 – *L'abstraction de la diffusion atomique*

La *diffusion atomique* rassemble toutes les propriétés de la *diffusion fiable* auxquelles vient s'ajouter une propriété de livraison de messages en ordre total. De ce fait, elle est aussi appelée *diffusion en ordre total.* Un important résultat théorique concernant la *diffusion atomique* et le problème du *consensus* qui a été présenté par Chandra et Toueg [31]. Ils ont démontré l'équivalence entre ces deux problèmes dans tout modèle permettant une résolution de la *diffusion fiable* [31]. Ceci signifie que tout algorithme permettant de résoudre le consensus résout la diffusion atomique et vice versa.

La *diffusion atomique* est une spécialisation d'une primitive plus générique appelé *diffusion sélective en ordre total* [35]. Cette dernière permet la diffusion des messages à un sous ensemble de processus, alors que la première considère comme destinataires du message tous les processus du système (Π), ou plus précisément, tous les processus appartenant au groupe.

4. Les problèmes d'accord et le consensus

De nombreux problèmes rencontrés dans la conception de systèmes fiables présentent la caractéristique commune suivante: les processus doivent se mettre d'accord sur un état déterminé du calcul. Ils définissent des services possédant au moins une propriété d'accord dans leurs spécifications. Dans la *diffusion atomique* [51], l'accord est fait sur un ensemble de messages et sur l'ordre de livraison de ces dernies; Dans la *gestion de la composition d'un groupe* [17], l'accord est fait sur l'ensemble de processus qui appartiennent au groupe à un moment donné.

4.1. Le consensus

Le problème du *consensus* est un dénominateur commun pour tous les problèmes d'accord car il factorise le besoin d'accord d'une manière générique. Chaque processus *pi* propose une valeur initiale *vi* et malgré l'existence de défaillances, les processus corrects doivent se mettre d'accord sur une valeur *v0* commune choisie parmi les valeurs *vi* proposées.

Le consensus uniforme La version uniforme tient compte de la décision prise par tous les processus, même ceux qui défaillent après avoir décidé, garantissant un comportement uniforme du service vis à vis de son utilisateur. Un "service d'accord uniforme" est donc caractérisé par les mêmes propriétés du service original, à l'exception de l'accord qui est étendu à l'ensemble des processus

Résolution du consensus Dans un environnement fiable (sans fautes) le *consensus* est trivialement résolu. Dans un système synchrone, il est également résolu, même en présence de fautes arbitraires, mais à condition qu'un nombre limité de processus se comporte malicieusement ($n \geq 3f + 1$) [65]. Le comportement uniforme des processus ne peut être assuré que lorsque des fautes bénignes sont considérées.

Dans un système asynchrone, le consensus n'a pas de solution déterministe [43]. Ce résultat fondamental entraîne l'impossibilité de résoudre d'autres problèmes où le besoin d'accord est récurrent: la diffusion atomique [31], l'élection [92], la validation atomique non bloquante [46], le service de gestion de la composition du groupe [24]. A l'origine de ces résultats on trouve la même cause: les incertitudes créés par le modèle asynchrone en ce qui concerne la détection de défaillances. Toutefois, dès qu'on ajoute quelque degré de synchronie, plusieurs de ces problèmes peuvent être résolus [34, 37, 42, 31].

4.2. La réduction entre problèmes d'accord

L'importance du consensus réside dans le fait que plusieurs autres problèmes d'accord sont réductibles à celui-ci. Donc, s'il existe une solution au consensus dans un modèle, on peut trouver des solutions à d'autres problèmes à partir d'une transformation d'algorithmes. Cette technique a été utilisée par Chandra et Toueg [31] pour résoudre le problème de la *diffusion atomique* [31] dans le modèle FLP [43] (qui stipule qu'il est impossible de résoudre le problème du consensus dans un système asynchrone sujet aux défaillances) avec des détecteurs non fiables.

5. Conclusion

De plus en plus d'applications et de services Internet sont conçus aujourd'hui autour des concepts de groupe et de communication de groupe. Dans ce chapitre, nous avons tout d'abord présenté une taxonomie correspondant à ces concepts, incluant différents critères de classification de groupes et des conditions d'intégrité. Nous avons ensuite analysé les propriétés et les exigences des différentes applications qui utilisent les communications de groupe (travail collaboratif assisté par ordinateur, téléconférences en temps réel, systèmes orientés groupe tolérants aux fautes)

Le service de groupe traite les problèmes liés à la composition et l'évolution du groupe. Nous avons présenté différentes caractéristiques que ce service doit offrir, liées au contrôle de la composition du groupe, la fiabilité des transmissions ou l'ordonnancement des paquets.

Dans le chapitre suivant, nous mettons l'accent sur l'impact de la mobilité sur le service de groupe et plus précisément, la mobilité des réseaux ad hoc.

Chapitre 3

Gestion de groupe dans les réseaux mobiles ad hoc

Ce chapitre présente les principales approches qui traitent du problème de service de groupe dans mobiles ad hoc

1. Introduction

Le service de gestion de groupe est traditionnellement étudié dans les systèmes distribués exécutés sur des réseaux fiables [89], généralement des réseaux fixes, les réseaux filaires là où les ruptures de liens et le partitionnement des réseaux sont rares. Dans le projet Transis [39, 96] les auteurs ont traité le problème de service de gestion de groupe et de communication de groupe dans des environnements où le réseau lui même peut être divisé, à cause des défaillances des nœuds et des liens de communications. Cependant, dans le contexte des environnements mobiles le service de gestion de groupe est affecté non seulement par l'état des nœuds et liens de communication, mais aussi par la position des nœuds mobiles.

Dans la littérature, il existe trois approches principales qui traitent le problème du service de groupe dans les environnements mobiles ad hoc : l'approche de Prakash et Baldoni [79], l'approche de [86] et l'approche [62]

Le reste du chapitre sera consacré à la présentation de ces approches.

2. Approche de Prakash et Baldoni [79] : Architecture pour la communication de groupe dans les réseaux ad hoc

2.1. Principe

Dans cette approche, les auteurs proposent d'organiser le service de gestion de groupe sous forme d'une pile de protocoles (c. f, Figure 3.1). Dans cette architecture, le concept de gestion de groupe repose sur une couche de proximité dont le rôle est de contrôler le changement de configuration du réseau en s'appuyant sur les primitives de la sous couche MAC de la couche de base du réseau mobile. Dans ce qui suit nous présentons le rôle détaillé de chacune de ces couches

Étant donné un nœud p et un paramètre distance *D*, la couche de proximité détermine tous les nœuds qui sont au voisinage du nœud p à une distance inférieure à *D.* Cette information est délivrée à intervalle constant, à la couche de gestion de groupe qui peut utiliser ces informations pour déterminer la

composition du groupe. La composition du groupe est déterminée seulement si c'est nécessaire et non à intervalle constant.

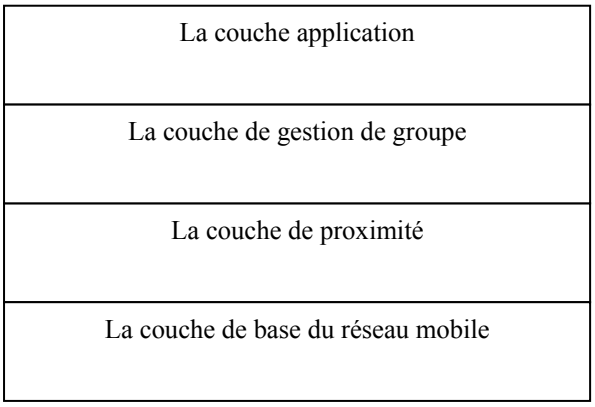

Figure 3.1. La pile protocolaire pour la communication de groupe mobile

2.2. La couche de proximité

La couche de proximité consiste en un protocole qui utilise les primitives de la couche MAC pour trouver tous les nœuds qui sont au voisinage de l'hôte mobile avec une certaine distance *d*.

La couche MAC fournit une communication point à point et les beacons. Chaque mobile émet un beacon régulièrement à chaque *t* unité de temps. La valeur de *t* dépend de : (i) la borne supérieure du nombre de nœuds qui peuvent être présents dans le domaine d'interférence d'un nœud ; (ii) le protocole de la couche MAC utilisée pour envoyer les beacons. Un beacon possède une portée limitée et sert comme un message de localisation « je suis là ». Les nœuds voisins qui sont dans la portée du beacon peuvent détecter la présence du nœud mobile même quand cette information n'est pas commandée par les couches supérieures.

Un réseau sans fils est connecté si :

Pour chaque nœud x il existe un nœud y telle que la distance (x , y) $\leq$ d , d est la portée des beacons ;

La distance (p , q) $\leq$ d $\Rightarrow$ connecté (p , q) ;

Quelques soit p , q , r , si connecté (p , q) et connecté (q , r) alors connecté (p , r) ;

Si pour chaque paire de nœuds p , q de l'ensemble des nœuds du réseau sont connecté (p , q) alors le réseau est dit connecté

Le test de D-proximité

Le protocole de la couche de proximité est exécuté chaque t' unités de temps, avec $t' > t$. Les communications de la couche de proximité sont synchrones et le délai de communication d'un message est délimité par t_p. Le but est de trouver tous les nœuds qui sont à une distance D d'un nœud donné. Durant le test de *D*-proximité les distances entre les nœuds peuvent changer, mais le graphe de connexion ne change pas avec une grand probabilité, ceci est due à la faible distance que peut couvrir un nœud durant le temps infiniment petit du test de D-proximité. Chaque instance de ce test peut être identifiée par une manière unique par le tuple : (*initiateur-id, timestamp, location stamp, D)* où *timestamp* et *location stamp*

représentent respectivement la valeur de l'horloge et la position du nœud initiateur. Si un nœud p initie le test de D-proximité alors initialement l'ensemble D-proximité contient uniquement le nœud p.

Si $D \leq d$ (la portée du nœud) le protocole de la couche de proximité envoie un message d'interrogation au niveau du nœud p et attend une réponse pendant $2t_p + t_r$ unités de temps qui correspond au délai de la couche de proximité, avec t_r est le temps maximale requit par un nœud pour répondre au message du test de *D*-proximité. Tous les nœuds qui sont à une distance inférieure à d reçoivent le message d'interrogation et envoient leurs positions. Après l'écoulement des $2t_p + t_r$ aucune réponse n'est acceptée. Pour chaque réponse, le nœud p calcule sa distance du nœud source de la réponse. Si cette distance n'est pas plus grande de *D* alors ce nœud est ajouté à l'ensemble *D*-proximité de p. Alternativement, le nœud p peut seulement écouter les beacons pendant t unités de temps. Tous les nœuds qui sont à une distance inférieure à d envoient leurs beacons durant cette période et l'ensemble *D*-proximité de p est ainsi construit sans avoir recours à des messages supplémentaires.

Si $D > d$, le nœud p ne peut pas atteindre tous les nœuds dans sont *D*-proximité directement avec une seule passe de messages, pour cela l'algorithme multi-passes suivant est utilisé

Initialement, tous les nœuds ne sont pas marqués, leurs pointeurs vers leur père sont NULL et l'ensemble de leurs fils est vide. Le nœud *p*, l'initiateur, envoie un message de propagation à tous les nœuds se trouvant à une distance inférieure à d. Ce message contient le 4-tuplé identifiant le test de *D*-proximité.

Lorsqu'un nœud reçoit son premier message de propagation d'un test de *D*-proximité, le nœud est marqué et dirige son pointeur père vers l'émetteur du message. Ce nœud propage ce message en lui ajoutant son identité et l'identité de son père. Si plusieurs messages d'inondations d'un même message arrivent à nœud non marqué en même temps, l'un d'entre eux est arbitrairement choisi et les autres sont ignorés.

Si un nœud déjà marqué reçoit un message contenant sa propre identité dans le champ père, il ajoute l'identité de l'émetteur du message à l'ensemble de ses fils. Une fois tous les fils déterminés, et si cet ensemble est vide le nœud réalise qu'il est une feuille et initialise la convergence vers l'initiateur.

Durant cette convergence, si un nœud feuille détermine qu'il est à une distance inférieure à *D* du nœud initiateur p il envoie un singleton contenant son identité a son père, sinon il envoie un ensemble vide a son père. Le calcul de distance est possible grâce aux informations sur la position de p transportés avec le message de propagation.

Une fois qu'un nœud q reçoit le message de convergence de l'ensemble de tous ses fils, il détermine sa propre distance du nœud initiateur p, si cette distance est plus grande que *D*, q envoie l'union de tous les ensembles reçus de ces fils, sinon il ajoute son propre identité à cette union.

Une fois que p l'initiateur reçoit le message de convergence de la part de tous ses fils, l'union de tous les ensembles reçus avec les messages avec lui même constitue l'ensemble des nœuds de son *D*-proximité

2.3. La couche de gestion de groupe

Due à la mobilité des nœuds dans les réseaux ad hoc, l'ensemble des nœuds qui se trouvent à la portée radio d'un nœud donné change avec le temps. Par conséquent l'ensemble des nœuds à vérifier est dynamique. De plus la communication entre deux participants dans la communication de groupe est différente de la communication point à point de la couche de proximité. Au surplus, les messages ont un délai fini mais imprévisible. Ceci rend difficile la tache de construction du groupe. Cette dernière se fait de manière asynchrone comme suit :

La formation du groupe initiée par un nœud p retourne des informations sur tous les nœuds qui sont à une distance inférieure à D du nœud initiateur. Ceci est diffèrent du test *D*-proximité car chaque nœud se trouvant dans la proximité peut ne pas appartenir au groupe.

Le protocole de construction de groupe est basé sur le protocole proposé dans [29]. On suppose que le nœud initiateur p connaît à priori l'ensemble de nœuds S, dans le D-proximité de p, qui contient le groupe g que p tente de former. Cette information est le résultat de l'exécution du protocole de la couche de proximité. Le protocole de construction de groupe procède comme suit :

Etape 1 : Un nœud p qui tente de rejoindre le groupe g, multicast un message REQUETE de position, aux nœuds de l'ensemble S, et attend la réponse. Ce message peut nécessiter le passage par plusieurs nœuds pour atteindre les destinataires. La latence de S_0 peut être non déterministe et significativement plus grande que t_p (le délai point à point de la couche de proximité), D'où l'affectation des communications asynchrones à la couche de gestion de groupe ;

Etape 2 : Un nœud $q \in S$ recevant le message de p procède comme suit :

Si q se trouve à une distance près définie D de p (égale au domaine d'interférence dans le contexte de l'allocation de canaux sans fils) , le nœud q envoie un message ACK à p avec les canaux utilisé à cet instant pour la communication ($busy_q$), le nœud q tente aussi de rejoindre le groupe g.

Si q ne se trouve pas à cette distance de p, q envoie un message NACK à p.

Etape 3 : En recevant les messages ACK ou NACK de chaque nœud de l'ensemble S, p envoie le message JOIN(g) a tous les nœuds qui ont répondu avec le massage ACK et les ajoute au groupe g, en recevant le message un nœud confirme son appartenance au groupe g, l'union de tous les ensembles des (*busy*) des membres de g dénote l'ensemble des canaux utilisés dans le domaine d'interférence, le nœud p de S_0 utilise un canal qui n'appartient pas a aucun ensemble (*busy*).

Etape 4 : Durant l'exécution des trois premières étapes de ce protocole, le protocole de la couche de proximité peut retourner un ensemble $S' \neq S$ dans la D-proximité de p, dans cette situation :

Si un nœud $r \in S' - S$, le nœud p envoie REQUETE à r et attend ça réponse.

Si un nœud $r \in S - S'$, le message ACK reçu de r est ignoré et le message JOIN(g) n'est pas envoyé à r , si ce message est envoyé avant de connaître S', seule l'ensemble ($busy_r$) est ignoré, mais le nœud reste dans le groupe g.

Logiquement, l'étape 1 de cet algorithme reste exécutée jusqu'à la fin de l'étape 3, l'algorithme d'allocation de canaux se termine à cause du nombre fini de nœuds dans le réseau.

Si le message ACK est immédiatement envoyé après la réception du message REQUETE, deux nœuds voisins peuvent simultanément entre en section critique ou peuvent utiliser le même canal ce qui cause une interférence, un algorithme d'exclusion mutuelle est nécessaire.

