Localisation de ressources dans les réseaux ad hoc

Françoise Sailhan

Numéro d'ordre :
de la thèse

THÈSE

présentée

DEVANT L'UNIVERSITÉ PARIS VI

pour obtenir

le grade de : ***DOCTEUR DE L'UNIVERSITÉ PARIS VI***

mention : informatique

PAR

Françoise SAILHAN

Équipe d'accueil **: Projet Arles, INRIA**
École doctorale : **Informatique, télécommunication et électronique**
Composante universitaire : **LIP6**

TITRE DE LA THÈSE :

LOCALISATION DE RESSOURCES DANS LES RÉSEAUX AD HOC

Devant la commission d'examen

Pierre Sens, Président
Eric Fleury, Rapporteur
David Simplot-Ryl, Rapporteur
Titos Saridakis, Examinateur
Valérie Issarny, Directrice de thèse

Remerciements

Je souhaite remercier Pierre Sens, Professeur a l'Université Pierre et Marie Curie, pour m'avoir fait l'honneur d'accepter la position de Président du jury. Je remercie également David Simplot-Ryl, Directeur de recherche CNRS/INRIA, et Eric Fleury, Professeur a l'INSA, pour avoir accepter la charge de rapporteur et pour leurs remarques pertinentes concernant mon mémoire; ainsi que Titos Saridakis, chercheur chez Nokia, pour sa présence lors de ma soutenance.

Je tiens à témoigner ma sincère gratitude à ma directrice de thèse, Valérie Issarny, et ce pour de multiples raisons. Tout d'abord, je lui suis reconnaissante d'avoir dirigé mes travaux de recherche tout en me laissant libre d'explorer des pistes à ma guise. De plus, je la remercie pour ses conseils avisés tout au long de cette thèse ainsi que pour avoir été présente et motivante. En conclusion, que dire sinon que travailler à ses côtés a représenté une expérience plus qu'enrichissante.

Merci à l'ensemble de l'équipe *Arles* pour son accueil et sa bonne humeur.

Je remercie particulièrement Agnès, Sonya et Jean Luc pour leur participation active à la relecture de mon mémoire, Rafic pour son aide lors de l'implémentation du prototype et Emmanuelle pour son expertise administrative et sa disponibilité.

Je souhaite une excellente continuation aux actuels thésards, Ferda, JinShen, David et Sonya, ainsi qu'à ceux qui leur succéderont sous peu (vous êtes entre de bonnes mains).

Enfin, je tiens à exprimer tout spécialement ma reconnaissance à Ferda, compagnon de bureau durant ces années passées au sein de l'équipe Arles. Nos longues discussions, sa gentillesse, sa patience ont constitué pour moi une réelle richesse.

Je remercie ma deuxième famille, c'est-à-dire mes amis et toutes les personnes que j'ai pu rencontrer au cours de mon chemin, pour m'avoir donné le meilleur d'eux-mêmes. J'adresse ma profonde gratitude à Marion, amie des premières heures, avec qui j'ai traversé plusieurs hivers polaires canadien ainsi qu'à Charles et Sébastien pour leur accueil à Londres lors de la rédaction de cette thèse.

Je n'oublierai pas de saluer Onar, mon fidel ordinateur, qui fut mis a rude épreuve, et finit un matin pas rendre l'âme en explosant (partiellement tout de même) alors que je n'avais pas fais toutes les sauvegardes. Merci Matthieu d'avoir alors éteind l'incendie.

Je souhaite maintenant remercier du fond du coeur, les fidèles parmi les fidèles: ma famille, mes parents, mon frère Jean Maurice et ma soeur jumelle Florence. Merci pour leur amour indéfectible.

Enfin, et comme le disent si joliment les Québécois, je souhaite une bonne envolée à tous ceux qui ont été présents et ont participé, directement ou indirectement, à cette thèse.

À mes parents, mon frère Jean Maurice et ma sœur jumelle Florence.

Agathos._ La connaissance n'est pas une chose d'intuition, pas même ici. Quant à la sagesse, demande avec confiance aux anges qu'elle te soit accordée!.

Oinos._ Mais, pendant cette dernière existence, j'avais révé que j'arriverais d'un seul coup à la connaissance de toutes choses, et du même coup au bonheur absolu.

*Agathos.*_ Ah! ce n'est pas dans la science qu'est le bonheur mais dans l'acquisition de la science! Savoir pour toujours, c'est l'éternelle béatitude; mais tout savoir, ce serait une damnation de démon ...

Oinos._ Mais puisque chaque minute augmente notre connaissance, n'est-il pas inévitable que toutes choses ne nous soient connues à la fin?

*Agathos.*_ Plonge ton regard dans les lointains de l'abîme! Que ton œil s'efforce de pénétrer ces innombrables perspectives d'étoiles ... Il nous est révélé ici que l'unique destination de cet infini de matière est de fournir des sources infinies, où l' âme puisse soulager cette soif de connaître, qui est en elle, - inextinguible à jamais, puisque l'éteindre serait pour l' âme l'anéantissement de soi-même…. Viens! nous laisserons à gauche l'éclatante harmonie des Pléiades, et nous irons nous abattre loin de la foule dans les prairies étoilées, au-delà des d'Orion, où, au lieu des pensées, de violettes et de pensées sauvages, nous trouverons des couches de soleil triples et de soleils tricolores.

Puissance de la parole
Edgar Poe

Résumé

Le nombre croissant des terminaux (PDAs, ordinateurs portables, téléphones) et l'expansion des réseaux sans fil (GSM, GPRS, UMTS, WLAN, Bluetooth) témoignent de la demande croissante des utilisateurs pour un accès à des ressources telles que des données ou des services applicatifs, à tout moment, et, n'importe où. Dans ce contexte, les réseaux *ad hoc* peuvent jouer un rôle important puisqu'ils ne nécessitent pas d'infrastructures pour communiquer, ils peuvent également être utilisés lorsque le déploiement d'infrastructures est coûteux ou sans objet, ou encore pour accroître la portée des réseaux à base d'infrastructure. De plus ces réseaux *ad hoc* peuvent opérer de façon autonome ou être connectés à d'autres types de réseaux afin de fournir à l'utilisateur un accès vers un ensemble plus important de ressources. Dans les réseaux filaires ou sans fil basés sur des infrastructures, les solutions proposées afin de localiser les ressources se basent majoritairement sur un répertoire centralisant les informations relatives aux ressources du réseau. Ces solutions sont inexploitables dans les réseaux *ad hoc* en raison du manque de connectivité induit notamment par les changements de topologie du réseau, et par l'épuisement des réserves énergétiques des batteries. Les solutions proposées pour les réseaux *ad hoc* se basent quant à elle sur des techniques de diffusion coûteuses, en terme de bande passante, et conduisent à surcharger le réseau. Cette inefficacité des solutions existantes demeure l'un des freins majeurs au développement d'applications pour les réseaux sans fil intégrant des réseaux *ad hoc*. L'objectif de cette thèse est d'offrir une localisation efficace des ressources afin de garantir un accès transparent aux ressources disponibles dans un environnement intégrant des réseaux *ad hoc*. Tout d'abord, cette thèse présente les travaux de recherche portant sur la localisation des terminaux et des ressources dans un réseau *ad hoc*. Nous introduisons ensuite les techniques que nous proposons pour effectuer une localisation des ressources adaptée aux caractéristiques des réseaux *ad hoc* (e.g., densité des nœuds, surface du réseau), et cela afin de limiter le trafic généré par cette localisation. Ceci nous conduit notamment à aborder distinctement la localisation de ressources dans les réseaux *ad hoc* à petite et large échelle.