3. Approche de Roman et Al. [86]

3.1. Définition du problème

La notion de groupe est utilisée par les applications distribuées pour palier à différents problèmes. Parmi les applications qui utilisent la notion de groupe certaines comme les applications de gestions de transactions requièrent un certain niveau de consistance dans la structure globale des données. Dans ce contexte, le problème de gestion du groupe est de trouver parmi les membres du groupe, ceux qui consentent à coopérer pour accomplir une tache spécifiée d'une manière fiable. Si on suppose que les nœuds et les liens sont fiables, i.e., il existe des mécanismes pour recouvrir des défaillances transitoires, la seule menace sur la consistance de la vue du groupe vient de la mobilité des hôtes qui entrent et sortent du domaine de couverture des uns des autres. La mobilité des hôtes est aléatoire et continue avec une vitesse maximale connue. Les hôtes voulant volontairement s'éteindre doivent signaler leur intention au préalable.

Le réseau mobile ad hoc est modélisé avec un graphe $C_0 = G (V, E_0)$,où V est l'ensemble des nœuds mobiles et E_0 est l'ensemble de liens de communication bidirectionnels entre ces nœuds. Le graphe C_0 change avec le temps, la présence d'une arrête (u, *v*) signifie que *v* est à la portée de transmission de *u* et vice versa. Ce graphe représente la connexion physique du réseau, toute tentative de maintenir une vue exacte de ce graphe est infaisable. Pour cela, la notion de graphe de connexion logique est introduite. Ce graphe $C = G$ (V, E) est un sous graphe de C_0. Les deux graphes gèrent le même ensemble de nœuds mais le graphe logique ne contient pas toutes les arrêtes.

Un groupe *G* est défini comme le sous graphe connexe qui a la plus grande taille, du graphe logique *C*. un nœud *u* est toujours membre d'un groupe, ce groupe est représenté par *G(u)*. La politique de gestion de groupe ajoute une arrête dans le graphe logique lorsque celle-ci apparaît dans le graphe physique et satisfait certaines propriétés, et l'élimine si elle disparaît du graphe physique ou elle ne satisfait plus certaines propriétés.

Le problème du maintien du groupe est défini comme étant le besoin de chaque nœud du graphe logique de connaitre les membres de son groupe et que cette information soit consistante pour l'ensemble du groupe tout le temps. Pour réaliser cette politique de gestion de groupe, il est supposé que la vitesse de déplacement des nœuds ne dépasse pas une vitesse maximale *Vmax* et que les protocoles de routage existent et ont des délais limités : td. Sous ces suppositions, le service de gestion de groupe possède deux principaux buts : (1) aucun message entre deux membres d'un groupe n'est perdu. (2) les messages sont envoyés et reçus dans la même configuration. Le premier but est assuré par l'utilisation de la politique d'admission basé sur la position des nœuds. Le deuxième est assuré par la création d'une barrière de synchronisation de changement de configurations.

3.2. Stratégie de la solution

Le but du service de gestion de groupe étant de masquer les problèmes de configurations du groupe aux applications utilisant le groupe et aussi d'assurer que ces changements de configurations n'affectent aucun traitement en cours. Pour assurer cela, le système doit deviner comment les nœuds sont organisés dans les groupes. La solution est d'utiliser le protocole de découverte de groupe pour déterminer qui est aux environs. Le protocole de reconfiguration pour renforcer l'atomicité des changements de configuration qui comprends la fusion de groupes qui sont en contactes et la séparation de groupes qui sont menacés par la possibilité d'une déconnexion. Les concepts clés de ce protocole est la distance de sécurité entre les nœuds ou les groupes qui détermine si un nœud a ou non le temps d'exécuter le changement de configuration avant que la déconnexion ait lieu

3.2.1. Distance de sécurité

Soit deux nœuds de même portée de transmission égale à *R*. La distance entre eux est dite de sécurité si elle est plus grand du seuil *r (v, t, t')*, défini comme la distance maximale que peut parcourir un nœud dans le temps t que prend l'accomplissement d'une communication avec succès. En suppose ici que les deux nœuds se déplacent aléatoirement avec une vitesse ne dépassant pas *v* et le temps maximal d'un changement de configuration atomique est *t'*.

La figure 3.2 montre que les nœuds *a* et *b* ne peuvent plus continuer à exécuter une tâche commune s'ils veulent garantir la délivrance des messages car ils peuvent à tout moment sortir de la portée de transmission de l'un et de l'autre. La solution consiste à n'accepter que les nœuds *a* et *b* ne soient dans le même groupe que s'ils sont assez proche l'un de l'autre, i.e., à une distance de l'un de l'autre égale à

$$r = R - 2v * (t + t').$$

La notion de la distance de sécurité est utilisée pour déterminer si deux groupes peuvent fusionner. Elle est aussi utilisée pour déterminer quand ils doivent se séparer pour maintenir les exigences du service de gestion de groupe. Pour savoir si deux groupes sont à une distance de sécurité ou non le leader du groupe maintient l'ensemble des positions des nœuds de son groupe, tous les membres du groupe communiquent

leurs positions régulièrement au leader qui vérifie constamment la distance de sécurité et la présence de nouveaux nœuds dans la région.

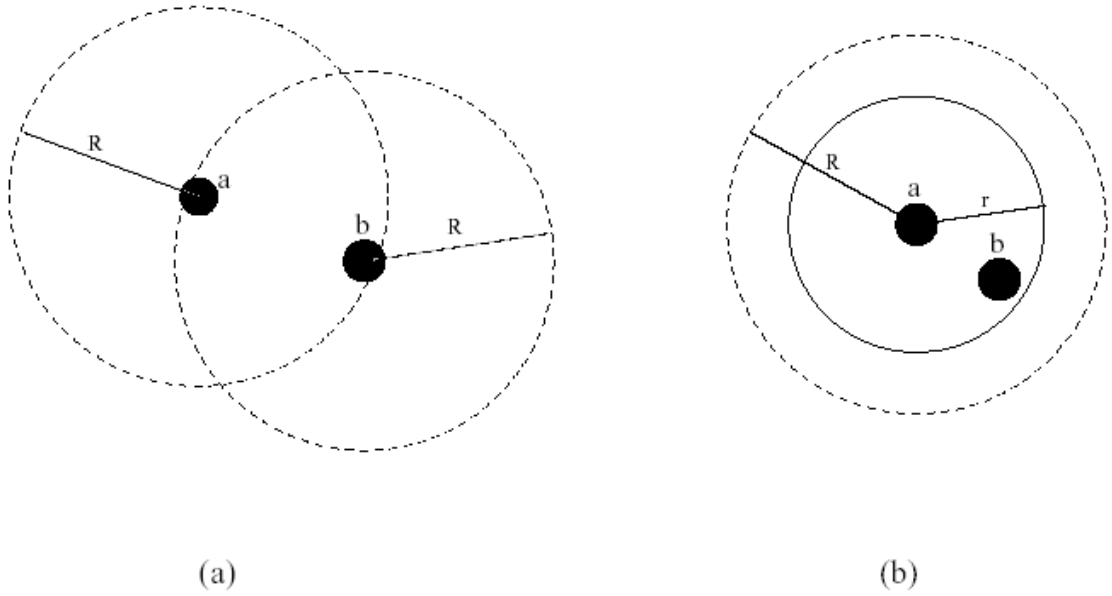

Figure 3.2 Exemple de distance de sécurité

3.3. Le protocole de découverte de groupe

Pour qu'un nœud mobile puisse rejoindre un groupe ou pour qu'un groupe puisse fusionner avec un autre, il est nécessaire de déterminer qui est dans son voisinage, Le protocole de découverte de groupe assure cette fonction.

Dans ce protocole, les nœuds utilisent la notion de distance de sécurité comme critère pour déterminer qui est assez proche pour être candidat a la fusion et reporte cette candidature de fusion à leur leader respectif. Ce mécanisme nécessite que chaque nœud broadcast régulièrement un message de découverte contenant sa position *(x, y)* et l'identifiant de son groupe *(gid)*, si deux groupes sont assez proches, plusieurs membres d'un groupe peuvent recevoir le message de découverte de nœuds de l'autre groupe. Si l'émetteur v n'appartient pas au même groupe que u avec une distance de sécurité, alors il informe son leader de groupe qui l'utilise pour la procédure de fusion.

Ce protocole permet au leader d'un groupe de maintenir une liste des groupes voisins qui sont assez proche pour être fusionnés.

3.4. Le protocole de reconfiguration

Le protocole de reconfiguration fusionne les groupes qui sont en contact et sépare ceux qui ne peuvent plus être ensemble plus longtemps. La fusion commence par la négociation entre les leaders des groupes

en contact. Une fois l'accord est établit tous les nœuds concernés reçoivent l'information des changements effectués. La figure 3.3 représente un exemple de fusion de deux groupes G1 et G2.

La séparation de groupe est déclenchée quand le leader constate que deux sous-groupes ne sont plus dans la distance de sécurité, il envoie immédiatement un message de séparation à tous les membres du groupe. Ce message contient les nouvelles informations des groupes résultants, la figure 3.4 représente un exemple de séparation d'un groupe g en deux groupes Z1 et Z2.

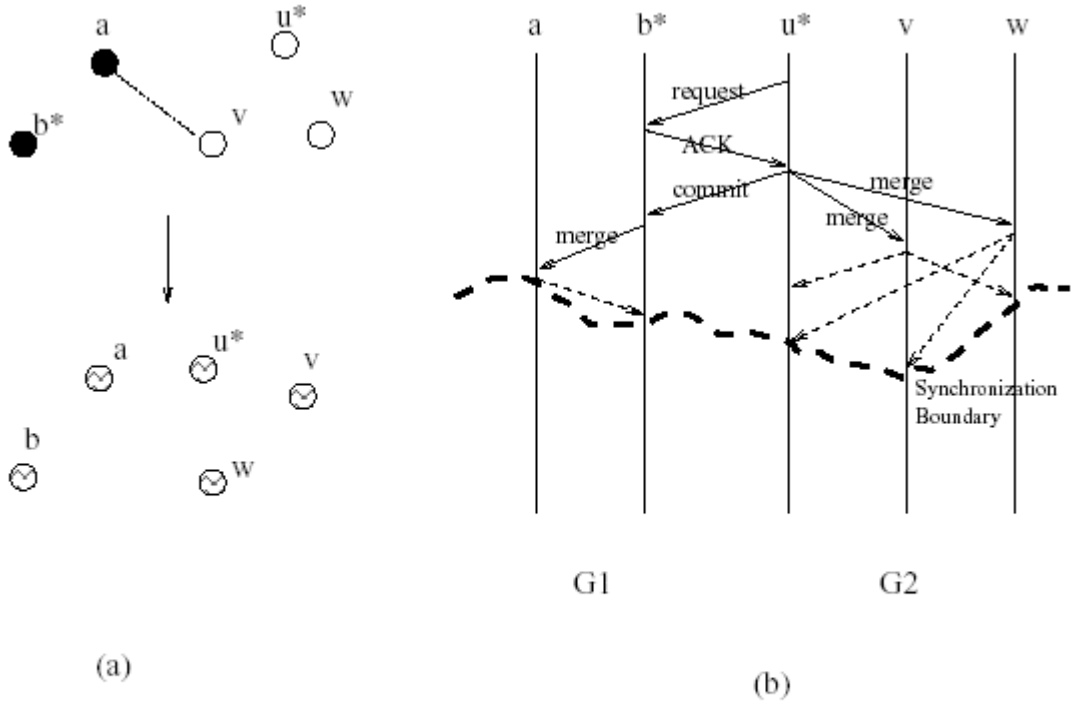

Figure 3.3 Le processus de fusion

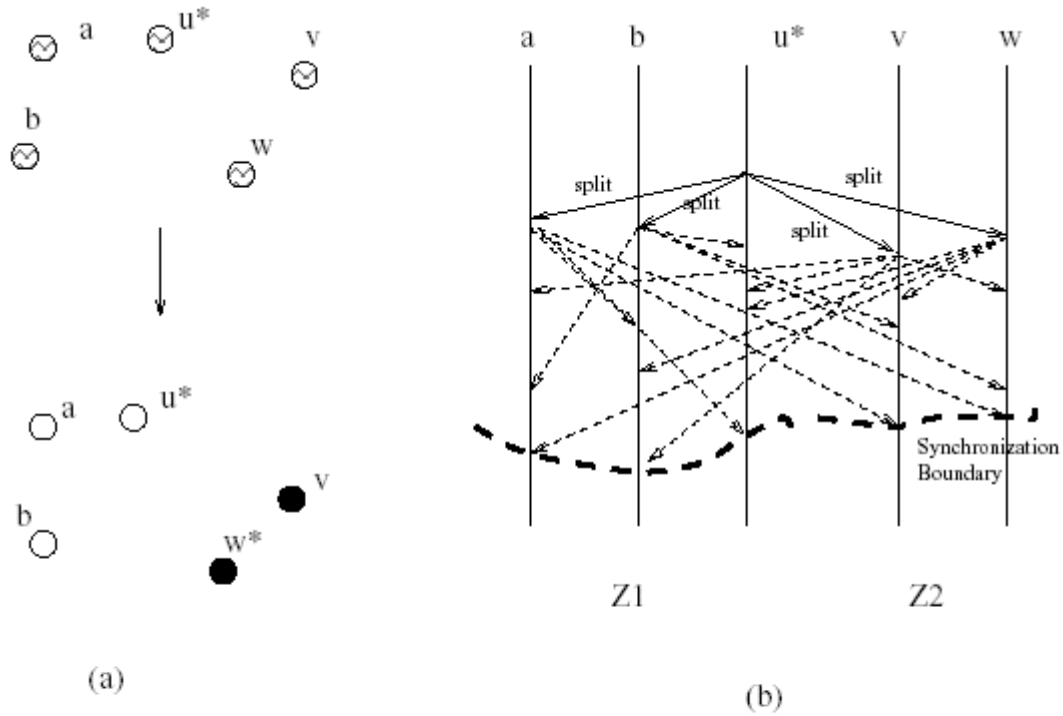

Figure 3.4 Le processus de séparation

4. Approche de Sailhan et Liu [62]

Dans la plupart des scénarios, le service de groupe est dépendant de la localisation des nœuds. Les critères d'adhésion au groupe sont la présence dans une surface géographique et le nombre de messages échangés entre les nœuds. Ceci convient le plus aux scénarios tel que d'interaction entre clients vendeurs dans les centres commerciaux, aire ports et les foires. Dans ce cas la communauté est formée par les nœuds proches, qui peuvent être dédiés au partage de ressources en tenant compte des propriétés de la Qualité de service (QoS), tel que l'amélioration de la disponibilité des données [16] ou le temps réel d'exécution [48] [90].

Cependant, le service de gestion de groupe peut dépendre également de la loyauté des membres du groupe. Par exemple, il est possible de restreindre l'adhésion pour permettre aux nœuds d'assurer des interactions sécurisées entre les membres. D'autres contraintes peuvent être considérées : la taille du groupe [72] et/ou niveau des membres du groupe ex, capacité de stockage du nœud.

4.1. Attributs du service de groupe

En premier lieu le groupe est défini par une fonction de fonctionnalité noté comme *f* qui peut caractériser les différentes caractéristiques supportées par le nœud comme les ressources et les services. On utilise *support(x, f)* pour dénoter le fait que x offre la fonction *f, Gf* pour dénoter un groupe avec la fonction f. i.e., $Gf = \{x \mid x \in N \text{ et } support(x, f)\}$.

4.1.1. Modèle de réseau

Le réseau est considéré comme un ensemble de *N* nœuds, et que chaque nœud *x* de *N* a un unique identifiant *Id(x).*

Proximity (x, p) retourne la distance géographique entre la position du nœud x et une position p.

Distance (x, p) retourne la distance géographique entre la position de deux nœuds x et y.

Hops (x, p) retourne le nombre de sauts entre x et y pour chaque y accessible de x.

4.1.2. Localisation

La définition du système de gestion de groupe sous les contraintes de la localisation des nœuds membre du groupe est comme suit :

- *Localisation unaware groupe* [58] [60]: est définie seulement par la fonction présentée par le groupe
- *Proximity based groups* [72] [71]: est définie comme la position des membres du groupe dans une zone géographique donné avec référence à un autre membre du groupe, c'est la position par rapport aux autres membres du groupe. Soit *pos* la position de référence géographique et *dist* la distance géographique maximale permise alors on aura :

Geographical_Proximity (Gf ; pos; dist) $\Leftrightarrow \forall x \in$ Gf° : Proximity(x; pos) < dist

Relative_Proximity (*Gf*; *dist*) $\Leftrightarrow \forall x, y \in Gf°$ *: Distance*(*x; y*) < *dist*

4.1.3. Échelle

Bounded groups [15] [86] [05] est définie en se basant sur le maximum de nombre de sauts *H* entre les nœuds et un nœud spécifique (généralement le leader du groupe noté *l*) :

Bounded (*Gf*; *H*) $\Leftrightarrow \forall x, y \in Gf°$*: Hops* (*x; l*) $\leq H$

Group size [23] est définie comme le nombre maximum de membres du groupe (et on considère la taille maximal du group est *N*)

size (*Gf*; *N*) $\Leftrightarrow |Gf| \leq N$

4.1.4. Crédibilité

Le service de gestion de groupe peut être composé que de nœuds qui se font confiance entre eux. Ceci veut dire que chaque nœud a une réputation à maintenir envers les autres. Cette réputation est basée sur les expériences passées ou des recommandations des autres nœuds crédibles [61]. Ainsi le service de groupe utilise la réputation noté *rep* entre les nœuds du groupe. On considère la crédibilité de la manière suivante :

Trusted (*Gf*; δ) $\Leftrightarrow \forall x, y \in Gf°$*: rep* (*x;* y) $\leq \delta$ où δ est un seuil fixé par l'aplication.