La première approche que nous proposons est entièrement distribuée. Les terminaux coopèrent ensemble afin de localiser des ressources sur le réseau. De plus, ce protocole inclut des techniques de caching et de préchargement afin d'améliorer les performances lors de la localisation et lors de l'accès aux ressources. Afin d'effectuer une localisation efficace des ressources dans un réseau *ad hoc* à grande échelle, nous introduisons une solution semi-

distribuée basée sur un ensemble de répertoire déployés dynamiquement sur le réseau *ad hoc* (ou possiblement hybride). Ces contributions sont complétées par une évaluation des performances incluant une étude théorique et pratique basée sur des simulations et des expérimentations. Enfin, nous présentons une mise en œuvre de notre protocole de localisation de ressources dans le contexte des services Web. Cet intergiciel supporte un accès ubiquitaire à des services Web mobiles en se basant sur une découverte dynamique des instances de services Web présentes dans l'environnement.

Abstract

Recent advances in wireless technologies, coupled with the concomitant abundance of portable devices, are opening up exciting opportunities for the future of wireless networking. In this context, MANETs (Mobile Ad hoc NETworks) can play an important role since they are autonomous, do no require any pre-existing infrastructure for communication purposes, can be conveniently deployed in locations where building a networking infrastructure is expensive or cumbersome, possibly to increase the reachability of infrastructure-based networks. In addition, *ad hoc* networks may operate whether in a standalone fashion, or be connected to other types of networks so as to provide users an access to a wide diversity of resources. However, MANETs are highly dynamic and suffer from frequent and unpredictable changes in the network topology due to the fact that devices are mobile and operate with low battery power. Considering these specific characteristics, it appears that existing approaches, mostly designed to operate in wired networks, cannot be adopted to locating and providing access to resources in pervasive computing environments. Indeed, these approaches are mostly centralized and are thus, unsuitable for infrastructure-less networks. This inefficiency, aggravated by the low connectivity and the mobility of nodes, remains a major obstacle in the development of applications for the wireless environment that integrates MANETs. Specifically, resource localization in pervasive computing environments has been an attractive area of research in wireless and/or mobile environments. However, proposed solutions are broadcast-based and result in overloading the constrained network. As a result, they do not solve the issue of scalability, mobility or adaptability. The issue addressed by this thesis is therefore to provide automatic and efficient resource localization and ubiquitous access to resources available in a pervasive computing environment integrating *ad hoc* network(s) with other interconnected network(s) (that can be infrastructure-based). This dissertation examines first the techniques and strategies that have been proposed in infrastructure-less environment. It then introduces a conceptual architecture to enable ubiquitous resource localization and access in infrastructure-less pervasive environments. This architecture is adapted to the characteristics of the *ad hoc* network (e.g., node density, network area). It leads us to tackle on separate way resource localization into small or scalable *ad hoc* networks. The first approach for resource localization is cooperative and fully distributed, and balances its

performance against its cost by conveniently balancing communication load among the devices. This protocol includes some energy-aware caching and prefetching strategies that improve the overall performance of the resource localization. In order to support efficient resource localization in a scalable *ad hoc* network, we then introduce a semi-distributed solution that is based on a set of directories dynamically deployed over the *ad hoc* (or possibly hybrid) network. In order to improve the performance of the proposed solution, we introduce several cross-layer optimizations.

The design of the resource discovery protocols is further complemented with extensive evaluation of performance, including both theoretical or/and practical studies based on simulation and implementation. Finally, we present the integration of our scalable resource discovery protocol in a middleware grounded on Web Service technologies. This middleware enables ubiquitous access to mobile Web services based on automatic discovery in dynamic pervasive computing environment.

Table des matières

Table des figures

Liste des tableaux

Chapitre 1 Introduction générale

Diverses technologies sans fil (par exemple, Bluetooth [12], IEEE 802.11 [43], HomeRF [57][1], HiperLAN [83][2]) ont été récemment proposées, dans le but de substituer les transmissions filaires par des ondes radio-électriques. Ces technologies sont adaptées à des contextes d'utilisation spécifiques et ont notamment donné naissance à deux types de réseaux : les réseaux personnels sans fil ou PAN (*Personal Area Network*) et les réseaux locaux sans fil ou WLAN (*Wireless Local Area Network*).

Un réseau personnel sans fil est principalement constitué d'équipements appartenant à un même utilisateur et distants de quelques mètres. La norme Bluetooth offre le meilleur exemple de technologie adaptée au PAN, en raison de la portée de transmission limitée d'un terminal Bluetooth et des modèles de communication qu'il définit. Par exemple, un réseau Bluetooth basique (ou *piconet*) prend en charge au maximum huit terminaux, un terminal étant désigné comme maître et les huit autres comme esclaves.

Un réseau local sans fil couvre le périmètre d'un réseau local, correspondant à celui d'un espace publique ou d'une entreprise. L'usage des technologies IEEE 802.11, HiperLAN et HomeRF est pertinent pour ce type de réseau puisque leur portée d'émission est de l'ordre de plusieurs centaines de mètres. Ces technologies utilisent deux méthodes différentes pour gérer la mobilité des utilisateurs. Certains réseaux, que nous appelons *réseaux à base d'infrastructures*, ne supportent que le mouvement des terminaux, le cœur du réseau restant fixe. L'infrastructure du réseau est composée de stations de base gérant la communication avec les terminaux présents dans une zone géographiquement limitée, appelée cellule, comme illustré par la figure 1-1.

[1] http://www.homerf.org
[2] http://portal.etsi.org/.

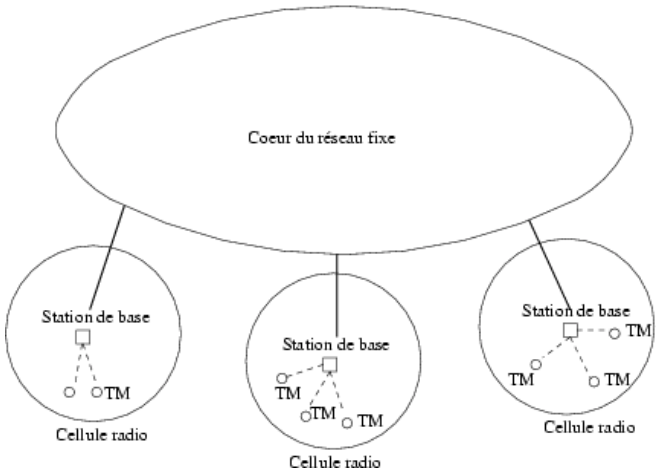


Figure 1-1 Modèle basé sur des infrastructures

D'autres réseaux, dits réseaux *ad hoc*, s'appuyant sur des technologies telles que IEEE 802.11 et HiperLAN, proposent une solution alternative afin de fournir une connectivité à des terminaux fixes ou mobiles qui demandent un déploiement rapide. Pour cela, ces réseaux sont composés de nœuds ne nécessitant pas la présence d'une infrastructure fixe pour communiquer. Les terminaux sans fil communiquent directement (de voisin à voisin). Les réseaux *ad hoc* peuvent exister de façon autonome. Ils peuvent aussi être connectés à d'autres types de réseaux sans fils ou filaire, à base d'infrastructure ou non, et former ainsi un réseau hybride.

Ces deux catégories de réseaux sans fil (réseaux à base d'infrastructure et réseaux *ad hoc*) partagent un certain nombre de problématiques communes, induites par les contraintes physiques inhérentes aux ondes radio-électriques : une faible bande passante et un faible taux de transmission (ou débit). Ces contraintes sont dues à :

- l'étroitesse de la plage de fréquence allouée par les Autorités de Régulation des Télécommunications (ART[3]) pour les réseaux radio-électriques,
- un canal de transmission susceptible d'être victime d'interférences, bruits ou parasites, pouvant affecter le signal,
- la distance entre deux terminaux sans fil qui constitue un vecteur non trivial d'atténuation et de distorsion du signal (selon la théorie du signal, l'atténuation est inversement proportionnelle à la distance séparant l'émetteur du récepteur),

[3] www.art-telecom.fr.

- l’impossibilité à contrôler la propagation des ondes dans une direction particulière, d’où une pollution du canal et des collisions fréquentes.

Par ailleurs, les réseaux sans fil sont en général caractérisés par l'hétérogénéité des terminaux interconnectés, tant au niveau matériel que logiciel. Ainsi, les capacités des terminaux sans fil varient de façon importante, les ressources mémoires et de stockage d'un téléphone, d'un PDA (*Personal Digital Assistant*) ou d'un ordinateur portable n'étant pas du même ordre. Toutefois, en raison de leur alimentation par batterie, ces terminaux présentent tous l'inconvénient d'avoir une autonomie énergétique réduite.

En plus de ces contraintes, les réseaux *ad hoc* doivent faire face à d'autres problématiques qui leur sont propres. Tout d'abord, les liens de communication dans les réseaux *ad hoc* sont asymétriques. Cela signifie que si un terminal reçoit un message d'un autre terminal, la réciproque n'est pas forcément vraie. Ensuite, l'accès au canal n'est pas contrôlé par une entité assurant la coordination des terminaux ou la synchronisation des communications. Par conséquent, tous les nœuds du réseau peuvent être victimes des interférences causées par l'émission de messages par les terminaux sans fil à portée de communication. Le taux d'erreur est donc accru tandis que le débit de transmission est restreint d'autant. Enfin, la topologie du réseau change avec la mobilité de ses terminaux mobiles.

Toutefois, malgré ces handicaps spécifiques, les réseaux *ad hoc* présentent des avantages indéniables, comme par exemple leur déploiement immédiat et leur faible coût d’utilisation du point de vue financier. Pour exploiter efficacement les possibilités des réseaux *ad hoc*, il reste à proposer des solutions permettant de surmonter leurs handicaps.

1.1 Motivations et problématiques

A l'heure actuelle, les réseaux *ad hoc* sont exploités dans des contextes d'application divers et des environnements variés. En effet, ils ont l'avantage de pouvoir être déployés rapidement et à faible coût puisqu'ils ne requièrent pas d'infrastructure fixe. Ils représentent ainsi une alternative intéressante dans de nombreuses situations dont les situations d'urgence (désastre naturel, tremblement de terre) impliquant la destruction totale ou partielle des infrastructures du réseau fixe. Ils sont aussi particulièrement utiles dans le cas d'applications militaires ou de sauvetage pour lesquelles il n'est pas envisageable d'installer des infrastructures. D'autre part,

leur utilisation est préconisée dans le contexte quotidien (réunions de travail ou collaboration fortuites de personnes, maison communicante), dès lors que les besoins ne justifient pas de déployer des infrastructures onéreuses.

Dans ces différents contextes d'application, un utilisateur a besoin d'accéder aux ressources numériques présentes dans le réseau *ad hoc*. Ces ressources numériques peuvent correspondre à des documents mis à disposition, par exemple lors d'une réunion de travail. Elles peuvent aussi désigner des ressources matérielles telles que des capteurs réalisant des prélèvements pour la domotique ou la surveillance des désastres naturels. Enfin, elles font référence à des services numériques effectuant des traitements et des calculs. C'est le cas par exemple d'un service numérique évaluant les pertes survenues sur une scène de désastre ou sur un champs de bataille et fournissant le résultat aux coordinateurs tout en proposant une stratégie.

Dans ces différents contextes d'application, un client mobile (qu'il s'agisse d'un utilisateur ou d'une application) doit pouvoir accéder, de façon transparente, c'est-à-dire, sans aucune intervention humaine, aux ressources numériques présentes dans le réseau *ad hoc*. Les protocoles de routage développés récemment permettent de répondre en partie à cette exigence en fournissant une connectivité multi-sauts et spontanée aux terminaux. Toutefois, pour qu'un client puisse accéder aux ressources disponibles, il doit aussi au préalable les localiser dans le réseau. Or, les solutions proposées pour localiser les ressources dans les réseaux à base d'infrastructure ne peuvent pas être appliquées directement dans les réseaux *ad hoc*. En effet, cette localisation est typiquement effectuée par un répertoire centralisant les informations relatives aux ressources du réseau. Dans un réseau *ad hoc*, cela conduit à surcharger une partie du réseau correspondant au voisinage du répertoire et à épuiser les batteries des terminaux s'y trouvant. De plus, cette localisation est particulièrement vulnérable aux déconnexions de cet unique répertoire.

Dans un réseau *ad hoc*, les approches préconisées pour localiser les ressources, se basent sur des techniques de diffusion. Cette localisation est soit effectuée par les terminaux mettant à disposition les ressources, qui assurent typiquement la diffusion des informations caractérisant les ressources (approche dite de *push*), soit pas le client qui interroge les nœuds constituant le réseau (approche dite de *pull*) en se basant également sur des techniques de diffusion. Dans le premier cas, garantir la disponibilité des ressources nécessite une forte fréquence de notification. De plus, l'augmentation du nombre de clients et/ou de terminaux mettant à

disposition des ressources accroît l'encombrement dans un réseau caractérisé par une bande passante étroite.

Il est nécessaire de proposer de nouvelles solutions afin de garantir une localisation efficace des ressources dans un réseau *ad hoc*, c'est-à-dire prenant en compte les contraintes caractéristiques des réseaux *ad hoc*, à savoir principalement le manque de connectivité, la bande passante réduite, et les capacités limitées des terminaux. De plus, les solutions proposées doivent fournir à l'utilisateur un niveau de qualité de service satisfaisant, ce qui signifie, en particulier, offrir un temps d'attente réduit pour l'utilisateur. Enfin, les réseaux *ad hoc* offrent des contextes d'exécution particulièrement riches notamment si l'on considère la présence d'autres réseaux auxquels ils sont connectés par des passerelles. Or, les solutions à la localisation de ressources existantes n'offrent pas, aux utilisateurs, la possibilité d'exploiter l'ensemble des ressources de l'environnement, ce qui inclut celles présentes sur d'autres types de réseaux.

1.2 Localisation efficace de ressources dans les réseaux ad hoc

L'objectif de cette thèse est d'assurer une localisation efficace des ressources dans un environnement sans fil *ad hoc*, c'est-à-dire s'adaptant dynamiquement aux changements survenant dans le réseau et prenant en compte ses contraintes, c'est-à-dire :

- l'absence d'infrastructures et les changements fréquents de topologie,
- les faibles capacités du réseau induites par le médium de communication utilisant des liens sans fil de type radio,
- les faibles capacités et l'hétérogénéité des terminaux,
- le manque de connectivité des terminaux.

Afin de limiter l'impact de chacune de ces contraintes, nous avons proposé des solutions dont le but est d'améliorer les performances, notamment en terme de temps de réponse et de trafic généré, lors de la localisation des ressources tout en prenant en compte les faibles capacités des terminaux constituant le réseau. D'autre part, ces solutions visent à augmenter la connectivité de l'utilisateur en lui permettant de localiser puis d'exploiter les ressources

disponibles, aussi bien dans le réseau *ad hoc* que dans les réseaux auquel ce dernier est connecté. Il en résulte trois contributions que nous introduisons ci-après.

Suivant les contextes d'application, les caractéristiques du réseau *ad hoc* varient. En effet, la densité des terminaux, le trafic transitant et la surface du réseau *ad hoc* ne sont pas du même ordre selon qu'il s'agisse d'un réseau créé dans le cadre d'une réunion, d'une application de secours, d'un événement sportif ou encore à l'échelle d'une ville. La localisation des ressources doit être adaptée aux caractéristiques du réseau *ad hoc,* ce qui inclut : la densité des nœuds, la superficie du réseau, le taux de mobilité des nœuds. Cela nous conduit à présenter deux approches pour gérer la localisation de ressources dans des réseaux *ad hoc,* respectivement destinées à un réseau à petite échelle et à un réseau à grande échelle.