De plus un certificat *SC* de confiance peut être établi par un tiers nœud crédible [08]

SC (Gf; *CA*) $\Leftrightarrow \forall x \in Gf°$*:* $x \in$ *authorized* (CA)

4.1.5. Qualité de service

Les groupes peuvent aussi être définis par certains attributs liés à la qualité de service [63] comme la fiabilité, performances et transaction pour les services et la charge du cpu, la mémoire et la batterie pour les ressources.

De plus le service de gestion de groupe peut être limité en termes de mobilité des nœuds pour limiter la probabilité qu'un nœud quitte le groupe. Par exemple, la mobilité des nœuds a été suggérée comme un critère d'intégration dans le groupe [86] (voir section3) et qui peut prendre en charge le groupe basé sur un modèle de mobilité [49]

En général, la déconnexion d'un nœud est causée soit par sa mobilité et /ou de sa pénurie de ressources.

4.1.6. Catégorisation des attributs

Les attributs cités ci-dessus peuvent être caractérisé par leur champ d'application qui peut être locale ou de groupe.

La différence entre *Geographical_Proximity* et *Relative_Proximity* est pour la premier le service de groupe peut être confirmé localement comme par exemple le calcul de la distance à partir d'un point fixe. Par contre pour le deuxième service de gestion de groupe doit être confirmé au niveau du groupe comme par exemple le calcule du nombre de nœud du groupe.qui est accomplit au moment de la formation du groupe. Noter que ces attributs ne sont pas exclusive pour chacun d'eux mais ils peuvent être combiné pour la définition du groupe

4.2. Modèle de gestion de groupe

A cause des challenges posés par la mobilité et les ressources limitées des nœuds, les auteurs de cette approche ont considéré les propriétés nécessaires pour le service de management de groupe :

- Economie des ressources : le coût en ressources du service de management de groupe doit être supporté par le système.
- Distribution : le service de management de groupe n'est pas une solution centralisée où un seul nœud gère tout le service à cause de la mobilité et au coût de la tache.
- L'aspect dynamique : le service de management de groupe doit s'adapter au fort aspect dynamique de la topologie du réseau.

Trois fonctions dans le service de management de groupe ont été définies :

1) Découverte des membres de groupe : découverte des nœuds mobiles éligibles pour être du groupe en accord avec les attributs applicables au service de gestion de groupe
2) Initialisation du groupe : échanges des informations nécessaires et applicables aux fonctionnalités et contraintes globales du service de groupe.
3) Management dynamique de groupe : mètre à jour les membres du groupe suivant l'aspect dynamique de la topologie du réseau

Il est considéré que chaque nœud du réseau est capable de connaitre son voisin direct et de détecter le départ de ses voisins directs et l'arrivée de nouveaux voisins directs à partir de la table de routage.

4.2.1. Broadcast de groupe

Notion de broadcast de *H* saut :

Pour un broadcaste de *H* sauts, l'utilisation du broadcast à deux sauts est adopté pour réduire la surcharge du trafic. L'émetteur du broadcast en utilisant les informations de la topologie en deux sauts réunies à partir des beacons périodiques, sélecte parmi les voisins à un saut le nombre minimal de nœuds qui peuvent couvrir tous les nœuds de moins de deux sauts et qui sont appelé les nœuds de broadcaste, seulement ces nœuds sélectionnés propagent le broadcast jusqu'a au *H* nième sauts. Les boucles sont évitées en utilisant un numéro unique pour le message.

4.2.2. Découverte des membres du groupe

A la création d'un groupe *Gf*, un nœud émet un message « *disc* »de découverte qu'il broadcaste à 2 sauts à ses voisins. Ce nœud est donc l'initiateur du groupe. Le message disc contient les attributs fonctionnels et non fonctionnels du groupe en création et son unique identifiant id. Par exemple pour créer un groupe de tous les nœuds se trouvant à proximité d'un point fixé, l'initiateur inclut cette référence dans le message « *disc* ».

En recevant le message « *disc* »un nœud vérifie la conformité avec ses propres contraintes, s'il est intéressé. Entre temps s'il est un nœud de broadcast il doit déterminer s'il diffuse le message « *disc* » selon les règles suivants : il ne diffuse pas le message « *disc* » si il n'a aucun voisins a part ceux de qui il reçut le message ou in informe les nœuds précédents (un nœud de broadcast ou l'initiateur) que les nœuds découverts de son coté ne sont pas dans les contraintes du groupe.

Contrairement aux nœuds non de broadcast qui ne font que répondre en exprimant leurs volontés de joindre le groupe, les nœuds de broadcast doivent attendre l'arrivée des réponses de tous leurs voisins avant de répondre à leur nœud précédent (un nœud de broadcast ou l'initiateur) avec un message d'adhésions « *join* », ce message contient les nœuds voulant joindre le groupe (y inclut lui-même si il veut joindre le groupe) et les informations relatives aux nœuds tels que la position… . Et si le nœud de broadcast ne reçoit aucun message d'adhésion « *join* », positive il répond par un message « *join* », négatif pour l'indiquer.

Dans le cas où plusieurs nœuds initiateur de groupe de mêmes fonctionnalités lancent des messages « *disc* » en même moment on peut les fusionner en choisissant un nouvel initiateur parmi ces initiateurs.

Si l'initiateur tombe en panne durant cette procédure elle doit être refaite.

4.2.3. Initialisation du groupe

Chaque groupe possède un leader appelé group leader responsable de gérer les activités du groupe tout en faisant respecter les contraintes du groupe et la communication intergroupe. Cependant, avant la formation du groupe l'initiateur du groupe c'est le leader temporaire du groupe. L'initialisation du groupe ne commence que lorsque l'initiateur reçoit toutes les réponses de ses voisins. Une des parties importantes de l'initialisation est d'imposer les contraintes globales de groupe dont les suivantes sont le plus importantes :

- **Proximité relative** : qui impose que dans chaque paire de deux nœuds, les nœuds doivent être à une distance donnée les uns des autres. Cette distance peut être calculée à base du message « *join* ».
- **Réputation :** qui impose que deux nœuds doivent avoir une bonne réputation entre eux. Etant très difficile d'imposer cette réputation entre les nœuds qui est confidentielle dans la plupart des cas est peu plausible de donner toutes les valeurs de réputation au leader. Elle peut être accomplie en considérant que le leader est assez digne de confiance pour coordonner les interactions entre membres du groupe, ou les recommandations du leader sont assez fort pour fortifier la réputation des membres entre eux. Cette réputation peut être imposée localement durant la phase de découverte des membres du groupe.
- **Taille** : qui impose que le nombre total des membres du groupe ne peut dépasser un seuil déterminé. Ceci peut se faire facilement en comptant le nombre de nœuds voulant joindre le groupe via le message « *join* ».

Noter que les contraintes globales de groupes peuvent conduire à des exclusions des nœuds du groupe même s'ils ont envoyées le message « *join* ». C'est ce qui ce passe pour la contrainte de taille par exemple. Si le leader a déjà accepté un nombre maximal de nœuds, ces nœuds exclus peuvent former un autre groupe en exécutant une autre procédure de découverte de groupe.

Dans le cas ou le groupe est basé sur les certificats. La procédure d'initialisation est complétée par un protocole de gestion de clé GKA établissant des clés partagées entre les membres du groupe. La clé est utilisée pour chiffrer tous les messages dans le groupe [08].

Pour la découverte des leaders des autres groupes, le leader diffuse périodiquement sur un rang plus vaste que la taille du groupe un message pour annoncer son existence aux éventuels leaders d'autres groupes avec les mêmes contraintes et fonctionnalités

4.2.4. Gestion dynamique de groupe

Comme c'est le cas de tous les nœuds, le leader peut quitter le groupe. Dans ce cas, il est nécessaire de faire passer le rôle de leader aux autres membres. Pour repartir la charge de gestion du groupe, la sélection du leader se fait comme suit :

Plusieurs critères on été proposé pour la sélection du leader dans les réseaux ad hoc, incluant :

(i) le plus connecté, où le nœud avec le plus grand nombre de voisins. Cette approche possède un très petit taux de changement de leader, mais soufre de faibles apports étant donné que le débit est partagé entre les membres du groupe [23].

(ii) extremum id (ex. [21]), où le nœud qui a le plus petit ou le plus grand id qui est choisit comme leader. Le point faible de cette approche est sa tendance envers les nœuds avec l'extremum id.

(iii) poids du nœud (ex. [07, 23, 95]) où une métrique intégrée pour l'évaluation de la convenance d'un nœud pour être le leader. Cette approche peut dépendre de divers facteurs comme les ressources et les nœuds voisins.

Ces critères sont essentiellement basés sur les attributs du groupe comme illustré dans la section 4.1. Par conséquent, il est naturel de les intégrer pour évaluer la convenance d'un nœud pour être le leader, par l'adoption des différents poids aux différentes métriques pour différents scenarios. En particulier ce qui est présenté ici, en considèrent les deux métriques suivantes pour l'élection du leader de groupe :

1) *La richesse en ressources*: les nœuds puissants avec des ressources abondantes sont considérées plus convenables pour devenirs des leaders. Cela est due au fait que le leader consomme plus d'énergie et de temps de calcul que les autres membres du groupe. En plus d'augmenter les performances du groupe car les leaders faibles ont tendance à être des obstacles. Les ressources suivantes sont prises en considération : charge du CPU ; mémoire ; batterie et bande passante.

2) *Le temps entre le présent et de la dernière fois ou il a été leader :*ce facteur contribue a al distributions de charges sur les membres de groupe.

Le poids global peut être calculé comme suit, avec une impotence relative aux différentes applications définies par l'initiateur du groupe (ex. w_1, w_{11}) :

Poids = Ressources * w_1 + temps_écoulé * w_2

$w_1 + w_2 = 1$

Ressources * w_1 = CPU_charge * w_{11} + mémoire *w_{12} + batterie *w_{13} bande_passante * w_{14}

$w1 = w_{11} + w_{12} + w_{13} + w_{14}$

Noter qu'il est facile d'étendre ce calcul pour couvrir d'autres métriques. Les métriques ont besoin de normalisation puisque elles sont dans des parties différentes et donc incomparables. Dans le cas où le poids global est identique, le leader est sélectionné au hasard. La rotation du rôle de leader commence s'il existe des nœuds qui ont un poids global plus grand que celui du leader actuel. Ceci est due à multiple raisons : Le poids du leader a diminué a cause du temps qu'il a passé comme leader, ou les ressources ont augmenté. Le changement du leader nécessite un transfert d'informations qui sont seulement tenues par le

leader (les leaders des groupes voisins). Le nouveau leader doit informer l'ensemble des membres du groupe des changements.

La gestion de groupe dans les réseaux ad hoc requiert une gestion des mobilités qui induisent des changements dans la liste des membres du groupe, d'une manière à la garder transparent aux applications. Si un nouveau nœud est détecté par la couche réseau, un mécanisme d'événements peut être installé sur le leader pour l'inviter à joindre le groupe, dans la condition ou le groupe est capable d'accepter plus de membres. Le départ de nœuds peut aussi être géré par le leader en envoyant un message de mise a jour de la liste des membres du groupe.

5. Conclusion

Dans ce chapitre nous avons présenté les approches les plus intéressantes pour le service de gestion de groupe dans les réseaux mobiles ad hoc. Nous avons décrit leurs principales caractéristiques et fonctionnalités afin de comprendre les stratégies utilisées pour la construction des groupes et leurs maintiens.

Assurer la gestion de groupes dans un réseau ad hoc est un problème très complexe vu l'aspect dynamique et l'évolution rapide de la topologie. En effet, les unités mobiles sont dynamiquement et arbitrairement éparpillées d'une manière où l'interconnexion peut changer à tout moment. Le but du service de gestion de groupe est de masquer aux applications utilisant les groupes les problèmes de changement de configuration du groupe.

Différentes approches ont été présentées : la première est une architecture multi niveaux qui contient un protocole de couche de proximité synchrone pour déterminer l'ensemble des nœuds voisins d'un nœud donné dans le réseau. Cette information est utilisée par le protocole de gestion de groupe pour la construction des groupes, utilisées par les applications distribuées. Des inexactitudes peuvent exister lors de la phase d'identification des membres du groupe due à la mobilité des nœuds. De plus cette architecture requière des capacités de la couche de routage qui est un axe de recherche active dans les réseaux ad hoc. La seconde approche étudiée utilise une vue logique de la connectivité des nœuds pour masquer les déconnexions, son but est de maintenir une vue consistante de l'état globale du système. La troisième approche regroupe les différents attributs du groupe ce qui permet de définir plus de contraintes rencontrées par le service de gestion de groupe.

Les solutions utilisées étant complémentaires, la nécessité de proposer une solution d'union est presque évidente. L'utilisation de plus d'informations sur les membres du groupe permet de prédire plus d'événements de déconnexion qui ne sont pas prises en compte par les solutions existantes, tel que l'utilisation de l'information sur l'énergie.

Chapitre 4

Energie et réseaux Mobiles Ad hoc

1. Introduction

Les réseaux mobiles ad hoc sont des réseaux sans fil multi-sauts où tous les nœuds coopèrent pour le maintien de la connectivite. Ce type de réseaux est très utile dans les situations où l'établissement d'un réseau temporaire est indispensable comme dans les cas des catastrophes. Dans de telles situations, les sources d'énergie se font rares et les nœuds mobiles sont alimentés par des batteries de durée limitée, il est alors nécessaire de minimiser leur consommation d'énergie.

La construction de tels réseaux pose des défis techniques significatifs, vu les contraintes imposées par l'environnement. Ainsi les appareils utilisés sur le terrain doivent êtres légers et mobiles. Ils doivent être économes en énergie. Plusieurs technologies sont développées pour atteindre ces objectifs. Ces technologies visent généralement des composants bien spécifiques pour optimiser leur consommation d'énergie. Par exemple, l'utilisation d'affichage à énergie faible [53], conception d'algorithmes de réduction de la consommation d'énergie du disque dur [36] [64] [98], utilisation de périphériques d'entrée/sortie de faible consommation d'énergie tels que les cameras [30], etc. D'autres recherches visent le développement de processeurs ayant une consommation réduite d'énergie et bien sur le développement des batteries de plus en plus performantes.

L'énergie consommée par l'interface réseau est la plus importante, surtout pour les petites unités [FOR94, DDB03]. Les premières recherches sur la minimisation de l'énergie consommée par les unités mobiles ont concerné le niveau matériel. Ces recherches visent les réseaux structurés autour d'infrastructures fixes qui n'ont pas le problème de limitation de source d'énergie.

La couche logicielle doit prendre en considération ces limitations, et plus particulièrement au niveau du protocole de routage, là où les paquets suivent des chemins multi-sauts. Les principaux axes de recherches actuels se concentrent sur la conception des applications et protocoles de la couche transport, MAC et surtout de routages, qui minimisent l'énergie consommée. Le but étant d'assurer la plus longue durée de vie aux batteries des terminaux, et éviter ainsi leurs déconnexions et le partitionnement du réseau.

Dans ce chapitre nous allons présenter les mécanismes existants pour réduire l'énergie consommée par les terminaux mobiles afin de rallonger la durée de vie des réseaux mobile ad hoc.

2. Mécanismes de réduction de la consommation d'énergie

Beaucoup d'efforts et de recherche sont actuellement focalisés sur la réduction de la consommation d'énergie dans tous les aspects d'une unité mobile. Dans ce qui suit, nous allons décrire les techniques de conservation d'énergie les plus importantes.