Dans un environnement *ad hoc* de petite taille, c'est-à-dire de la taille d'un réseau local tel qu'un musée, une maison ou un campus universitaire, nous proposons une approche collaborative entièrement distribuée et cela afin de ne pas créer de goulet d'étranglement [99]. Plus précisément, les terminaux coopèrent entre eux afin de localiser les ressources dans le réseau *ad hoc*. Cette approche limite le trafic induit par la localisation d'une part en ciblant les hôtes vers lesquels sont dirigés les requêtes, et d'autre part en ne diffusant pas de requêtes de localisation dans l'ensemble du réseau. De plus, cette solution tire partie de la présence d'infrastructures, sans y avoir recours systématiquement, afin de faire un compromis entre le trafic généré sur le réseau *ad hoc* et sur les réseaux avec lesquels il est interconnecté. Cette solution prend en compte les capacités énergétiques limitées des terminaux grâce à une coopération permettant de sélectionner les terminaux à partir desquels sont accédées les ressources.

Dans un réseau *ad hoc* à large échelle caractérisé par une forte densité de terminaux et une surface importante, à l'échelle d'une opération militaire ou de secours ou encore d'une ville, nous proposons une approche semi-distribuée [97]. Plus précisément, notre solution se base sur des répertoires effectuant la localisation des ressources pour le compte des autres terminaux du réseau. Le déploiement dynamique et homogène de répertoires collaborant ensemble est effectué de façon à fournir une localisation efficace, en terme de temps de réponse et de trafic généré, des ressources dans l'ensemble du réseau [96]. Cette approche est particulièrement adaptée aux réseaux à large échelle puisque, d'une part, elle limite le nombre de terminaux interrogés et renvoyant des réponses, et d'autre part, elle ne se base pas sur des

techniques de diffusion pour localiser les ressources présentes dans l'environnement. De plus, cette solution permet la localisation de ressources dans le réseau *ad hoc* et dans les réseaux avec lesquels ce dernier est interconnecté. Notre solution peut enfin être déployée aussi bien dans un réseau *ad hoc* que dans un réseau à base d'infrastructures.

La localisation et l'accès aux ressources dans un réseau *ad hoc* soulèvent plusieurs problèmes. Les changements de topologie du réseau, la faible bande passante et le fort taux de collision et de retransmission sont à l'origine d'un temps d'attente important pour les utilisateurs. Cette attente est d'autant plus accrue lorsque les ressources correspondent à des contenus. Dans ce cas, le temps d'attente incluse le temps induit par la localisation des contenus et par leur rapatriement. Or, ce temps d'attente lorsqu'il est perçu par l'utilisateur affecte directement son niveau de satisfaction. Afin de limiter le temps d'attente de l'utilisateur, nous proposons d'utiliser des techniques de préchargement [99] et de *caching* [72] afin d'offrir aux utilisateurs une localisation et un accès efficace aux contenus. Un cache correspond à une zone de stockage utilisée pour conserver temporairement les contenus en vue d'une réutilisation. Un cache vise à réduire le temps d'attente de l'utilisateur lors de la localisation et de l'accès à un contenu. Trois avantages résultent de l'utilisation de caches. Tout d'abord, l'utilisation de la bande passante est réduite puisque moins de requêtes de localisation sont envoyées et moins de contenus sont échangés. De plus, le niveau de charge du fournisseur du contenu diminue puisqu'il doit répondre à moins de requêtes. Enfin, les déconnexions sont masquées aux utilisateurs et l'accessibilité aux contenus est accrue puisque les contenus sont disponibles et ce même si le fournisseur n'est plus joignable, ce dernier étant par exemple déconnecté ou en panne. Afin d'augmenter les performances du cache, c'est-à-dire d'augmenter le taux de réponses positives, nous proposons des techniques de préchargement, basées sur un algorithme de prédiction, et qui s'adaptent dynamiquement au contexte du réseau. Le préchargement consiste en effet à envoyer une requête pour localiser une ressource avant que l'utilisateur n'en fasse la demande.

Les contributions que nous venons d'introduire, ont été appliquées dans le contexte de la localisation et de l'accès Web [96] [98]. Le prototype de l'intergiciel Ariadne a été développé pour participer à l'essor des applications de l'informatique diffuse. Il s'appuie sur les standards du Web et s'exécute au dessus de réseaux MANETs. L'intergiciel Ariadne contient les composants effectuant à la fois la localisation et l'accès aux fonctionnalités offertes par un

service Web dans un réseau *ad hoc* mobile. Par ailleurs, cet intergiciel répond aux exigences fondamentales suivantes :

- il garantit un certain niveau d'interopérabilité entre les terminaux malgré leur hétérogénéité, aussi bien au niveau matériel que logiciel, en se basant sur les standards du Web,
- il répond aux exigences requises par les scénarios en permettant aux services de fournir des fonctionnalités complexes et à l'utilisateur d'effectuer des interactions avec les services Web.

La suite de ce document est constituée de six chapitres. Dans le chapitre 2, nous dressons un état de l'art sur la localisation de ressources dans un environnement *ad hoc*. Nous illustrons cet exposé par des travaux portant sur la localisation de terminaux et de ressources dans les réseaux *ad hoc* en mettant l'accent sur la consommation énergétique en résultant. Dans le chapitre 3, nous présentons notre protocole de localisation de ressources de type collaboratif entièrement distribué et adapté pour une utilisation dans un réseau *ad hoc*. Il permet aux utilisateurs de localiser de façon spontanée les ressources disponibles dans un environnement fortement dynamique. De plus, nous définissons une politique de gestion de cache et une stratégie de préchargement dans le but d'améliorer les performances. Nous proposons par ailleurs, une évaluation théorique du protocole. Il en résulte que ce dernier est particulièrement adapté aux réseaux *ad hoc* de faible taille, ses performances étant liées au nombre de terminaux composant le réseau. Dans le chapitre 4, nous proposons donc un protocole de localisation de contenus particulièrement efficace dans un réseau *ad hoc* de grande taille. Ce protocole se base sur un ensemble de répertoires déployés dynamiquement sur le réseau et assurant la localisation de contenus pour les utilisateurs. Dans le chapitre 5, nous mettons en œuvre cette proposition. Finalement, nous concluons ce document par un résumé de nos contributions et par une ouverture sur les perspectives en résultant.

Chapitre 2 Réseaux ad hoc

2.1 Introduction

Dans un réseau *ad hoc* sans fil, les terminaux, éventuellement mobiles, peuvent communiquer sans requérir la présence d'une infrastructure particulière lorsqu'ils sont à portée d'émission et de réception l'un de l'autre. Les réseaux *ad hoc* représentent donc une alternative intéressante aux réseaux à infrastructures, notamment pour les applications distribuées qui sont dynamiquement déployées, comme par exemple les applications dans le domaine militaire. Toutefois, de nombreux défis doivent être relevés pour que les réseaux *ad hoc* puissent être effectivement exploités par les utilisateurs, pour le partage et l'accès aux *ressources numériques*[4] offertes par les terminaux composant le réseau. Un premier défi est lié aux caractéristiques intrinsèques des réseaux sans fil : une communication dans un réseau *ad hoc* est immédiatement interrompue dès lors que les terminaux impliqués se trouvent hors de leur portée radio respective, ce qui peut être fréquent dans le cas de terminaux fortement mobiles. Cette dynamique du réseau soulève le problème de l'intégration dans différents réseaux. En particulier, un utilisateur nomade, du fait de sa mobilité, est souvent amené à rencontrer de nouveaux réseaux. Mais, il ne peut pas *a priori* accéder à leurs ressources numériques puisqu'il ne les connaît pas. De la même manière, un terminal ne peut pas interagir avec des terminaux ciblés s'il ne connaît pas leur identifiant. Or, un terminal ne peut pas compter sur les infrastructures sous-jacentes du réseau pour lui fournir ces informations ou pour relayer ses messages.

De manière générale, l'exploitation effective des réseaux *ad hoc* est rendue difficile par les spécificités de ces réseaux, qui sont :

- l'étroitesse et la diminution de la bande passante allouée du fait de la formation de goulets d'étranglement occasionnés par le partage du canal de communication,
- les changements de topologie du réseau, causés notamment par des déconnexions volontaires ou involontaires des utilisateurs, ainsi que par des obstacles et des interférences dans les liaisons radio, qui perturbent les communications.