2.1. Gestion du Disque

Une des méthodes de conservation d'énergie est d'arrêter le disque dur lorsque le mobile est en mode veille (idle). [94] présente une analyse quantitative du coût et des profits de la mise en arrêt du disque en mode veille. Les tests sont exécutés sur des machines DOS et UNIX. Le paramètre mesuré est le *délai d'arrêt* du disque. Le délai d'arrêt est l'intervalle de temps qui sépare le moment où le disque entre en veille de celui où il est arrêté. Les résultats montrent que le maximum de sauvegarde d'énergie est atteint en utilisant un délai de 2 secondes qui est différent du délai de 3 à 5 minutes recommandé par les fabricants. D'après [94], avec des délais petits le disque s'arrête plus longtemps et de là il économise plus d'énergie.

L'inconvénient de mettre en arrêt le disque avec des délais courts est le temps et l'énergie nécessaire pour remettre le disque en route, qui se répercutent sur les délais de l'utilisateur. Cependant, Les simulations données dans [94] montrent que l'arrêt du disque se produit 8-15 fois par heure. Ceci se traduit par 16-30 secondes de délai de l'utilisateur, ce qui est raisonnable comparé au gain d'énergie engendré.

2.2. Gestion du processeur

L'énergie consommée par le processeur est directement proportionnelle à la tension fournie, la capacitance des différents périphériques, et la fréquence de fonctionnement. Plus la fréquence du processeur est grande plus il consomme de l'énergie. Cela est dû à la mémoire CMOS qui change d'état à chaque cycle de l'horloge. Chaque changement d'état est un court-circuit qui consomme de l'énergie.

L'énergie nécessaire au processeur est donnée par la formule CV^2F [66], où C est la capacitance des fils, V est la tension fournie et F est la fréquence de fonctionnement. Il existe plusieurs algorithmes pour réajuster la fréquence de l'horloge en mode veille. L'idée principale est d'équilibrer l'utilisation du processeur entre utilisation intensive et mode veille. La gestion des processus est un moyen efficace pour accomplir cette tache.

En général, presque tout les processus ont un deadline pour être exécuté. D'après [66], même lorsqu'il y a des processus qui demandent à être exécutés, le processeur passe par des moments d'inactivité. En utilisant un algorithme d'ordonnancement approprié, le processeur peut se mettre en mode veille durant ces moments d'inactivité. Durant, ces moments le processeur peut être arrêté et la tension réduite permettant ainsi une économie d'énergie. L'idée de base de ces algorithmes d'ordonnancement consiste à prolonger le traitement des processus dans le temps en respectant leurs deadlines.

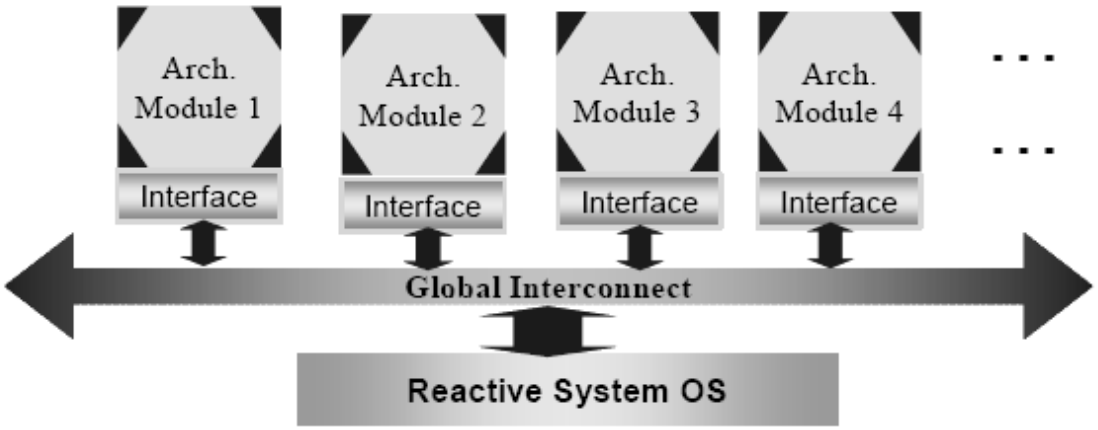

Figure 4.1 Architecture d'un nœud mobile

2.3. Allocation de la mémoire

Dans les unités mobiles, l'accès à la mémoire est une des tâches les plus consommatrices d'énergie [94]. Comme la plupart des unités mobiles n'ont pas de mémoire secondaire, la consommation d'énergie par la mémoire est cruciale et doit être optimisée. Avec l'apparition des *Direct Rambus DRAM (RDRAM),* les unités mobiles peuvent être dans différents états d'énergie : actif, en attente, en veille ou éteint.

Il existe deux méthodes pour gérer l'état de ces mémoires :

- Statique : la mémoire fonctionne tout le temps avec le même mode énergétique.
- Dynamique : dans cette méthode, le système essaye de prévoir les requêtes d'accès en se basant sur des modèles d'accès pour pouvoir minimiser l'utilisation de la mémoire.

Les politiques d'arrangement de code et des données peuvent aussi aider à réduire la consommation d'énergie. Si les pages actives sont groupées ensembles sur la même carte mémoire, la partie restante de la mémoire peut être éteinte. Cette technique aide à réduire l'énergie consommée durant les lectures de données. Les simulations données dans [94] montrent que l'on peut atteindre de 6% à 50% d'économie d'énergie avec ces méthodes.

2.4. Contrôle de la puissance de transmission

Dans tous les protocoles de routage standards des réseaux ad hoc, aussi bien proactifs que réactifs les paquets sont transmis en utilisant une puissance maximale fixe, pour couvrir tout le domaine de puissance. En fait, les paquets de données ne sont généralement pas propagés, mais plutôt destinés à un voisin unique soit pour le router, soit par ce qu'il a atteint sa destination finale, qui se trouve généralement à une distance inférieure au domaine de puissance maximal. Au lieu d'envoyer sous cette puissance, il est très intéressant d'utiliser une puissance dynamique qui dépend de la distance entre les nœuds [32].

Pour bien comprendre le gain d'énergie provenant de l'utilisation de ces techniques, il faut d'abord présenter un modèle de calcul d'énergie de transmission, nous présentons celui proposé dans [91, 59]:

L'énergie consommée pour transmettre un paquet de données de taille D octets via un lien donné, peut être modélisé par la formule simplifiée suivante :

$E(D) = K1*D + K2$

Tel que :

K 1 = Pt packet * 8 / BR

K 2 = (Pt MAC * D MAC + Pt packet * D header) * 8 / BR

BR : est le débit

Pt^{MAC} : la puissance à laquelle les paquets de la couche MAC (de contrôle) sont envoyés

D^{MAC} : la taille (en octets) des paquets de la couche MAC

Pt^{paquet} : la puissance à laquelle les paquets de données sont envoyés

D^{header} : la taille (en octets) des entêtes des paquets de données

Les protocoles de routage standards utilisent la valeur maximale de Pt^{paquet}, pour couvrir tout le domaine de puissance, ainsi l'énergie maximale ne dépend que de la taille du paquet:

Emax (D) = K1max*D + K2max

Tel que K_{1max} et K_{2max} sont respectivement obtenus à partir des expressions de K_1 et K_2 en remplaçant Pt^{packet} par sa valeur maximale.

La puissance de transmission minimale Pt requise pour une réception correcte est:

Pt(d) = Pr * dn / K

Tel que :

d est la distance, et K est une constante

n : est un paramètre d'atténuation, il prend pratiquement la valeur de 4.

Pr : est une puissance seuil (puissance minimale pour une bonne réception)

L'énergie consommée en utilisant cette puissance minimale est:

E_{min} (D, d) = K_{1min} D + K_{2min}

Tel que K_{1min} et K_{2min} sont respectivement obtenus à partir des expressions de K_1 et K_2 en remplaçant Pt^{packet} Par Pt(d)

Théoriquement, si on utilise cette puissance minimale le gain d'énergie pour la transmission d'un paquet de taille D à une distance d, par rapport à l'utilisation de puissance maximale est :

S (D, d) = E_{max} (D) - E_{min} (D, d)

La figure 3.1 inspirée de [91] montre le gain d'énergie par rapport à la distance, on remarque que plus les nœuds se rapprochent (la distance est petite), plus le gain est important.

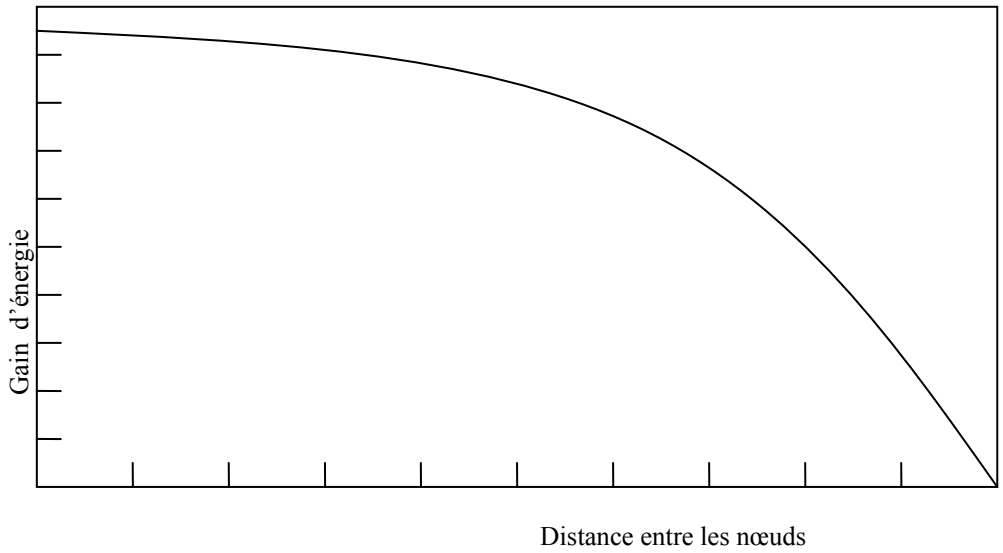

Figure 4.2 gain d'énergie par rapport à la distance

Puisque cette courbe n'est pas linéaire, l'utilisation des chemins à plusieurs sauts (plus longs), peut économiser l'énergie part rapport à l'utilisation des chemins les plus courts.

Pour bien comprendre ce phénomène on va présenter un exemple.

Soit trois nœuds alignés *a*, *b* et *c*, et supposons que c est à la portée de communication de a, mais qu'il est plus loin de *a* que *b* (*b* est entre *a* et *c*), le protocole classique (qui choisi les plus courts chemins) envois les paquets directement de a à c, si le chemin choisi est *a-b-c*, alors l'énergie requise pour envoyer un paquet de taille *D* est :

E_{multi} (D, d, d1) = E (D, d1) + E (D ,d-d1)

Tel que *d1* est la distance entre *a* et *b*, et d celle entre a et c et *E (D, d1), E (D, d-d1)* sont respectivement l'énergie d'envoi du paquet de *a* à *b* et de *b* à *c*.

Si *E(Dd)* est l'énergie d'envoi du paquet directement de a à c, alors le gain d'utilisation du chemin le plus long *S (D, d, d1)* est :

S(D,d,d1) = E(D,d) - E_{multi} (D,d,d1)

c-à-d : S(D,d,d1) = E(D,d) - E(D,d1) - E(D,d-d1)

En fait l'énergie économisée obtenue en utilisant plusieurs sauts dépend des distances d et d1.

On remarque que, pour les grandes valeurs de d, le gain est important, et il diminue avec la diminution de d. On remarque aussi que le gain diminue quand b s'approche de a ou de c, et peut même devenir négative (c à d que le chemin le plus court a-c consomme moins d'énergie que a-b-c).

2.5. Eteindre l'interface réseau

Dans les protocoles MAC standards (tels que IEEE 802.11 [74]), les interfaces réseaux sont toujours allumées, et même si elles ne sont entrain ni d'émettre ni de recevoir des données, elles seront mise en mode veille. Les résultats des expériences réalisées par jean-Pierre Ebert, Brian Burns et Adam Wolisz de l'université technique de Berlin, montrent que la consommation d'énergie dans ce mode est importante et très proche du mode de réception [55].

Dans la figure 4.3, la transmission du nœud A au nœud B est entendue par hasard (OVERHEARD) par le nœud C, qui est un voisin de A. Le nœud C va perdre son énergie en recevant un paquet qui n'est pas destiné à lui, ou en restant en mode veille durant la période de transmission. Dans ce cas, il est clair que le nœud C doit être éteint durant cette transmission pour conserver son énergie.

Dans [93], les auteurs proposent un protocole dit PAMAS (Power-Aware Multiple Access Protocol with Signaling). Dans ce protocole, un nœud peut éteindre son interface réseau dans l'un des cas suivants :

Le nœud capte des transmissions qui ne le concernent pas et il n'a pas de paquets à émettre.

Au moins un voisin est en train d'émettre et au moins un voisin est en train de recevoir. Dans ce cas même si le nœud a des paquets à émettre, il ne peut le faire de peur d'interférences.

Si tous les voisins sont en train d'émettre et le nœud n'est pas un récepteur.

Le problème fondamental de cette solution est comment estimer la durée durant laquelle le nœud reste éteint ?

Dans tous les protocoles MAC des réseaux ad hoc, les nœuds vérifient le canal en échangeant les messages RTS/CTS (Ready To Send et Clear To Send). Ainsi l'émetteur envoie un message RTS, et le récepteur répond par un message CTS s'il reçoit correctement RTS, et par la suite l'émetteur commence la transmission. Dans PAMAS, cet échange de messages RTS/CTS prend place dans des canaux spéciaux de telle sorte à ne pas perturber les transmissions de données en cours. Ces messages contiennent la taille des paquets qui vont être émis. Ainsi, n'importe quel nœud dans le voisinage peut utiliser cette information pour déterminer la durée de la transmission et éteindre leur interface durant cette durée. Cependant, un autre problème reste posé : quand un nœud s'éteint, ensuite s'allume pendant une nouvelle transmission de données (il ne capte pas les messages RTS/CTS de cette transmission). Dans ce cas, le nœud doit être capable d'estimer la durée de transmission restante. Une présentation détaillée de ce protocole est disponible dans [93].

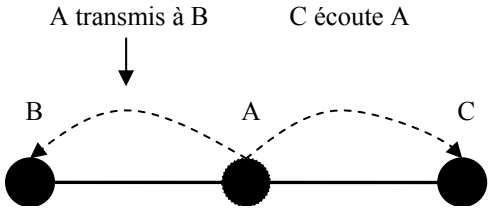

Figure 4.3 Consommation inutile

2.6. Utilisation des antennes dirigées (DIRECTIONAL ANTENNAS)

Les protocoles et algorithmes proposés supposent que les nœuds sont équipés d'antennes omnidirectionnelles, c'est à dire que les nœuds ont des angles de couverture de 360°. Alors ils n'ont pas besoin de pointer leurs antennes pour communiquer. Bien que cette méthode soit très simple, elle

cause un gaspillage important d'énergie, puisque l'énergie est propagée vers toutes les directions. Il est donc plus avantageux d'utiliser des antennes dirigées pour économiser l'énergie.

Le gain d'énergie des antennes dirigées par rapport aux omnidirectionnelles dépend des angles (d'élévation et de largeur) utilisés. Plus l'angle est petit, plus le gain est important, ce gain dépend aussi de la suppression du lobe secondaire. Les définitions des ces angles est bien illustré dans [22].

Si on suppose que l'énergie dégagée dans le lobe secondaire est négligeable c'est à dire que toute l'énergie est propagée dans le lobe principal, et que toute l'énergie consommée par l'antenne est convertie en radiation (l'efficacité de l'antenne est 100%,). Alors le gain d'énergie est : [06]

Gain = 4Π / (θ *φ)

Ou θ et φ sont respectivement les angles d'élévation et de largeur (AZIMUTH) en radian.

Si l'émetteur et le récepteur utilisent, tous les deux, des antennes dirigées, le gain total d'énergie est alors :

Gain = Gain (TX) * Gain (RX)

Ou *Gain (TX)* et *Gain (RX)* sont donnés par la formule précédente.

Bien que ce gain soit dans le cas idéal, et qu'en pratique les contraintes posées ne sont pas satisfaisables, ce qui fait diminuer le gain d'énergie. L'utilisation des antennes dirigées permet aux nœuds d'un réseau ad hoc de communiquer en consommant beaucoup moins d'énergie par rapport à l'utilisation des antennes omnidirectionnelles. En plus, cette technique permet d'assurer plus de sécurité, et particulièrement de confidentialité, de données, en diminuant la probabilité de détection.

Mais tout ceci nécessite un protocole qui soit normalement au niveau de la couche MAC pour diriger les antennes, et synchroniser les utilisateurs. L'exécution de ce protocole cause une consommation d'énergie supplémentaire, ce qui cause un compromis avec le gain. Dans [06] on propose des solutions basées sur cette technique.

2.7. Utilisation d'un protocole de routage efficace

Dans les premiers protocoles de routage construits pour les réseaux ad hoc, les chemins sont choisis en se basant sur la minimisation du nombre de sauts, ou de délais. De cette façon un ensemble limité de nœuds peut être utilisé dans plusieurs routes, ce qui cause une perte importante d'énergie, et ainsi une déconnexion rapide de ces nœuds.

Beaucoup d'études théoriques ont montré que l'énergie consommée pour le routage des paquets de données dans les réseaux ad hoc peut être énormément réduite, par rapport aux protocoles classiques (MIN-HOP ROUTING). Mais en pratique ce but est très difficile à réaliser. En fait, même sans tenir compte de la limitation en sources d'énergie, la mobilité posait, et pose toujours un grand problème aux protocoles de routage. Les auteurs de [33] montre que l'augmentation de la mobilité cause une

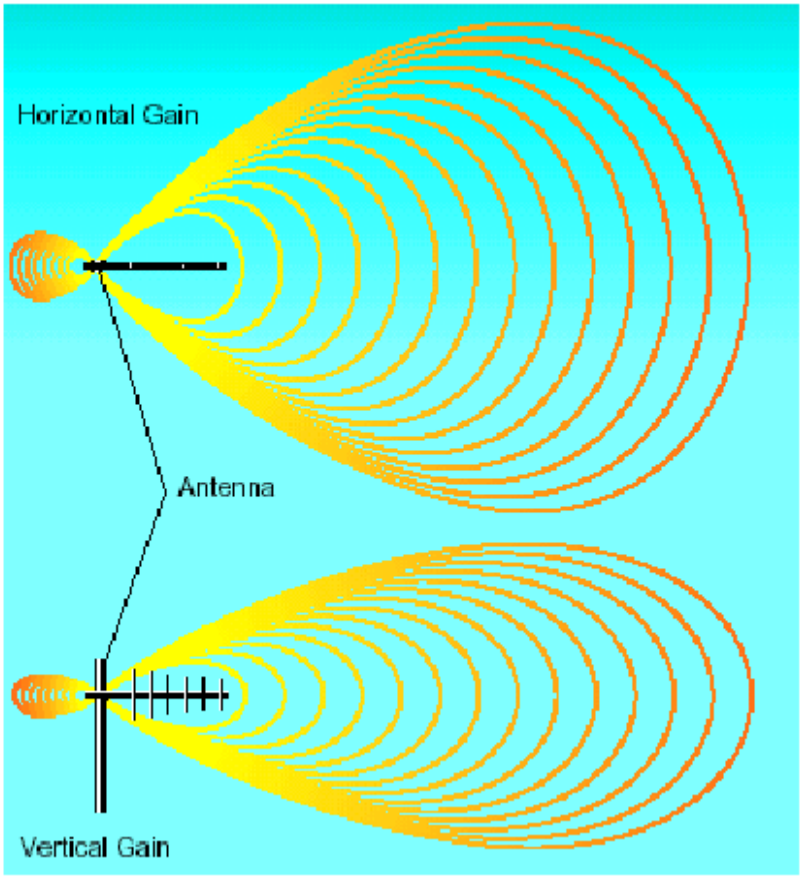

Figure 4.4. Gain des antennes dirigées

forte dégradation de performances des protocoles de routage, et que la conception d'un protocole de routage qui s'adapte rapidement au changement fréquent de la topologie causé par la mobilité, est un grand défit.

Maintenant si on tient compte du facteur d'énergie, le problème de routage devient encore plus compliqué. Le but est non seulement de trouver des chemins stables et résistants à la mobilité, mais aussi de router les paquets, d'une part via des nœuds qui ont suffisamment d'énergie, et d'autre part via des chemins qui cause le minimum de consommation d'énergie. Les travaux actuels ajoutent des modifications aux protocoles classiques, en proposant des nouvelles techniques et critères pour la découverte et le choix des chemins, ce qui permet de créer de nouvelles versions de protocoles

existants nommés (POWER-AWARE ROUTING, MINIMUM ENERGY ROUTING, ou aussi ENERGY EFFICIENCY ROUTING).

Il est à noter que la technique de contrôle de la puissance de transmission peut être utilisée pour la conception d'un tel protocole de routage.

3. Conclusion

Le problème de l'économie d'énergie est de plus en plus pris en considération. L'intégration de techniques d'économie d'énergie dans tous les protocoles et applications s'exécutant sur les réseaux mobiles ad hoc est devenue plus qu'indispensable. Ce problème doit être pris en compte à tous les niveaux matériels et logiciels. Toutes les applications doivent tenir compte de l'énergie pour la conserver d'une part et être capable de s'exécuter avec d'autre part.

Chapitre 5

Service de gestion de groupe énergie aware pour les réseaux mobiles ad hoc

1. Introduction

Dans les chapitres précédents nous avons vu que dans les différentes approches existantes le service de gestion de groupe est dépendant essentiellement de la localisation des nœuds, et aussi de la fonction du groupe, cependant ces deux contraintes ne sont pas suffisantes pour un service de groupe fiable. Pour cela, il faut prendre en confédération les autres contraintes du réseau mobile ad hoc. L'information sur l'énergie étant majeure dans les réseaux ad hoc a suscitée beaucoup de travaux qui ont montré que c'est un problème qui nécessite la participation de tous les services s'exécutant sur le réseau ad hoc pour faire face a cette contrainte.

Proposer un service de gestion de groupe qui prend en considérations les contraintes d'énergie s'avers indispensables, pour cela, nous proposons un service de groupe plus fiable en introduisant la notion d'énergie dans toutes les phases de gestion du groupe.

Dans ce chapitre nous allons présenter un nouveau service de gestion de groupe on se basant essentiellement sur ce qui a été déjà fait dans ce domaine et en combinant les différentes solutions présentés dans les chapitres précédents, tout en proposant diverses améliorations que nous avons jugés indispensables .

2. Définition du problème

Le rôle du service de gestion de groupe est de maintenir la liste, appelée *vue*, des processus qui sont opérationnels (i.e. corrects et connectés) au sein du groupe. Cette liste évolue dans le temps avec de nouveaux membres qui entrent dans le groupe et d'anciens membres qui quittent le groupe, en sont exclus ou tombent en panne. Lorsque cette liste change, le service de gestion de groupe notifie les membres du groupe en *installant* une nouvelle vue. Les vues sont identifiées par leur numéro de séquence en partant de la vue initiale v_0 du groupe à sa formation.

Un service de gestion de groupe peut être soit à partition primaire (primary partition system) [18] soit à partitions multiples (partitionable system) [02, 69]. Dans un service de gestion de groupe à partition primaire, les vues installées par les nœuds du réseau sont totalement ordonnées (i e. dans le cas d'un partitionnement du réseau, seuls les nœuds qui sont dans la partition primaire sont considérés membres du groupe). Dans un service de gestion de groupe à partitions multiples, les vues sont partiellement ordonnées i.e. plusieurs vues disjointes peuvent coexister en parallèle. Chacune de ces vues évolue avec la partition correspondante du réseau. Tout se passe comme si plusieurs groupes évoluaient en parallèle jusqu'à ce que le réseau ne soit plus partitionné.

Vue la tendance dynamique de la topologie des réseaux ad hoc les déconnexions peuvent engendrer des partitionnements fréquentes et les réseaux partitionnés peuvent ne jamais fusionner. Ceci nous a amené dans ce travail à nous limiter à l'étude des services de groupe à partition primaire.

En général, un membre peut quitter le groupe par sa propre volonté, il est défaillant, explicitement s'il est sollicité de quitter le groupe ou expulsé par d'autres membres du groupe, et enfin à cause de sa mobilité et /ou de sa pénurie de ressources. La ressource la plus susceptible d'êtres taire est évidement la batterie. Par conséquent, l'information sur l'énergie doit être prise en compte dans les différentes étapes du service de gestion de groupe. La mobilité des nœuds est aléatoire et continue avec une vitesse maximale connue. Les nœuds voulant volontairement s'éteindre doivent signaler leur intention au préalable.

Le protocole de gestion de groupe doit gérer ces changements dynamique de manière cohérente i.e. tout membre du groupe doit avoir une vue consistante du groupe malgré les défaillances.

2.1. Principe

Le protocole que nous proposons est appelé *SGGEAMAnet* (*Service de gestion de groupe énergie aware pour les réseaux mobiles ad hoc*). Sa principale valeur ajoutée est la prise en compte de la contrainte d'énergie. Mais, est aussi une solution qui combine les avantages des trois solutions présentées dans notre état de l'art. Le but étant d'obtenir un service de gestion de groupe plus complet et utilisable dans un environnement mobile assujetti à des défaillances énergétique des membres. Pour cela nous avons introduit l'information sur l'énergie dans les différentes étapes du protocole et surtout dans la phase du maintien du groupe en prévenant les déconnexions prévisibles dues aux défaillances des membres du groupe en pénurie d'énergie. Pour se faire, notre protocole utilise la notion de réserve d'énergie de sécurité qui permet au membre atteignant ce seuil de se déconnecter sans perte de données et permet d'éviter l'affectation de rôles particuliers à certains membres du groupe. Ce qui rend le protocole plus efficace et cela en traitant plus de défaillances prévisibles.

Dans ce qui suit nous allons présenter les concepts du service de gestion groupe, nécessaires pour développer notre solution.

2.2. Concepts de bases

Notation :

G: désigne un groupe.

$v_i(G)$: est la liste des membres courants du groupe *G* (vue courante), *i* est un entier positif ou nul qui désigne le numéro de séquence de la vue. Il croit à chaque mise à jour de la liste.

Le Modèle de réseau

Le réseau est considéré comme un ensemble de *S* nœuds, et chaque nœud *x* de *S* a un unique identifiant *Id(x).*

Distance (x, y) retourne la distance géographique entre la position de deux nœuds *x* et *y*.

Hops (x, y) retourne le nombre de sauts entre x et y pour chaque y accessible de *x*

Energie (*x*) représente la quantité d'énergie disponible pour le nœud *x*

La position

La position est définie comme la position des membres du groupe dans une zone géographique donnée avec référence à un autre membre du groupe, c'est la position par rapport aux autres membres

du groupe. Soit *pos(x)* la position de *x* par rapport à une référence géographique fixe, si « *dist* » est la distance géographique maximale permise alors on aura :

$\forall x, y$ *Distance*$(x; y) = |pos(x) - pos(y)|$

$\forall x, y \in G°$: *Distance*$(x; y) <$ *dist*

- *La connectivité*

La couche MAC fournit une communication point à point et les beacons. Chaque nœud *x* émet un beacon régulièrement à chaque *t* unité de temps. La valeur de *t* dépend de : (i) la borne supérieure du nombre de nœuds qui peut être présent dans le domaine d'interférence d'un nœud ; (ii) le protocole de la couche MAC utilisé pour envoyer les beacons. Un beacon possède une portée limitée et sert comme un message de localisation « je suis là ». Les nœuds voisins qui sont dans la portée du beacon peuvent détecter la présence du nœud mobile même quand cette information n'est pas commandée par les couches supérieures.

Un réseau sans fils est connecté si :

Pour chaque nœud *x* il existe un nœud *y* tels que la *Distance*$(x; y) \leq d$, *d* est la portée des beacons ;

La *Distance*$(x; y) \leq d \Rightarrow$ *x et y* sont connecté directement *Hops (x, y)=1*;;

Quelques soient *p , q , r* des nœuds du réseaux , si *p* est connecté directement à *q* et *q* est connecté directement a *r* alors *p* est connecté a *r* avec deux sauts *Hops (p, r)=2*;

Si pour chaque paire de nœuds *p, q* de l'ensemble des nœuds du réseau *p et q* sont connectés alors le réseau est dit connecté

La notion Connecté *(G; H)* : est définie en se basant sur le maximum de nombre de sauts *H* entre les nœuds et un nœud spécifique (généralement le leader du groupe noté *l*) :

Connecté $(G; H) \Leftrightarrow \forall x, y \in G°$: *Hops* $(x; l) \leq H$

- *Taille*

La taille du groupe est définie comme le nombre maximum de membres du groupe (et on considère la taille maximal du group est *N*)

taille $(G; N) \Leftrightarrow |G| \leq N$

- *Energie*

Soit Energie(*x*, *t*) l'énergie disponible pour le nœud *x* à un moment donné *t*

On définit aussi *Emin* comme étant l'énergie minimale requise pour transmettre les données nécessaires avant épuisement de la batterie, on définit aussi *Emoy(x)* la vitesse moyenne de consommation d'énergie pour le nœud *x,* qui est un facteur de la charge suporté par le nœud

Emoy = (Energie(x, t) + Energie(x, t+1) + ……. + Energie(x, t+n-1))/n n >1

3. Services de gestion de groupe

Dans les réseaux mobiles ad hoc la configuration du réseau change à cause de la mobilité des nœuds, dans l'architecture suivante un nœud peut construire un groupe par la sélection des nœuds de son voisinage qui satisfait les critères requis

Nous définissons les fonctions du service de gestion de groupe comme suit :

Phase d'initialisation : consiste en un protocole qui utilise les primitives de la couche MAC pour trouver tous les nœuds qui sont dans le voisinage d'un nœud mobile. Le nœud mobile doit être assez fiable pour les sélectionner comme étant susceptibles de devenir membre du groupe

Phase de construction du groupe : échanges des informations nécessaires et applicables aux fonctionnalités et contraintes globales du groupe.

Phase de Maintien du groupe : mètre à jour les membres du groupe suivant l'évolution des nœuds et de la topologie du réseau

Nous considérons que chaque nœud du réseau est capable de connaitre son voisin direct et de détecter le départ de ses voisins directs et l'arrivée de nouveaux voisins directs à partir de la table de routage.

3.1.1. Notion clé de la solution proposée

Le concept clé de ce protocole est la notion de sécurité qui permet de déterminer si un nœud a le temps d'exécuter le changement de configuration avant que son éventuelle déconnexion ne se produise. Ce concept de sécurité utilise deux attributs : la distance de sécurité et l'énergie de sécurité.

La notion de sécurité est définie comme étant l'assurance qu'un nœud est capable d'établir une communication avec ses voisins avant son éventuel disparition du réseau soit à cause de son déplacement ou à cause de l'épuisement de ses ressources énergétiques.

Soient deux nœuds, la notion de sécurité est vérifiée si les deux nœuds ont un seuil énergétique défini comme l'énergie minimale *Emin* requise pour accomplir une communication avec succès. De plus, étant de même portée de transmission égale à *R*, la distance entre eux est dite de sécurité si elle est plus grand que le seuil $r(v, t , t')$ défini comme la distance maximale que peut parcourir un nœud dans le temps *t* que prend l'accomplissement d'une communication avec succès, en supposant que les deux nœuds se déplacent aléatoirement avec une vitesse ne dépassant pas v et le temps maximal d'un changement de configuration atomique est t' .

Les nœuds a et b ne peuvent plus continuer à exécuter une tâche commune s'ils veulent garantir la délivrance des messages car ils peuvent à tout moment sortir de la portée de transmissions de l'un et de l'autre,

La solution est d'accepter que les nœuds *a et b* soient dans le même groupe que s'ils ont assez d'énergie i.e. énergie $a < e$ et énergie $b < e$ (*e* étant la réserve d'énergie minimale requise pour adhérer au groupe) et assez proche l'un de l'autre, i.e., à une distance de l'un de l'autre égale à $r = R - 2v * (t + t')$.

La notion de sécurité est utilisée pour déterminer, si un nœud peut rejoindre le groupe et déterminer quand il doit se séparer de ce dernier, pour maintenir les exigences du service de gestion de groupe. Pour savoir si les membres du groupe sont en sécurité ou non, le leader du groupe maintient l'ensemble des positions des nœuds de son groupe et leur état énergétique où les membres du groupe communiquent leur position et leur état énergétique régulièrement au leader, qui vérifie constamment le respect vérification de la notion de sécurité et la présence de nouveaux nœuds dans la région.

3.2. Phase d'initialisation

Cette phase consiste en la collecte d'information sur le voisinage. Elle consiste en deux tâches :

3.2.1. Découverte du voisinage

La découverte du voisinage est une phase de collecte d'information sur les nœuds en utilisant toutes les informations présentes dans les couches de base du réseau mobile ad hoc tel que la couche Mac et les tables de routage.

La couche MAC fournit une communication point à point et les beacons. Ce qui permet de calculer les distances.

La table de routage fournit un état partiel des voisins, ce qui permet de connaître une bonne partie des nœuds existants dans le réseau.

3.2.2. Initialisation du groupe

Cette phase permet à un nœud *p* du réseau de connaître l'état temporaire du réseau qui contiendra le groupe *G* que p tente de créer et à priori l'ensemble des nœuds existant.