[4] Nous employons le terme *ressource numérique*, ou plus simplement *ressource*, pour dénoter les entités du réseau qui peuvent être accédées et partagées par les différents terminaux le constituant. Une ressource peut ainsi correspondre a un contenu (multimédia), un service logiciel ou une ressource matérielle.

- L'utilisation des batteries en tant que source d'énergie par les terminaux, qui affecte la durée de vie du réseau.

Dans ce chapitre, nous présentons les différentes solutions apportées dans la littérature pour effectuer une gestion dynamique des réseaux *ad hoc* répondant aux défits soulevés par ces derniers et adaptées leurs spécificités. Plus précisément, nous définissons de quelle façon est assurée la localisation dynamique des terminaux et ressources associées dans l'environnement (§ 2.2). Puis, nous introduisons les mécanismes utilisés afin de réduire la déperdition énergétique résultant de la communication, qu'elle relève de l'accès au canal de communication, de l'envoi de messages, ou de la localisation de ressources (§ 2.3). Finalement, nous concluons ce chapitre par une synthèse des travaux existant et par une présentation de notre contribution (§ 2.4).

2.2 Localisation de ressources en environnement ad hoc

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

- une étape de localisation des terminaux qui rend possible l'accès aux ressources mises à disposition par des terminaux distants, et
- une étape de localisation des ressources numériques.

Nous détaillons ces deux étapes dans les deux sections suivantes.

2.2.1 Protocoles de routage pour la localisation de terminaux

Dans un réseau *ad hoc*, la localisation de terminaux permet d'augmenter la connectivité malgré une portée de transmission limitée. Elle est effectuée par les protocoles de routage, également appelés protocoles multi-sauts. La fonction principale de ces protocoles de routage est de fournir le chemin le plus court, en terme nombre de sauts, entre une source et une destination de telle façon que le nombre de messages générés soit minimum. Ces protocoles de routage se décomposent en quatre familles suivant qu'ils soient *proactifs*, *réactifs*, *hybrides* ou *géographiques*. Les protocoles *proactifs*, présentés dans la section 2.2.1.1, maintiennent à jour les tables de routage en recherchant régulièrement des routes. A l'inverse, les protocoles *réactifs*, aussi appelés protocoles à la demande (*on demand*), traités dans la section 2.2.1.2, sont caractérisés par le fait qu'une route n'est recherchée que lorsqu'un message doit être envoyé vers une destination. Les protocoles *hybrides*, abordés dans la section 2.2.1.3, utilisent à la fois une approche réactive et proactive. Enfin, les protocoles *géographiques*, présentés dans la section 2.2.1.4, enrichissent les protocoles précédents en incluant des informations géographiques dans les tables de routage qu'ils maintiennent.

2.2.1.1 Protocoles de routage proactifs

Afin de prendre en compte les changements de topologie du réseau *ad hoc*, les protocoles de routage proactifs échangent, à intervalle de temps régulier, des messages de contrôle destinés à la maintenance des tables de routage. Le protocole DSDV (*Destination-Sequenced Distance Vector*) [64] est l'un des premiers protocoles mis au point par le groupe MANET. Il s'agit d'un protocole orienté destination (plus connu sous le nom de *distance vector protocol*) basé sur l'algorithme de Bellman-Ford [9*,29,10*]. Ce type de protocole suppose que tous les nœuds du réseau disséminent une copie de leur vecteur de distance (*distance vector*), c'est-à-dire, de leur table de routage. Le vecteur de distance associé à un nœud du réseau indique les autres nœuds du réseau qui lui sont accessibles et le nombre de sauts nécessaires pour atteindre ces nœuds.

Qualifions de *N-voisins* d'un nœud du réseau, les nœuds qui lui sont distants de *N* sauts, et de *N-voisinage*, l'ensemble de ces nœuds. Dans le cas particulier où *N*=1, les terminaux *1-voisins* sont à portée de transmission du nœud considéré.

Chaque nœud (ou routeur) DSDV diffuse périodiquement une copie de son vecteur de distance à ses *1-voisins*. Lors de la réception d'un vecteur de distance, le nœud récepteur compare le coût (exprimé en nombre de sauts) nécessaire pour atteindre chaque nœud donné par le vecteur de distance reçu, avec celui défini dans sa propre table de routage. Si pour un nœud donné, le coût fourni par le vecteur de distance reçu est inférieur à celui de la table de routage locale, alors, la table de routage est mise à jour en conséquence. Chaque modification survenant dans la table de routage est ensuite publiée par le nœud lorsqu'il diffuse son propre vecteur de distance. Le principal défaut de DSDV réside dans la lente convergence des tables de routage. Ce problème survient pour deux raisons, d'une part les échanges de vecteurs de distance ne sont pas synchronisés, et d'autre part une route moins coûteuse est remplacée dans la table de routage par une route plus coûteuse, si et seulement si, le nœud ayant publié les deux routes est le même.

A titre d'illustration, prenons la situation suivante dans laquelle les tables de routage convergent lentement. Le réseau est constitué de trois nœuds *A*, *B* et *C* tels qu'il existe un lien de communication bidirectionnel entre *A* et *B* et entre *B* et *C*. Lorsque le nœud *A* reçoit le vecteur de distance de *B*, il met à jour sa table de routage; *C* devient alors accessible par *A* en deux sauts. Si le lien bidirectionnel se rompt entre *B* et *C*, alors *B* expulse de sa table de routage la route vers *C*. Cependant, lorsque *B* reçoit le vecteur de distance envoyé par *A*, il met à jour sa table de routage en stipulant que *C* est accessible en 3 sauts, alors que *C* n'est plus accessible. Lorsque *A* reçoit le vecteur de distance envoyé par *B*, *A* modifie sa table de routage de telle sorte que *C* est maintenant accessible en 4 sauts. Concrètement, une boucle, à l'origine du problème de la lente convergence des tables de routage de *A* et *B*, a été créée entre *A* et *B*. Ce défaut de DSDV est hérité du protocole RIP (*Routing Information Protocol*) [54] dont il s'inspire, et qui est utilisé pour le routage dans les réseaux filaires.

Afin de résoudre le problème précédent, les protocoles OLSR (*Optimized Link State Routing*) [23] et TBRPF (*Topology dissemination Based on Reverse-Path Forwarding*) [60] ont été proposés. Ces protocoles se basent sur le protocole OSPF (*Open Shortest Path First*) [56], qui résout le problème de la non convergence des tables de routage. Les protocoles OLSR et TBRPF, orientés topologie (*link state*), sont également qualifiés de protocole SPF (*Shortest Path First*). Cette classe de protocole se distingue par le fait que les routeurs collectent des informations relatives à l'état des liens. Ainsi, chaque nœud peut maintenir un arbre de routage à partir de lui-même et l'utiliser pour acheminer les messages de données vers une

destination. Cette approche nécessite que chaque routeur maintienne à jour une vue du réseau et que chaque routeur (cas du protocole TBRPF), ou un sous-ensemble d'entre eux (cas du protocole OLSR) collecte l'état des liens et les diffuse. La collecte d'informations sur l'état des liens dans le voisinage est réalisée par chaque nœud grâce aux messages *hello* diffusés périodiquement par les 1-voisins. Toutefois, afin que tous les nœuds puissent avoir une vue globale du réseau, ces informations collectées doivent être diffusées sur le réseau. Les protocoles OLSR et TBRPF diffèrent au niveau des techniques utilisées pour disséminer ces informations. Le protocole OLSR définit un sous-ensemble *C* de nœuds appelés *relais multi-points* (ou MPRs pour *Multi-Point Relay*), qui sont responsables de la diffusion des messages de contrôle. Ce sous-ensemble *C* est sélectionné par chaque nœud *i* parmi ses 1-voisins de telle sorte que *C* couvre[5] tous les nœuds pouvant être atteints en deux sauts par *i* (voir figure 2-1). Chaque nœud *i* annonce périodiquement à ses 1-voisins, les nœuds qu'il a désignés comme MPRs, lesquels sont responsables de la dissémination dans le réseau, de l'état des liens avec les nœuds 1-voisins.