Le nœud *p* émet un message de découverte « *disc* » à ses voisins directs en utilisant un broadcast à deux sauts.

L'utilisation du broadcast à deux sauts est adopté pour réduire la surcharge du trafique. L'émetteur du broadcast en utilisant les informations de la phase d'initialisation sur les voisins et la topologie, sélectionne parmi les voisins à un saut le nombre minimal de nœuds qui peuvent couvrir tous les nœuds de moins de deux sauts et qui sont appelés les nœuds de broadcast. Seuls ces nœuds sélectionnés propagent le broadcast. Les boucles sont évitées en utilisant un numéro unique pour le message. Le but de ce message est de collecter les informations sur les nœuds présents dans le réseau qui sont susceptibles d'appartenir au groupe en vérifiant au fur et à mesure la notion de sécurité. Un ensemble *S* est ainsi formé

3.3. La phase de construction du groupe

La création d'un groupe *G* se traduit par la présence d'un nœud initiateur qui émet un message d'invitation « *init* » en utilisant un Broadcast de groupe à ces voisins appartenant à *S*.

Ce nœud est donc le leader temporaire du groupe. Le message « *init* » contient les attributs fonctionnels et non fonctionnels du groupe en création et son unique identifiant *id*. Par exemple, pour créer un groupe de tous les nœuds se trouvant à proximité d'un point fixé l'initiateur inclut cette référence dans le message « *init* ».

En recevant le message « *init* » un nœud vérifie la conformité des contraintes du groupe avec ces propres contraintes, s'il est intéressé. Entre temps s'il est un nœud de broadcast il doit déterminer s'il diffuse le message « *init* » selon la règle suivants : il ne diffuse pas le message « *init* » s'il n'a aucun voisin appartenant à l'ensemble *S* à part ceux de qui il a reçu le message ou informe le nœud dont il a reçus le message « *init* » que les nœuds découverts de son côté ne sont pas dans les contraintes du groupe.

Contrairement aux nœuds non de broadcast qui n'ont qu'à répondre en indiquant leur volonté d'adhérer au groupe, les nœuds de broadcast doivent attendre l'arrivée des réponses de tous leurs voisins avant de répondre à leur nœud précédent (un nœud de broadcast ou l'initiateur) avec un message d'adhésions « *join(G) »*. Ce message contient les nœuds voulant se joindre au groupe (le nœud émetteur inclut s'il veut adhérer au groupe) et les informations relatives aux nœuds tels que la position et le seuil énergétique, etc… . Et si le nœud de broadcast ne reçoit aucun message d'adhésion « *join(G) »* positive il répond par un message « *join(G »)* négatif pour l'indiquer. Si le nœud de broadcast ne reçoit pas le message d'adhésion *join(G)* positive dans un délai défini dans les contraintes globales du groupe G, considéré comme si, il a répondu par un message join (G*)* négatif.

Dans le cas ou plusieurs nœuds initiateurs de groupe de mêmes fonctionnalités lancent le message « *init* ». Au même moment, on peut les fusionner. Le nouveau leader est choisi parmi ces initiateurs, ce choix est systématique en prenant l'initiateur qui a le plus d'énergie.

Si l'initiateur tombe en panne durant cette procédure elle doit être refaite.

La construction effective du groupe ne commence que lorsque l'initiateur reçoit toutes les réponses de ses voisins. Une des parties importantes de l'initialisation est d'imposer les contraintes globales du groupe dont les plus importantes sont les suivantes :

La notion de sécurité : qui impose que chaque nœud doit être en mesure de satisfaire la notion de sécurité et qui se traduit dans l'énergie de sécurité pour chaque donné et la distance de sécurité entre deux nœuds voisins les uns des autres qui peut être calculée a partir du message « *join(G)* ».

La Réputation **:** qui impose que deux nœuds doivent avoir une bonne réputation entre eux. Etant très difficile d'imposer cette réputation entre les nœuds qui est confidentielle dans la plupart des cas, il est peu plausible de donner toutes les valeurs de réputation au leader. Elle peut être accompli en considérant que le leader est assez digne de confiance pour coordonnées les interactions entre membres du groupe, ou que les recommandations du leader sont assez forte pour fortifier la réputation des membres entre eux. Cette information sur la réputation peut être imposée localement durant la phase de découverte des membres du groupe. Cette réputation est assurée par l'utilisation de certificats.

La procédure de construction de groupe est complétée par un protocole de gestion de clé GKA établissant des clés partagées entre les membres du groupe. La clé est utilisée pour chiffrer tout les messages dans le groupe [08].

Taille : qui impose que le nombre total des membres du groupe ne peut dépasser un seuil déterminé .ceci peut se faire facilement en comptant le nombre de nœuds voulant joindre le groupe via le message « *join* ».

L'application de ces contraintes peut engendrer des exclusions de nœuds du groupe même s'ils ont envoyé le message favorable « *join* ». Ceci qui ce passe pour la contrainte d'énergie par exemple si le leader a jugé le nœud incapable d'assurer sa mission à cause de son état énergétique. Ces nœuds exclus peuvent former un autre groupe en exécutant une autre procédure de construction de groupe.

3.4. Phase de Maintien du groupe

Le maintien du groupe dans les réseaux ad hoc nécessite de prendre en compte les changements induits par la nature dynamique de ces réseaux : mobilité, manque de ressources, la non fiabilité des nœuds, et les liens de communication. Le leader du groupe doit introduire les changements dans la liste des membres du groupe, d'une manière à garder les changements transparent aux applications.

Une vérification périodique des contraintes et la notion de sécurité sur les membres du groupe est nécessaire pour le maintien du groupe.

Si un membre atteint les limites de sécurité, il est sollicité de quitter le groupe par le leader, en transmettant tout les données nécessaires à la bonne continuation du groupe sans pertes d'information. Une mise à jour de la liste des membres du groupe est systématiquement exécutée

Si un nouveau nœud est détecté par la couche réseau, le leader peut l'inviter à rejoindre le groupe, dans la condition ou le groupe est capable d'accepter plus de membres. Un message « *inv(G »)* est lancé par le leader au nouveau nœud détecté. En recevant le message « *inv(G)* » le nouveau nœud vérifie la conformité des contraintes du groupe avec ces propres contraints, s'il est intéressé, il repond avec un message d'adhésions « *join(G)* ». Le message contient l'information sur le nœud voulant adhérer au groupe telle que la position et le seuil énergétique. Si le leader ne reçoit le message d'adhésion « *join(G)* » positive dans un délai défini dans les contraintes globales du groupe *G,* il est considéré comme si le nouveau nœud détecté a répondus par un message « *join(G)* » négatif.

Le départ de nœud doit aussi être géré par le leader en envoyant un message de mise à jour de la liste de membre de groupe.

Comme c'est le cas de tous les nœuds, le leader peut quitter le groupe, il est nécessaire de faire passer le rôle du leader aux autres membres, aussi pour repartir la charge de mangement de groupe, la sélection du leader se fait selon les critères suivants :

- *La richesse en ressources*: les nœuds puissants avec des ressources abondantes sont considérées plus convenables pour devenirs des leaders. Cela est dû au faite que le leader consomme plus d'énergie et de temps de calcul que les autres membres du groupe. Les leaders faibles ont tendance être des obstacles. Les ressources suivantes sont prises en considération : charge du CPU ; la mémoire ; l'énergie et la bande passante.
- *Le temps écoulé depuis la derniers fois ou il été leader (elapsed time) :* ce facteur contribue a al distributions de charge sur les membres de groupe.

Le poids global est calculé comme suit, avec une impotence relative aux différentes applications définies par l'initiateur du groupe (ex. w_1, w_{11}) :

weight = Resources * w_1 + elapsed_time * w_2

$w_1 + w_2 = 1$

Resources * w_1 = chargedu cpu * w_{11} + memoire *w_{12} + energie*w_{13} + bande passante * w_{14}

$w1 = w_{11} + w_{12} + w_{13} + w_{14}$

Une liste de classement des leaders potentiels est dressée au niveau de chaque leader potentiel et qui est mise à jour chaque fois que la mise a jour de la liste des membres du groupe est effectuée.

La rotation du rôle de leader commence si le leader actuel se voit déclassé dans cette liste. Ceci est dû à multiple raisons. Le poids de leader a diminué à cause du temps qu'il a passé comme leader, ou ses ressources ont diminuées.

Le nombre de changement de leader dans un intervalle de temps donné est fixé dans les contraintes globales du groupe.

Le nouveau leader doit informer l'ensemble des membres du groupe des changements.

4. Simulations :

L'objectif majeur de notre simulation est d'évaluer le comportement de notre solution face à différents scenarios, taille du réseau, et la quantité de ressources d'énergie que pouvons-nous réserver l'aspect dynamique des réseaux mobiles ad hoc.

4.1. Environnement de simulation

Pratiquement, les environnements NS2 (Network Simulator 2) [77], Opnet [78] et GloMoSim [87] sont les plateformes de simulation de réseau les plus avancées et les plus utilisées par la communauté des chercheurs. NS2 est considéré actuellement l'outil de simulation le plus populaire destiné aux réseaux mobiles Ad hoc. Il contient l'implémentation de tous les protocoles proposés pour ce type de réseaux. Ces protocoles sont situés à différents niveaux de l'architecture. Il comprend principalement une implémentation fidèle à l'implémentation réelle et qui permet son extension de manière efficace. Aussi, l'utilisation du même environnement de simulation qu'on métrise le plus nous permet de valider notre implémentation notre solution et surtout de mener une bonne évaluation de la solution.

4.1.1. Le simulateur NS2 (Network Simulator2)

NS2 (Network Simulator2) est un simulateur à événements discrets développé par le projet VINT "Virtual InterNetworked Testbed" de l'Université de Californie – Berkely, il est destiné à simuler les

protocoles de communication des réseaux filaires et sans fil (locaux et satellitaires) tel que TCP, le routage, multi casting, multimédia, Mobile-IP, ...

Le simulateur NS2 est en libre accès à tous les chercheurs afin d'unifier les efforts et construire un simulateur de réseau aussi pratique que riche et varié. Nous utilisons la version 2.32 du package NS2 dans notre environnement de simulation.

4.1.2. Architecture de NS2

NS2 est implémenté en C++ et OTcl (Object oriented Tool Command Language) ce qui allie rapidité d'exécution et facilité de paramétrage des simulations. Un code C++ est compilé et donc rapide en exécution cependant difficilement modifiable, il est approprié pour l'implémentation des détails d'un protocole. Le code OTCL est interprété, par conséquent, lent en exécution mais facilement modifiable, il est approprié pour la configuration et le contrôle des scénarios de simulations.

Le package NS2 inclut une hiérarchie de classe compilée d'objets C++ et une hiérarchie de classe interprétée d'objets OTcl. Ces deux hiérarchies sont étroitement liées ; quand l'utilisateur crée un nouvel objet par l'interpréteur OTcl, un objet correspondant est aussi crée dans la hiérarchie compilée. Bien entendu, les objets peuvent être accédés aussi bien en OTcl qu'en C++ grâce à la mise en place de procédures d'appel entre OTcl et C++. Les trois classes les plus importantes pour établir cette correspondance entre les objets compilés et les objets interprétés sont:

La classe Tcl contient les méthodes que le code C++ utilise pour accéder et communiquer avec l'interpréteur. La classe Tcl encapsule l'instance en cours de l'interpréteur OTcl, appelle les procédures OTcl via l'interpréteur, recherche et retourne les résultats à l'interpréteur et surtout sauvegarde et interroge les objets « TclObjects ».

La classe TclObject est la classe de base pour tous les objets (interprétés et compilés) du simulateur. Un objet de TclObject est créé et configuré par l'utilisateur via l'interpréteur OTcl. Cette classe permet aussi de lier cinq types de variables (entier, réel, bande passante, temps, booléen) de la hiérarchie compilée avec ceux de la hiérarchie interprétée.

La classe TclClass est une classe très importante lors de l'implémentation d'un nouveau protocole. Elle offre deux fonctionnalités : (1) construire la hiérarchie interprétée pour correspondre à la hiérarchie compilée ; (2) offrir des méthodes pour instancier un nouveau objet TclObject.

Compte tenu de nos objectifs, nous avons bien sur essentiellement mis à contribution la partie IP/UDP/TCP en environnement sans fil. NS permet de positionner sur un plan virtuel des mobiles équipés d'émetteurs radio, et il gère la mobilité de ces nœuds dans le temps. Pour qu'un paquet émis sur une interface sans fil sous NS2 soit reçu, il faut qu'il arrive au destinataire avec un niveau de signal supérieur à un certain seuil. Ce seuil est par défaut de 3.652×10^{-10} W, et nous l'avons laissé à cette valeur pour toutes les simulations qui seront présentées par la suite.

Bien que le simulateur NS2 soit le plus utilisé par la communauté des chercheurs vu la diversité des protocoles qu'il simule, il est complexe en terme de code. Le nombre de composants du simulateur NS2 étant énorme on ne peut le représenter de manière globale. Dans le reste du document, nous nous concentrons sur les composants de NS2 nécessaires pour notre travail à savoir : les réseaux mobiles ad hoc et le modèle d'énergie adopté.

4.1.3. Les réseaux mobiles ad hoc dans NS2

Le simulateur NS2 propose une panoplie d'implémentations des réseaux mobiles ad hoc et cela en proposant un model de nœud mobile et la majorité des protocoles de routage standard propre aux réseaux mobiles ad hoc. Notre choix de protocole de routage étant AODV [80, 81, 19], car son implémentation nous permet de récupérer plus d'infos sur les nœuds du voisinage

L'utilisation de **802.11 DCF (**Distributed Coordination Function) [74] s'impose étant alliée au coût modéré Des équipements, elle joue primordial pour l'utilisation de 802.11 dans un contexte ad hoc. Elle peut en effet peut être utilisée directement sans aucune modification.

Notre choix se porte sur le modèle de propagation radio free-space [52].Ce modèle de propagation est le plus simple proposé par NS2. D'après ce modèle, la puissance reçue diminue de manière continue avec la distance. Comme la réception d'un paquet n'est possible que si la puissance est supérieure `a un certain seuil, nous obtenons logiquement une courbe où le débit est maximum jusqu''a une certaine distance (600 m) puis ensuite nul. Le seuil de communication est la valeur par défaut de NS2 ($3.652.10_{10}$ W).

Ce modèle de propagation représente les zones de communication comme un cercle autour de l'émetteur. Si un récepteur est dans ce cercle il reçoit tous les paquets, s'il est en dehors il n'en reçoit aucun. Ce qui justifie notre choix puisque notre solution se base sur le même model.

4.1.4. Le modèle d'énergie dans NS2

Le modèle d'énergie dans NS est un attribut de la classe *nœud.* Il représente le niveau d'énergie d'un nœud mobile. Il est caractérisé par une valeur initiale (initialEnergy_), une valeur décrémentée lors de chaque transmission (DecrTxPower_) et une valeur décrémentée suite à la réception d'un paquet (rxPower_). Les valeurs d'énergie requises pour l'émission et la réception dépendent de l'interface radio utilisée dans la simulation. Un nœud mobile peut aussi être dans l'état *idle.* La quantité d'énergie consommée dans ce mode est dépendante de la couche MAC (Medium Access Control) utilisée.

4.2. Implémentation de la solution dans NS

Les différents modules de notre protocole peuvent êtres implémentés sous forme classes, qui s'exécutent selon l'étape du protocole. Les agents *MobileNode* qui représente les nœuds du réseau font appel au classes constituant l'implémentation de notre protocole afin d'exécuter les différents étapes de ce dernier.

4.2.1. Class DecV

Cette classe permet de définir le module de détection de voisinage. Elle permet la mise à jour et la gestion de la liste des voisins à un saut et à deux sauts

4.2.2. Class Sec

Cette classe permet de définir le module de vitrification de la notion de sécurité pour chaque nœud à chaque fois qu'on lui fait appel.

4.2.3. Class Init

Permet au nœud voulant crée un groupe de faire une présélection des nœuds intéressant à contribuer au groupe

4.2.4. Class Leader

Permet de choisir le leader de groupe parmi les nœuds de présélectionné

4.2.5. Class Const

Permet de crée la liste des nœuds du groupe construit

4.2.6. Class Groupe

: permet de créé l'entité groupe qui stocke les informations propres au groupe

4.2.7. Class maint

Permet le maintien du groupe

4.3. Les éléments de la simulation

4.3.1. Les scénarios

Nous proposons trois différents scénarios dans le but d'observer le comportement de notre solution, cela pour mieux différencier les comportements liés à la connectivité des nœuds de ceux liée aux métriques de simulation ou de ceux qui sont due au hasard.

Les simulations sont effectuées dans un réseau ad hoc généré sur une surface carré de 10000*10000 mètre avec 1000 nœuds se déplacent aléatoirement avec des vitesses aléatoires. Les scenarios de mobilité simulé avec une vitesse de déplacement entre 0 et 5 mètre/second, dans les scenarios de faible mobilité et de 0 à 50 mètre/second dans le scenario avec une fortes mobilité, dans chaque scenario les rayons de transmission sont choisit entre 100 et 500 mètres.

Différents types de nœuds consomment l'énergie avec différents taux. La consommation d'énergie due au calcul interne des nœuds étant fixe, un nœud consomme 0.6% de son énergie totale chaque minute…. A chaque début de simulation les taux sont variables.

4.3.2. Génération des scénarios

La génération des mouvements des nœuds se fait par l'outil setdest avec les paramètres précédemment cités.

La génération du trafic se fait aléatoirement par l'outil tcpgen. En plus des messages de contrôle du protocole, la génération du trafique entre les membres se fait une fois le groupe est formé ainsi à chaque fois que la composition du groupe change.

Le temps de simulation	1000 secondes
Modèle de propagation	Free space
La couche MAC	La 802.11
Antennes	Omnidirectionnelles
Energie initiale	variable
Energie de transmission	0.666 W
Energie de réception	0.395 W
Energie dans l'état *idle*	0 W
Taille d'un message d'intérêt	100 (+25) octets
Taille d'un message de données	150 (+25) octets
Taille d'un message de donné entre membre du groupe	150 (+25) octets

4.3.1. Les paramètres de la simulation

Les paramètres utilisés dans notre simulation sont :

Nous avons utilisé des valeurs variables de la quantité d'énergie initiale au niveau des nœuds. Ceci est fait dans le but de refléter plus la réalité. Ceci nous permet d'examiner la durée de vie des différents nœuds dans le réseau en présence de petites quantités d'énergie et donc de considérer le partitionnement du réseau.

4.3.2. Les métriques

Nous avons choisi un ensemble de métriques pour évaluer le comportement de notre solution. Chacune des métriques est évaluée en fonction du nombre de noeuds déployés, de la charge de données, et de la densité des nœuds dans le réseau. Selon la valeur attribuée à la quantité d'énergie initiale au niveau des nœuds, nous proposons l'évaluation de deux ensembles de métriques :

4.3.3. 1. Le délai moyen

L'information échangée dans les réseaux mobiles ad hoc est souvent de type donnée en temps réel de coopération. Cette information est extrêmement critique, et le délai d'acheminement de l'information des nœuds sont souvent mis à des délais très serré sinon l'information devient caduque. Dans le cas des coopérations dans un groupe assurer un délai moyen de livraison de message est un critère de fiabilité. Pour cela nous comparons les délais moyens de transmission de données avant et après la formation du groupe. Cette métrique calcule la latence moyenne observée entre la transmission d'un message de donnée et sa réception.

4.3.4. 1. La fréquence de changement de leader

Cette métrique calcule la fréquence de changement de leader. Cette métrique a pour bute de montrer la stabilité de la solution quand les ressources réseaux viennent à manquer.

4.4. Résultats et interprétation des résultats

Nous allons présenter les résultats obtenus durant nos expériences de simulation. Nous Présentons le comportement de notre solution durant l'évolution des ressources dans le réseau à fur et à mesure du temps d'exécution des simulations.

4.4.1. Le délai moyen

La figure 5.1 présente les résultats relatifs aux délais moyens de notre solution dans les trois scenarios simulés. Les valeurs obtenues sont toujours comprises entre 0.1s et 1.4 ms tout le long de la durée des simulations. Ce si montre que notre solution montre un délai moyen raisonnablement acceptable et qu'elle fonctionne dans les délais raisonnable. Vers la fin des simulations les délais explosent, et cela est dû à l'épuisement des énergies des nœuds du réseau.

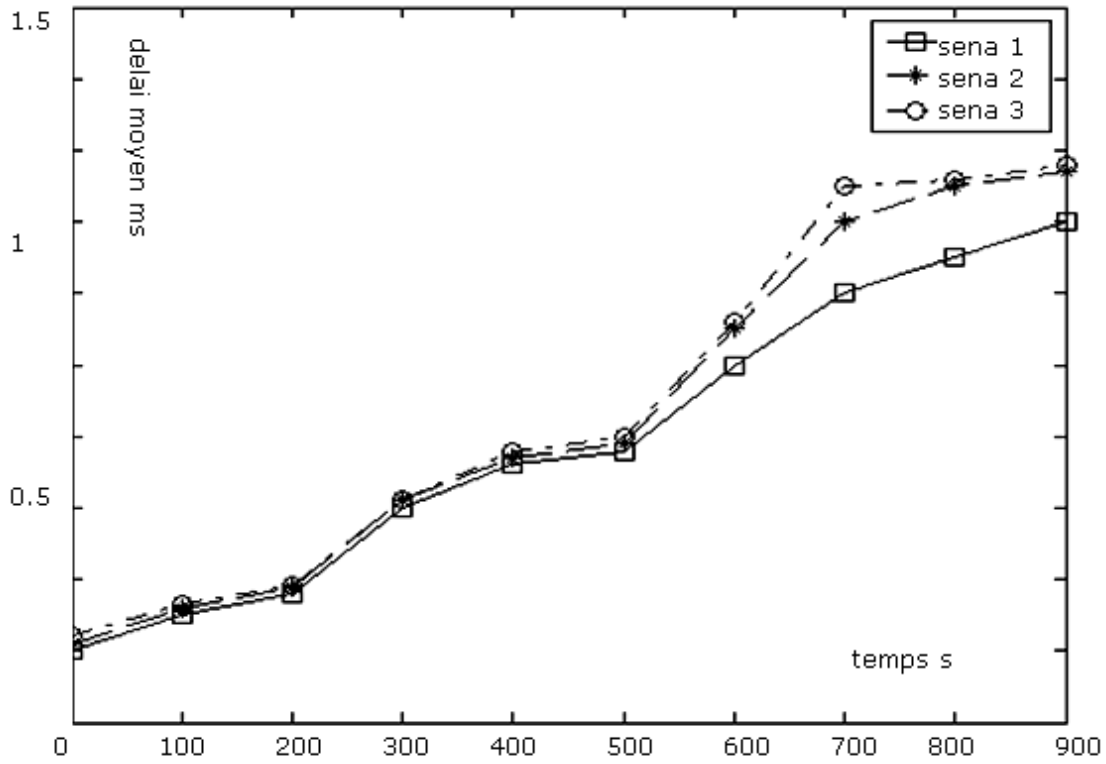

Figure 5.1 : Le délai moyen

4.4.2. . La fréquence de changement de leader

La figure 5.2 montre que la fréquence de changement de leader est stationnaire au début de la formation du groupe mais qu'elle explose vers la fin des simulations, ceci est due au fait que le critère de changement de leader se base essentiellement sur le niveau d'énergie des membre du

groupe.

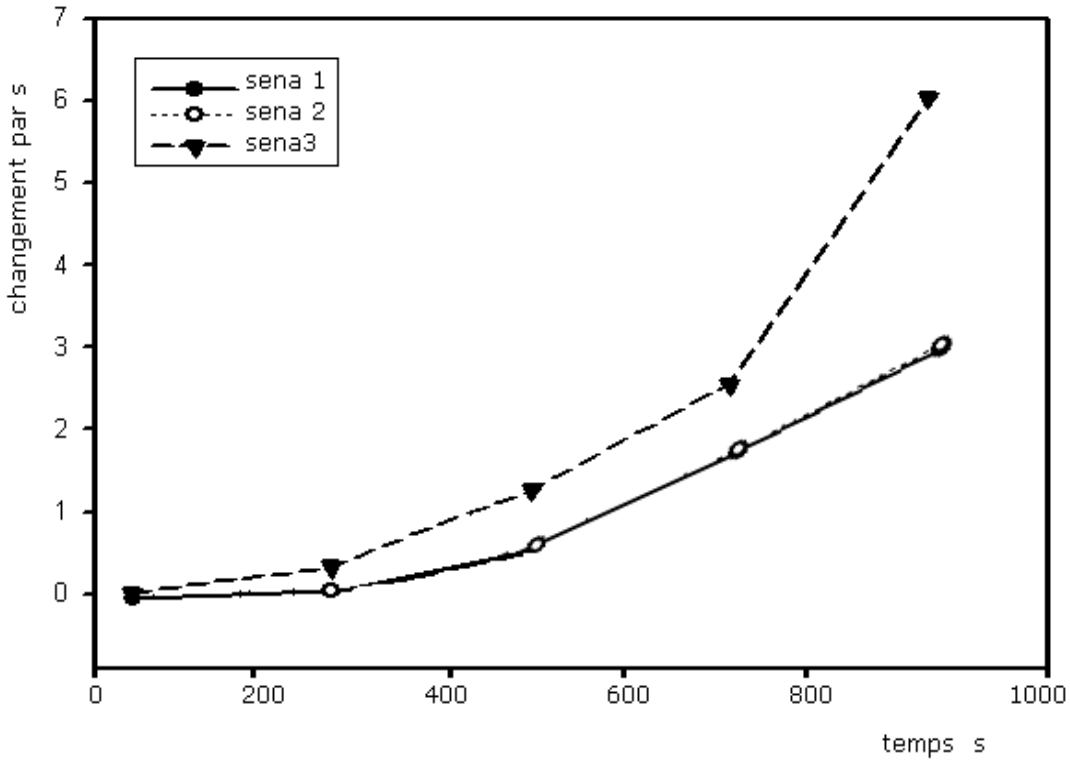

Figure 5.2 : fréquence de changement de leader

5. Conclusion

Le service gestion de groupe dans un réseau ad hoc où l'état du réseau peut changer à tout moment a fait l'objet de nombreux travaux dans la littérature. Toutes ces propositions existantes traitent des cas spécifiques des réseaux ad hoc, aucune des propositions existantes ne traite pas toutes les contraintes des réseaux mobiles ad hoc.

Dans ce chapitre nous avons proposé un nouveau service qui permet la transparence vis-à-vis des applications utilisant les groupes tout en maintenant le plus possible une vue constante du groupe et cela en traitant les contraintes des réseaux ad hoc sans restrictions, notre solution utilise l'information énergétique des nœuds dans tous les étapes du service de gestion de groupe chose inexistante dans les solutions proposées jusqu'à présent.

Nous avons monté par le biais des simulations le bon fonctionnement de notre solution dans les conditions normal et on a vue son comportement dans les cas extrêmes.

Conclusion Générale

Assurer la gestion de groupes dans un réseau mobile ad hoc est un problème très complexe vu l'aspect dynamique et l'évolution rapide de la topologie de ces réseaux. En effet, les unités mobiles sont dynamiquement et arbitrairement éparpillées d'une manière où l'interconnexion entre les nœuds du réseau peut changer à tout moment. Le but du service de gestion de groupe est de masquer aux applications utilisant les groupes les problèmes de changement de configuration du groupe.

Nous avons commencé par faire un état de l'art de solutions existantes dans la littérature. Nous avons remarqué qu'aucune solution existante ne prend en compte la contrainte d'énergie des nœuds du groupe dans un réseau mobile ad hoc. De plus, les approches existants génèrent un nombre excessif de messages de contrôles ce qui induit une consommation importante d'énergie.

Nous avons ensuite étudié les méthodes de préservation d'énergie dans les réseaux mobiles ad hoc et nous avons déduit que la préservation de l'énergie doit être présente dans toutes les applications et les services implémentés dans ces réseaux. Ceci nous a permis d'élaborer un service de gestion de groupe plus adapté aux réseaux mobiles ad hoc. En effet notre approche inclue l'état énergétique des nœuds dans toutes les actions du service de gestion de groupe. Ceci permet en particulier de gérer les déconnexions des nœuds et mieux répartir les tâches afin de prolonger la durée de vie du groupe. Nous avons surtout géré les déconnexions dues aux pannes sèches et qui entrainent inévitablement des pertes importantes des données et dans les cas extrêmes la dissociation des groupes et la perte totale de l'effort collectif effectué. En plus, notre solution permet de répartir les des membres du groupe selon leurs ressources effectives.

Nous avons fait une étude expérimentale de notre approche qui a montré le bon fonctionnement des cette solution

Références Bibliographiques

[01] Anceaume (E.), Charron-Bost (B.), Minet (P.) et Toueg (S.). – *On the Formal Specification of Group Membership Services*. – Rapport technique n° TR95-1534, Depto of Computer Science, Cornell University, aout 1995.

[02] Amir (Y.), Dolev (D.), Kramer (S.), Malkhi (D.). "Membership algorithms for multicast communication groups". In proc. of the 6th Intrl. Workshop on Distributed Algorithms (WDAG-6), pp. 292-312, Novembre, 1992.

[03] Khaldoune Al Agha and Laurent Viennot. "Spatial reuse in wireless LAN networks". *Rapport de recherche, réseaux et systèmes, Projet HIPERCOM (INRIA, France), 9 pages, Mai 2000.*

[04] E. ANIQUE, "Communication de groupe : service et protocole de transport à fiabilité intègre", *Ph.D. thesis*, Université Pierre et Marie Curie, Paris, France, December 1997.

[05] A. D. Amis, R. Prakash, T. H. P. Vuong, and D. T. Huynh. Max-Min D-cluster formation in wireless ad hoc networks. In Proc. of IEEE INFOCOM, 2000.

[06] Akis Spyropoulos and C.S. Raghavendra, "Energy Efficient Communications in Ad Hoc Networks Using Directional Antennas", *Department of Electrical Engineering-SystemsUniversity of Southern CaliforniaLos Angeles, IEEE INFOCOM*, juin 2002.

[07] S. Basagni. Distributed clustering for ad hoc networks. In Proc. of Int'l Symp. Parallel Architectures, Algorithms, and Networks, 1999.

[08] R. Bhaskar. Group key agreement in ad hoc networks. Technical Report 4832, INRIA-Rocquencourt, France, 2003.

[09] Nadjib Badache. "La mobilité dans les systèmes répartis". TSI. Technique et science informatiques ISSN 0752-4072 CODEN TTSIDJ 1998, vol. 17, no8, pp. 969-997 (2 p.1/4), Janvier 1998.

[10] Bartoli (A.) et Babaoglu (¨O .). – Selecting a primary partition in partitionable asynchronous distributed systems. *In : Proc. of the 16th Symposium on Reliable Distributed Systems*. – octobre 1997.

[11] Babaoglu (¨O .), Bartoli (A.) et Dini (G.). – Programming partition-aware network applications. *Lecture Notes in Computer Science*, vol. 1752, 2000, pp. 642–658.

[12] Babaoglu (¨O .), Bartoli (A.) et Dini (G.). – On programming with view synchrony. *In : ICDCS '96; Proceedings of the 16th International Conference on Distributed Computing Systems; May 27-30, Hong Kong*. pp. 3–10. – IEEE, mai 1996.

[13] Babaoglu (¨O .), Davoli (R.), Giachini (L.A.) et Baker (M.G.). – Relacs: A communications infrastructure for constructing reliable applications in large-scale distributed systems. *In :* 156 Bibliographie *Proc. of the 28th Hawaii International Conference on System Sciences*, pp. 612–621. – janvier 1995.

[14] Babaoglu (O.), Davoli (R.) et Montresor (A.). – *Group Communication in Partitionable Systems: Specification and Algorithms*. – Technical Report n° UBLCS-98-1, University of Bologna (Italy). Department of Computer Science., avril 1998.

[15] L. Briesemeister and G. Hommel. Localized group membership service for ad hoc networks. In Proc. Of the Int'l Workshop on Ad Hoc Networking, 2002.

[16] Boulkenafed (M), Issarny (V). Middleware service for mobile ad hoc data sharing, enhancing datan availability. *In Proc. of the 4th ACM/IFIP/USENIX Int'l Middleware Conf., June 2003.*

[17] Birman (K.). – *Reliable Distributed Computing with the ISIS Toolkit*, chap. Virtual Synchrony. – Los Alamitos, CA, IEEE Computer Society Press, 1994.

[18] Birman (K.), van Renesse (R.). " Reliable distributed computing with the isis toolkit ". chap. Virtual Synchrony. Los Alamitos, CA, IEEE Computer Society Press, 1994.

[19] J. Broch, D. Johnson and D. Maltz. "The dynamic source routing protocol for mobile ad hoc network". *IETF, Internet Draft, draft-ietf-manet-dsr-00.txt, March 1998.C.*

[20] Birman (K. P.), van Renesse (R.) et Vogels (W.). – Spinglass: Secure and scalable communications tools for mission-critical computing. *In : International Survivability Conference and Exposition. DARPA DISCEX-2001*. – Anaheim, California, juin 2001.