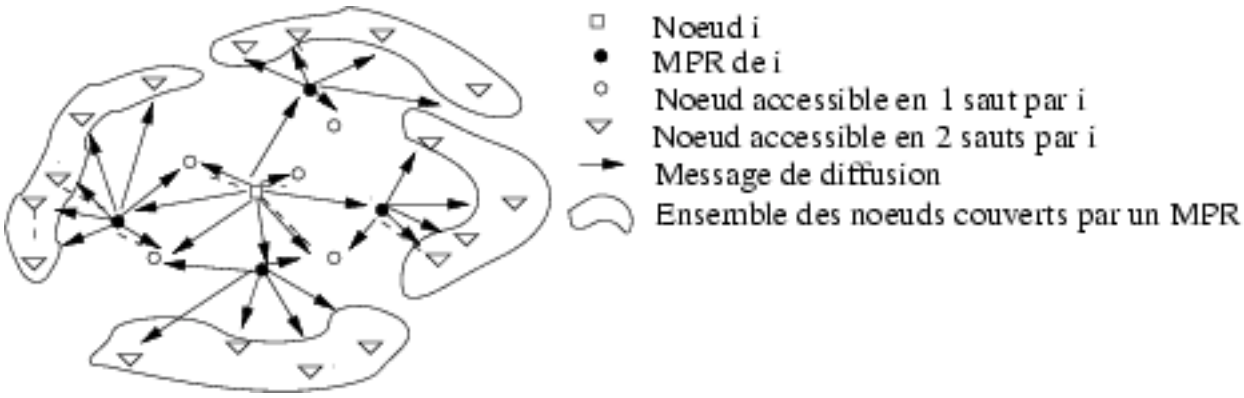
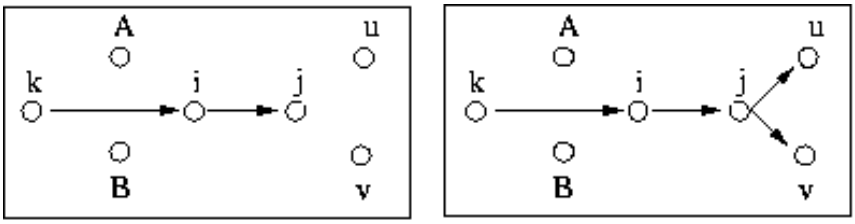


Figure 2-1 Relais multi-points du protocole OLSR

Contrairement à OLSR pour lequel seul les nœuds MPRs diffusent l'état des liens, avec TRBPF, chaque nœud diffuse aux nœuds 1-voisins l'état des liens du réseau. Plus précisément, chaque nœud extrait une vue partielle (ou un sous-arbre), à partir de la vue générale du réseau caractérisée par son arbre de routage. Cet arbre partiel est généré de la façon suivante: un nœud *i* inclut dans son arbre partiel un nœud *j* s'il détermine que l'un de ses 1-voisins *k* l'utilise comme nœud intermédiaire pour atteindre *j*. La F igure 2-2 (a) illustre ce choix : le nœud *k* peut utiliser le chemin *k-A-i-j*, *k-B-i-j* ou *k-i-j* pour atteindre le nœud *j*. Le chemin le plus court (en nombre de sauts) correspond à *k-i-j*. Par conséquent, le nœud *i* sait

[5] En d'autres termes, lorsque les nœuds MPRs du nœud *i* rediffusent un message diffusé par *i*, tous les nœuds 2-voisins de *i* reçoivent le message.

qu'il est utilisé comme nœud intermédiaire par le nœud *k* pour atteindre *j*. Après avoir inclus ses 1-voisins dans son arbre partiel, le nœud *i* inclut les nœuds pouvant être atteints par les nœuds déjà inclus dans l'arbre partiel. La F igure 2-2 (b) illustre cette étape; les nœuds *u* et *v* peuvent être atteints par le nœud *j* déjà inclus dans l'arbre partiel.



(a) **(b)**

F igure 2-2 Sélection des nœuds à insérer dans un arbre partiel

Les nœuds inclus dans l'arbre partiel du protocole TBRPF correspondent aux nœuds MPRs du protocole OLSR. Cependant, avec TBRPF, un nœud se définit lui-même comme relais, alors qu'avec OLSR, un nœud relais (MPR) est défini par ses voisins. Le concept fondamental de ces deux protocoles est que seul un sous-ensemble de nœuds (les nœuds relais) est habilité à *acheminer* les messages de données dans le réseau.

Les protocoles OLSR et TBRPF utilisent deux approches différentes pour restreindre le trafic généré par l'envoi de messages de contrôle. OLSR réduit le trafic généré par les messages de contrôle puisque seuls les nœuds MPRs sont habilités à envoyer des messages de contrôle stipulant l'état des liens entre le nœud lui-même et ses nœuds 1-voisins. En revanche, le protocole TBRPF permet à tous les nœuds d'envoyer des messages de contrôle. Afin de restreindre le trafic généré par les messages de contrôle, TBRPF utilise deux sortes de messages de contrôle, ayant des fréquences d'envoi différentes : chaque nœud diffuse périodiquement (par exemple, toutes les 5 secondes) son arbre partiel et plus fréquemment (par exemple, toutes les secondes) uniquement les changements intervenus dans l'arbre partiel. Ceci permet notemment d'informer rapidement les nœuds voisins qui sont les plus affectés par un changement de topologie locale, lors de la modification de l'état d'un lien.

Le principal inconvénient des protocoles proactifs est leur coût en terme d'utilisation de la bande passante, qui est induit par l'échange continuel de messages de contrôle nécessaires à la maintenance des tables de routage. Ce coût est disproportionné lorsque les nœuds échangent peu de messages de données ou si les changements de topologie sont peu fréquents.

2.2.1.2 Protocoles de routage réactifs

Les protocoles réactifs, également appelés protocoles à la demande (*on demand*), génèrent *a priori* moins de messages de contrôle que les protocoles proactifs, puisqu'ils ne maintiennent pas à jour leurs tables de routage. Ainsi, les informations contenues dans les tables de routage, n'étant pas rafraîchies périodiquement, ne sont utilisables que pendant un temps limité (déterminé par la durée de vie d'une route), ou dans le pire des cas, une seule fois. Le processus de localisation d'un terminal se décompose par conséquent en deux étapes importantes : la première étape correspond à la diffusion d'une requête de localisation sur le réseau, et la seconde consiste à collecter des réponses reçues. Plusieurs protocoles réactifs ont été proposés pour différents contextes d'utilisation. Nous proposons un aperçu des protocoles les plus connus en définissant dans quels contextes d'applications ils s'avèrent être particulièrement performants.

Le protocole TORA (*Temporary-Ordered Routing Algorithm*) [62] est un protocole *initié à la source* (*source initiated*), développé prioritairement pour des réseaux *ad hoc* à forte mobilité. Afin de supporter un nombre important de suppressions de liens induites par la forte mobilité des nœuds, plusieurs chemins potentiels, entre une source *i* et une destination, sont calculés. De plus, afin de réduire le trafic de contrôle résultant de la suppression d'un lien, ce protocole notifie rapidement les changements aux nœuds 1-voisins, et propage périodiquement ces changements au reste du réseau, mais à une faible fréquence. Le protocole TORA détermine les chemins entre une source et une destination en créant un arbre acyclique dirigé (*directed acyclic graph*) entre la source et la destination. Cet arbre est crée en utilisant un algorithme de création de route basé sur une pondération des nœuds. Plus précisément, le poids de chaque nœud est égal à la distance séparant le nœud source du nœud destination, la destination ayant un poids nul. Cet arbre est créé par un nœud source *i*, lors de l'étape de localisation par *i* de la façon suivante. Le nœud *i* diffuse un message de localisation sur le réseau. Un nœud intermédiaire recevant le message de requête de localisation s'affecte un poids non déterminé (NULL). Ce message est ensuite rediffusé par les nœuds intermédiaires le recevant jusqu'à ce qu'il atteigne un nœud 1-voisin de la destination. Un nœud 1-voisin de la destination diffuse alors une réponse en assignant un poids égal à 1 au message. Un nœud recevant une réponse ajoute à la liste de ces 1-voisins ayant déjà répondus (cette liste étant initialement vide) l'identifiant du nœud intermédiaire ayant envoyé ce message et le poids de ce message. Puis, il vérifie si le poids minimum de tous les voisins ayant déjà répondus est supérieur à celui du

message reçu. Si tel est le cas, il rediffuse le message en incrémentant le poids du message de 1 et en s'affectant comme poids celui du message. Au contraire, si le poids minimum de tous les voisins ayant déjà répondu est inférieur à celui du message, il assigne au message de réponse le plus petit poids de tous les voisins ayant répondu, en l'incrémentant de 1 puis il rediffuse le message. Ainsi, chaque nœud intermédiaire ayant fait suivre un message de réponse c'est attribué un poids correspondant à la distance le séparant de la destination.

Le protocole ABR (*Associativity-Based long-lived Routing* protocol) [78] est adapté à un réseau composé de nœuds peu mobiles et de nœuds fortement mobiles. Ce protocole tire partie de la présence de nœuds peu mobiles dans le réseau en les utilisant pour acheminer les messages. Les liens entre les nœuds peu mobiles sont supposés être plus stables que ceux des nœuds fortement mobiles, puisque ces liens se brisent moins souvent que les autres. Chaque nœud du réseau conserve ainsi une table d'associativité définissant sa *stabilité d'associativité* (ou de connectivité) avec chacun de ses 1-voisins. Cette table est mise à jour suite à la diffusion périodique (sur un saut) de messages *hello* par chaque nœud. Pour découvrir une route vers une destination, un nœud diffuse une requête de localisation. Chaque nœud intermédiaire, lorsqu'il reçoit le message, vérifie qu'il n'est pas la destination et rediffuse ensuite le message en y ajoutant son adresse et sa table d'associativité (c'est-à-dire sa stabilité d'associativité avec chaque nœud 1-voisin). Un nœud intermédiaire recevant ce message supprime de la table d'associativité contenue dans le message, la stabilité d'associativité entre le nœud lui ayant fait suivre le message et les 1-voisins de ce dernier afin de ne conserver que celle le concernant lui et le nœud qui lui a expédié le message. Finalement, quand la requête de localisation atteint la destination, elle est capable de choisir la meilleure route parmi les différentes routes produites, c'est-à-dire, celle offrant la meilleure stabilité de connectivité.

Le protocole CBRP (*Cluster Based Routing Protocol*) [47] est un protocole plus particulièrement adapté aux réseaux *ad hoc* caractérisés par une faible mobilité. En effet, ce protocole est basé sur une gestion de groupes difficiles à maintenir si les nœuds sont fortement mobiles. Plus précisément, ce protocole décompose le réseau en groupes de rayon égal à un saut. Chaque groupe est composé d'un unique coordinateur ayant une complète connaissance du groupe (c'est-à-dire des membres du groupe et des liens entre lui-même et les membres du groupe). Chaque nœud du réseau maintient deux tables : une table de ses 1-voisins, et une table des groupes adjacents composée de la liste des groupes adjacents et de leur coordinateur respectif. Ces deux tables sont maintenues à jour grâce aux messages *hello*

diffusés périodiquement par chaque nœud sur deux sauts. Ce message contient l'état du nœud (du point de vue de son appartenance à un groupe), l'identifiant du coordinateur du (ou des) groupe(s) au(x)quel(s) appartient le nœud, une liste des nœuds 1-voisins et une liste des groupes adjacents incluant l'identifiant de leur coordinateur respectif. Lorsqu'un nœud reçoit un message *hello*, il rafraîchit ses deux tables. Plus précisément, si l'expéditeur du message *hello* ne fait pas partie de son groupe, il définit l'expéditeur comme étant une passerelle vers un groupe adjacent, et ajoute cette information à sa table de groupes adjacents. Par ailleurs, ces messages *hello* sont utilisés pour élire un coordinateur. En effet, un nœud, n'ayant aucun coordinateur, spécifie dans le message *hello* qu'il cherche un coordinateur, c'est-à-dire que le nœud se trouve dans l'état *indécis*. Après expiration d'un délai, si ce dernier ne reçoit pas en réponse un message *hello* de la part d'un coordinateur, alors, il se définit comme coordinateur. Dans le cas contraire, il se définit comme membre d'un groupe. L'utilisation de groupes permet de limiter le trafic généré lors de la localisation d'un nœud. Plus précisément, lorsqu'un nœud souhaite localiser une destination, il diffuse (sur un saut) sa requête contenant : le coordinateur du (ou des) groupe(s) au(x)quel(s) il appartient, la liste des coordinateurs adjacents et des passerelles correspondantes. Les nœuds passerelles (spécifiés dans la requête) font suivre l'information aux coordinateurs adjacents. Ces derniers répondent à la requête si la destination appartient au groupe qu'ils coordonnent. Dans le cas contraire, ils font suivre la requête aux coordinateurs des groupes adjacents.

Les protocoles AODV (*On-Demand Distance Vector routing protocol*) [63] et DSR (*Dynamic Source Routing protocol*) [48] sont deux protocoles populaires, qui réagissent particulièrement bien si la densité et la mobilité des nœuds ne sont pas très importantes [25]. Notons que DSR réagit mieux à la mobilité et à la densité des nœuds que AODV. Toutefois, de par sa conception, DSR provoque un trafic de contrôle plus important que AODV en termes de taille de message et de nombre de messages générés. La comparaison de ces deux protocoles montre que le type d'information conservée et les mécanismes de localisation appliqués sont différents. En particulier, les informations de routage stockées par un nœud DSR correspondent à un chemin, alors qu'un nœud AODV ne conserve que l'identifiant du prochain nœud (c'est-à-dire, du nœud 1-voisin) faisant transiter le paquet vers la destination. Avant d'initier une session de données vers une destination, un nœud AODV ou DSR (lorsqu'il ne conserve pas la route correspondante) diffuse une requête de localisation sur le réseau, contenant l'identifiant de la source et de la destination. Un nœud intermédiaire AODV enregistre l'adresse de la source ayant initié le processus de localisation et l'adresse du nœud

intermédiaire ascendant (c'est-à-dire du nœud ayant envoyé le paquet) afin de pouvoir ré-acheminer ultérieurement la réponse vers la source ayant initié le processus de localisation. En revanche, un nœud DSR enregistre la route (entre la source et le nœud lui-même). Puis, un nœud (AODV ou DSR) réagit de deux façons différentes selon qu'il dispose ou non de l'information de routage. Lorsque le nœud ne possède pas les informations de routage correspondant à la route demandée, il (re)diffuse la requête de localisation sur le réseau. Un nœud AODV ne modifie alors pas le message tandis qu'un nœud DSR y ajoute son identifiant (c'est-à-dire, son adresse IP). Lorsque l'information de routage est connue par le nœud intermédiaire, ce dernier répond à la source. Le message de réponse inclut l'identifiant de la destination dans le cas d'AODV. Dans le cas de DSR, le message inclut la route de la source vers le nœud intermédiaire (cette route étant extraite de la requête de localisation), suivie de la route entre le nœud intermédiaire et la destination (cette route étant obtenue à partir de la table de routage locale). Un nœud intermédiaire recevant le message de réponse de la source à l'origine de la création de la requête de localisation envoie le paquet au prochain nœud intermédiaire en extrayant son adresse du message de réponse dans le cas de DSR, ou en extrayant l'adresse du prochain nœud intermédiaire des tables de routage dans le cas d'AODV.