[21] M. Boulkenafed, D. Sacchetti, and V. Issarny. Using group management to tame mobile ad hoc networks. In Proc. of the IFIP TC8 Working Conf. on Mobile Info. Systems, 2004.

[22] C. A. Balanis “Antenna Theory: Analysis and Design” 2nd. New York: Wiley, 1997.

[23] M. Chatterjee and D. Das, S. K.and Turgut. WCA: A weighted clustering algorithm for mobile ad hoc networks. Cluster Computing, 5(2), 2002.

[24] Chandra (T. D.), Hadzilacos (V.), Toueg (S.) et Charron-Bost (B.). – On the impossibility of group membership. *In : Proceedings of the 15th Annual ACM Symposium on Principles of Distributed Computing (PODC’96)*. pp. 322–330. – New York, USA, mai 1996.

[25] Chockler (G. V.), Keidar (I.) et Vitenberg (R.). – Group communication specifications: a comprehensive study. *ACM Computing Surveys*, vol. 33, n° 4, décembre 2001, pp. 427–469.

[26] Chandy (K. M.) et Misra (J.). – How processes learn. *Distributed Computing*, vol. 1, 1986, pp. 40–52.

[27] Cristian (F.). – Reaching agreement on processor group membership in synchronous distributed systems. *Distributed Computing*, vol. 4, n° 4, avril 1991, pp. 175–187.

[28] Cristian (F.). “Synchronous and Asynchronous Group Communication,” *Communications of the ACM*, vol. 39, n° 4, avril 1996, pp. 88–97.

[29] Cristian (F.), Schmuck (F.), “Agreeing on Processor Group Membership in Timed Asynchronous Distributed Systems,” *Tech. Rep. CSE95-428, University of Califonia, san Diego, 1995.*

[30] A. Chandrakasan, T. Simon, J. Goodman and W. Rabiner,’’ Signal Processing for an ultra low power Wireless Video Camera”, *3rd International Workshop on Mobile Multimedia Communications, Princeton, NJ*, September 1996.

[31] Chandra (T.) et Toueg (S.). – Unreliable failure detectors for reliable distributed systems. *Journal of ACM*, vol. 43, n° 2, mars 1996, pp. 225–267.

[32] D. Djenouri; N. Badache, 'Optimisation de la consommation d'énergie pour le routage dans les réseaux mobiles ad hoc', Thèse de Magister, USTHB, Juillet 2003

[33] Nadjib Badache, Djamel Djenouri, Abdelouahid Derhab. "Mobility Impact on mobile ad hoc routing protocols", *ACS/IEEE International Conference on Computer Systems and Applications (AICCSA'03)*, Tunis 2003.

[34] Dolev (D.), Dwork (C.) et Stockmeyer (L.). – On the minimal synchronism needed for distributed consensus. *Journal of the ACM*, vol. 34, n° 1, janvier 1987, pp. 77–97.

[35] Défago (X.). – *Agreement-Related Problems: From Semi-Passive Replication to Totally Ordered Broadcast*. – Switzerland, Thèse de PhD, Ecole Polytechnique Fédérale de Lausanne, août 2000. Number 2229.

[36] F. Doughs, F. Kaashoek, B. Marsh, R. Caceres, K. Lai and J. Tauber, "Storage Altemativ~ for Mobile Computers", *Proc. 1994 Symposium on Operating Systems Design and Implementation, OSDI,* November 1994.

[37] Dwork (C.), Lynch (N.) et Stockmeyer (L.). – Consensus in the presence of partial synchrony. *Journal of the ACM*, vol. 35, n° 2, avril 1988, pp. 288–323.

[38] Dolev (D.) et Malki (D.). – The transis approach to high availability cluster communication. *Communications of the ACM*, vol. 39, n° 4, avril 1996, pp. 64–70.

[39] Dolev (D.), Malki (D.) et Strong (R.). – A framework for partitionable membership service. *In : Proceedings of the 15th Annual ACMSymposium on Principles of Distributed Computing (PODC '96)*. pp. 343–343. – New York, USA, mai 1996.

[40] Ezhilchelvan (P.), Macedo (R.) et Shrivastava (S.). – Newtop: A fault-tolerant group communication protocol. *In : Proc. of the 15th International Conference on Distributed Computing Systems*. – Vancouver, Canada, juin 1995.

[41] Ezhilchelvan (P. D.), Macêdo (R. A.) et Shrivastava (S. K.). – Newtop: A fault-tolerant group communication protocol. *In : Proceedings of the 15th International Conference on Distributed Computing Systems (ICDCS'95)*. pp. 296–306. – Los Alamitos, CA, USA, mai 30–juin 2 1995.

[42] Fetzer (C.) et Cristian (F.). – The timed asynchronous distributed system model. *IEEE Transactions on Parallel and Distributed Systems*, juin 1999, pp. 642–657.

[43] Fischer (M.), Lynch (N.) et Paterson (M.). – Impossibility of distributed consensus with one faulty process. *Journal of ACM*, vol. 32, n° 2, avril 1985, pp. 374–382.

[44] G.H Forman and J.Zahrojan "The challenge of mobile computing" IEEE Computer, 27 (4), pp38-47, April 1994

[45] Garbinato (B.). – *Protocol Objects & Patterns for Structuring Reliable Distributed Systems*. – Switzerland, Thèse de PhD, Ecole Polytechnique Fédérale de Lausanne, 1998.

[46] Guerraoui (R.). – Revisiting the relationship between non-blocking atomic commitment and consensus. *In : Proceedings of the 9th International Workshop on Distributed Algorithms (WDAG95)*, ed. par Helary (Jean-Michel) et Raynal (Michel). pp. 87–100. – Le Mont-Saint-Michel, France, 13–15 septembre 1995.

[47] Hayden (M.). – *The Ensemble System*. – Thèse de PhD, Cornell University, 1998.

[48] Hugues (B), Cahill (V). Towards real-time event-based communication in mobile ad hoc wireless networks. *In Proc. of the 2nd Int'l Workshop on Real-time LANs in the Internet Age, July 2003.*

[49] X. Hong, M. Gerla, G. Pei, and C. Chiang. A group mobility model for ad hoc wireless networks. In Proc. of the ACM MSWiM, 1999.

[50] Hickey (J.), Lynch (N.) et van Renesse (R.). – Specifications and proofs for ensemble layers. *Lecture Notes in Computer Science*, vol. 1579, 1999, pp. 119–133.

[51] Hadzilacos (V.) et Toueg (S.). – *Distributed Systems*, chap. Fault Tolerant Broadcasts and Related Problems, pp. 97–145. – Addison-Wesley, 1993.

[52] H.T. Friis. A note on a simple transmission formula. Proc. IRE, 34, 1946.

[53] Harris (E.P.), Warren (K.W.). "Low Power Technologies: A System Perspectives", *3rd I nternatinal Workshop on Mobile Multimedia Communications, Princeton, NJ*, Septembre 1996.

[54] ITU-T, INTERNATIONAL TELECOMMUNICATION UNION, "Multi-peer Communications Framework", *ITU-T Recommendation X.601*, March 2000.

[55] Jean-Pierre Ebert, Brian Burns, and Adam Wolisz, Technical University Berlin "A trace-based approach for determining the energy consumption of a WLAN network interface", T*elecommunications Networks Group Sekr. FT5-2 – Einsteinufer 25 – 10587 Berlin – Germany*, february 2002

[56] Kal (L.), Hadzilacos (V.). – Asynchronous group membership with oracles. *In : Proc. of the 13th Symp. on Distributed Computing - DISC*. pp. 87–100. – Springer-Verlag, 1999.

[57] Lamport (L.). – Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM*, vol. 21, n° 7, juillet 1978, pp. 558–565.

[58] J. Luo, P. Eugster, and J.-P. Hubaux. PILOT: Probabilistic lightweight group communication system for mobile ad hoc networks. IEEE transactions on mobile computing, 3(2), 2004.

[59] L. Buttyán, J-P. Hubaux "Stimulating Cooperation in Self-Organizing Mobile Ad Hoc Networks", *Technical Report No. DSC/2001/046, Swiss Federal Institution of Technology, Lausanne*, 31 August, 2001. http://icawww.epfl.ch/hubaux/

[60] J. Luo, J.-P. Hubaux, and P. Eugster. PAN: Providing reliable storage in mobile ad hoc networks with probabilistic quorum systems. In Proc. of the 4th ACM MobiHoc, 2003.

[61] J. Liu and V. Issarny. Enhanced reputatation mechanism for mobile ad hoc networks. In Proc. Of 2nd Int'l Conf. on Trust Management, 2004.

[62] Liu, J., Sailhan, F., Sacchetti, D., Issaeny, V., "Group Management for Mobile Ad Hoc Networks: Design, Implementation and Experiment", Proceedings of the 6th IEEE International Conference on Mobile Data Management (MDM'2005), May 2005, INRIA

[63] J. Liu and V. Issarny. QoS-aware service location in mobile ad hoc networks. In Proc. of the 5th IEEE Int'l Conf. on Mobile Data Management, 2004.

[64] K. Li, R. Kumpf, P. Horton and T. Anderson,"A Quantitative Analysis of Disk Drive Power Management in Portable Computers", *Proceedings 1994 USENIX, San Fiancisco, CA, pp. 279-291,* 1994.

[65] Lamport (L.), Shostak (R.) et Pease (M.). – The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, vol. 4, n° 3, juillet 1982, pp. 382–401.

[66] F. Li, R. Sutton, J. Rabaey, “Low Power Operating System for Hetrogeneous Wireless Communication Systems”, Workshop on Compilers and Operating Systems for Low Power 2001, September, 2001

[67] Mühlethaler Paul. 802.11 et les r´eseaux sans fil. Paris, Eyrolles, 2002. 281 p.

[68] Malloth (C.). – *Conception and Implementation of a Toolkit for Building Fault-Tolerant Distributed Applications in Large Scale Networks*. – Thèse de PhD, Ecole Polytechnique Fédérale de Lausanne, 1996.

[69] Moser (L.E.), Amir (Y.), Melliar-Smith (P.M), Agarwal (D.A.). " Extended virtual synchrony". In 14th Intl. Onf. On Distributed Computing Systems (ICDS), pp. 56-65, Juin, 1994.

[70] L. MATHY, G. LEDUC, O. BONAVENTURE, AND A. DANTHINE, "A Group Communication Framework", in *Proceedings of Broadband Islands '94*, Hamburg, Germany, June 1994, pp 167-178.

[71] R. Meier, M.-O. Killijian, R. Cunningham, and V. Cahill. Towards proximity group communication. In Proc. of the 1st Workshop on Middleware for Mobile Computing, 2001.

[72] Meissner (A), Musunoori (S. B) - Group integrity management support for mobile ad hoc communities. *In Proc. of the 1st Workshop on Middleware for Pervasive and Ad Hoc Computing, 2003.*

[73] Moser (L.E.), Melliar-Smith (P.M), Agarwal (D.A.), Budhia (R.K.) et Lingley-Papadopoulos (C.A.). – Totem: a fault-tolerant multicast group communication system. *Communications of the ACM*, vol. 39, n° 4, avril 1996, pp. 54–63.

[74] Mühlethaler Paul. 802.11 et les r´eseaux sans fil. Paris, Eyrolles, 2002. 281 p.

[75] Mishra (S.), Peterson (L.L.) et Schlichting (R.D.). – Consul: a communication substrate for fault-tolerant distributed programs. *Distributed Systems Engineering Journal*, vol. 1, n° 2, 1993, pp. 87–103.

[76] Malloth (C.) et Schiper (A.). – View synchronous communication in large scale systems. In : Proc. of the 2nd Open Workshop of the ESPRIT project BROADCAST (number 6360). – 1995.

[77] The Network Simulator - ns-2, ‘The *ns* Manual’, The VINT Project, collaboratoin between researchers at UC Berkeley, LBL, USC/ISI, and Xerox PARC, Kevin Fall and Kannan Varadhan Editors, 2003, http://www.isi.edu/nsnam/ns.

[78] OPNET Technologies, ‘ OPNET Modeler accelating networks R&D’, 2004, www.opnet.com

[79] R. Prakash and R. Baldoni. Architecture for Group Communication in Mobile Systems. In Proc. of the IEEE Symposium on Reliable Distributed Systems (SRDS), pages 235–242, October 1998.

[80] Charles. E. Perkins, Elizabeth M. Royer and Samir R. Das. "Ad hoc on demand distance Vector (AODV) routing". IETF, Internet Draft, draft-ietf-manet-aodv-05.txt, March 2000.

[81] Charles. E. Perkins. "Ad hoc on demand distance Vector (AODV) routing". IETF, Internet Draft, draft-ietf-manet-aodv-00.txt, November 1997.

[82] Charles. E. Perkins. "Ad hoc on demand distance Vector (AODV) routing". IETF, Internet Draft, draft-ietf-manet-aodv-00.txt, November 1997.

[83] Charles. E. Perkins and Elizabeth M. Royer. "Ad hoc on demand distance vector (AODV) algorithm". In Proceeding of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'99), New Orleans, Louisiana, USA, February 1999.

[84] Charles. E. Perkins and Elizabeth M. Royer. "Ad hoc on demand distance vector (AODV) algorithm". In Proceeding of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'99), New Orleans, Louisiana, USA, February 1999.

[85] Gruia-Catalin Roman, Qingfeng Huang, Ali Hazemi:, "On maintaining Group Membership Data in Ad Hoc Networks," Washington University, St Louis, Technical Report wucs-00-26, April 16, 2000.

[86] Gruia-Catalin Roman, Qingfeng Huang, Ali Hazemi: "Consistent Group Membership in Ad Hoc Networks". ICSE 2001: 381-388

[87] Ricciardi (A.). – *The Group Membership Problem in Asynchronous Systems*. – Cornell University, 1993.

[88] Raynal (M.) et Tronel (F.). – Group membership failure detection: a simple protocol and its probabilistic analysis. Distributed Systems Engineering, vol. 6, n° 3, septembre 1999, pp. 95–102.

[89] R. Friedman and R. van Renesse. Strong and Weak Virtual Synchrony in Horus. Tr95-1537, Cornel University, Department of Computer Science, August 1995.

[90] Sha (S). Chen (K), Nahrstedt (K). Dynamic bandwidth management for single-hop ad hoc wireless networks. *In Proc. of IEEE Int'l Conf. on Pervasive Computing (PerCom), 2003.*

[91] Sheetalkumar Doshi, Timothy X Brown "Design Considerations for an On-Demand Minimum Energy Routing Protocol for a Wireless Ad Hoc Network", *Department of Electrical Engg. University of Colorado at Boulder, ICC*, April 2002.

[92] Sabel (L. S.) et Marzullo (K.). – *Election Vs. Consensus in Asynchronous Systems*. – Rapport technique n° TR95-1488, Cornell University, Computer Science Department, février 1995.

[93] Suresh Singh, Mike Woo and C. S. Raghavendra. "Power-Aware Routing in Mobile Ad Hoc Networks", *Proceedings of the fourth annual ACM/IEEE international conference on Mobile computing and networking, p.181-190 Dallas, Texas, United States*, October 25-30, 1998,

[94] V. Tiwari, S. Malik, A. Wolfe, "Power analysis of embedded software: A first step towards software power minimizations". *IEEE Transactions on Very Large Scale Integration, 2(4): 437-445*, December 1994.

[95] S. Vasudevan, J. Kurose, and D. Towsley. Design and analysis of a leader election algorithm for mobile ad hoc networks. In Proc. of 12th IEEE ICNP, 2004.

[96] van Renesse (R.), Birman (K.) et Maffeis (S.). – Horus: a flexible group communication system. *Communications of the ACM*, vol. 39, n° 4, pp. 76–83, avril 1996.

[97] X. Zeng, R. Bagrodia and M. Gerla. "GloMoSim: A library for the parallel simulation of large-scale wireless networks", *Proceeding of the 12th Workshop on Parallel and Distributed Simulations, PADS'98*, May 1998.

[98] S. Zdonik, M. FranMin, R. Alonso and S. Acharya,''Are"disks in the aif' just pie in the sky?", i*EEE Workshop on Mobile Computing Systems and Applications, Santa Cruz, CA, pp. 12-19*, December 1994.

[99] Neiger (G.). – A new look at membership services (extended abstract). In : Proceedings of the 15th Annual ACM Symposium on Principles of Distributed Computing (PODC ’96). pp. 331–340. – New York, USA, mai 1996.