Les protocoles proactifs offrent de meilleures performances, en terme de temps de réponse, que les protocoles réactifs. En effet, une route est rapidement disponible, puisqu'un nœud souhaitant communiquer dispose immédiatement des informations de localisation nécessaires. En contrepartie, un protocole proactif nécessite une capacité de calcul et de mémoire plus importante que celle requise par les protocoles réactifs. En revanche, les protocoles réactifs induisent un temps de réponse accru, qui est dû à la découverte des routes à la demande. En effet, le temps de réponse de cette découverte comprend la diffusion d'une requête de localisation sur le réseau et l'attente inhérente à l'obtention d'une réponse. Les protocoles de types réactifs et proactifs affichent des performances différentes selon les spécificités du réseau. Si le réseau est dense ou si les nœuds échangent fréquemment des informations, un protocole réactif est plus coûteux qu'un protocole proactif puisque la diffusion excessive de requêtes de localisation tend à inonder le réseau. Un protocole proactif, quant à lui, n'affiche pas des performances équivalentes aux protocoles réactifs si le trafic généré par les nœuds est faible, puisqu'il surcharge inutilement le réseau en vérifiant de façon continue la localisation des nœuds. Pour ces différentes raisons, des protocoles hybrides, réalisant un compromis entre les protocoles proactifs et réactifs, ont émergé.

2.2.1.3 Protocoles de routage hybrides

Les protocoles de routage hybrides combinent les modèles proactifs et réactifs. Ces protocoles, tels que ZRP (*Zone Routing Protocol*) [35] et SHARP (*Sharp Hybrid Adaptive Routing Protocol*) [68], définissent deux zones :

- une *zone proactive* (ou *intra-zone*) correspondant au *N*-voisinage au sein duquel est appliqué un modèle proactif, et
- une *zone réactive* (ou *inter-zone*) correspondant au reste du réseau (c'est-à-dire, excluant le *N*-voisinage), au sein de laquelle une approche réactive est appliquée.

Les protocoles hybrides tirent profit du fait que les nœuds lointains communiquent moins souvent entre eux que les nœuds voisins, et qu'il est donc plus avantageux de bien connaître la topologie locale alors que la connaissance globale du réseau est comparativement coûteuse et moins utile.

Le protocole ZRP définit son propre protocole IARP (*IntrAzone Routing Protocol*) [35] pour maintenir les informations de routage dans une intra-zone. Afin de réduire le trafic généré lors de la diffusion de requêtes de localisation, deux techniques sont appliquées :

- la technique dîte de *bordercasting,* avec laquelle seuls les nœuds distants de *N* sauts (correspondant aux nœuds se trouvant à la périphérie de l'intra-zone), retransmettent les messages de requêtes,
- la technique dîte d'*overlapping,* avec laquelle seul un nombre restreint de nœuds périphériques dont les zones de transmission ne se recouvrent pas entre elles, retransmettent les messages.

Le protocole SHARP définit pour sa part le protocole SPR (*SHARP Proactive Routing protocol*) pour maintenir proactivement les informations relatives aux terminaux se trouvant dans le voisinage d'un nœud donné, que nous appelons nœud central par la suite. Plus précisément, le protocole SPR se base sur les algorithmes de routage des protocoles DSDV et TORA. Le nœud central maintient les informations de routage grâce à un arbre acyclique dirigé (*directed acyclic graph*) dont la racine est le nœud central. Pour créer et mettre à jour cet arbre, il utilise l'algorithme de construction de l'arbre du protocole TORA (§ 2.2.1.2).

Afin de pouvoir acheminer les messages entre deux nœuds ne se trouvant pas dans la même zone proactive, ZRP se base quant à lui sur son propre protocole IERP (*IntErzone Routing*

Protocol) [35]. Le protocole SHARP s'appuie, quant à lui, sur le protocole AODV. Par ailleurs, afin de définir une taille optimale de la zone proactive, ZRP adapte la taille cette zone à la mobilité des nœuds (en supposant que celle-ci est directement responsable des cassures de liens), alors que SHARP adapte la taille de la zone proactive à partir d'un modèle de coût. Ce dernier quantifie le coût de maintenance d'une zone proactive (en fonction du nombre moyen de cassures de lien survenues avec les nœuds voisins) et du coût d'une approche réactive (en fonction du taux moyen de messages de données reçus et du nombre de sources à l'origine de ces données). Par conséquent, des zones proactives ne sont présentes qu'auprès des nœuds dits ``*chauds*'', c'est-à-dire, recevant du trafic de la part d'un nombre important de sources. La taille de la zone est alors petite si le trafic généré par la réception des données est faible, sinon cette taille est plus grande. De plus, la taille de la zone est aussi définie en fonction du coût affecté à la maintenance de cette zone.

2.2.1.4 Protocoles de routage géographiques

Les protocoles de routage géographiques utilisent des coordonnées géographiques (par exemple, fournies par GPS) afin d'acheminer les messages de façon efficace vers la destination, en routant les messages dans la direction de la destination. Pour atteindre cet objectif, les coordonnées géographiques des nœuds sont incluses dans les tables de routage. Concrètement, un nœud inclut l'adresse IP et la position de la destination (fournie par le protocole de routage) dans le message de données à expédier, et envoie ce message dans la direction de la destination. Les nœuds intermédiaires répètent le même mécanisme jusqu'à ce que le message atteigne la destination. Nous donnons ci-après des exemples représentatifs de protocoles géographiques en reprenant trois protocoles qui sont respectivement de type proactif, réactif et hybride.

Le protocole DREAM (*Distance Routing Effect Algorithm for Mobility*) [8] maintient à jour, de façon proactive, les tables de routage. Pour cela, chaque nœud envoie des informations relatives à sa localisation, à tous les nœuds du réseau. Plus précisément, partant du principe qu'un nœud perçoit moins la mobilité d'un autre nœud lorsque ce dernier est éloigné, et donc que l'impact de cette mobilité est moins important au niveau du routage, les notifications de changement de positions sont moins souvent diffusées aux nœuds lointains qu'elles ne le sont aux nœuds proches. Par ailleurs, les nœuds caractérisés par une forte mobilité diffusent plus souvent leurs coordonnées géographiques que les autres.

Le protocole ZHLS (*Zone-Based Hierchical Link State Routing*) [58] est un protocole hybride. Il décompose, de façon statique, le réseau en zones disjointes à partir d'une carte. Chaque nœud connaît la topologie de sa zone (approche proactive) et les nœuds jouant le rôle de passerelle (*gateway*) entre les zones. Ces nœuds passerelles sont connus par les autres nœuds puisqu'ils diffusent sur l'ensemble du réseau les informations concernant les zones qu'ils connectent. Ainsi, chaque nœud du réseau connaît le découpage du réseau en zones et les nœuds donnant accès à ces zones. Lorsqu'un nœud désire connaître la localisation d'un autre nœud, il vérifie que le nœud ne se trouve pas dans sa zone. Si ce n'est pas le cas, il envoie une requête de localisation vers toutes les zones présentes dans le réseau en utilisant les informations correspondant aux nœuds passerelles pour atteindre ces zones.

Le protocole LAR (*Location-Aided Routing protocol*) [50] est un protocole réactif. Il limite la surcharge du réseau induite par la localisation d'un nœud, en définissant soit une zone de recherche, soit les coordonnées du nœud. Pour localiser un nœud, c'est-à-dire, identifier ses coordonnées géographiques ou la zone dans laquelle il se trouve, le protocole LAR utilise les dernières coordonnées, vitesses et directions connues et suivies pour le nœud à localiser. Lors de la recherche du nœud, deux techniques sont proposées : soit le nœud demandeur inclut les coordonnées hypothétiques de la destination, soit il définit un carré de recherche dans lequel la destination est supposée se trouver. Ainsi, l'inondation du réseau est contrôlée par les nœuds durant l'étape de localisation car ils ne transmettent une requête que si la requête se dirige dans la bonne direction, c'est-à-dire, vers la zone de recherche.

2.2.1.5 Optimisation de la localisation au moyen de caches

Le domaine des réseaux *ad hoc* a fait l'objet de nombreux travaux de recherche centrés sur la définition de protocoles de routage. L'étude approfondie des comportements des divers protocole de routage proposés dans la littérature a notamment suscité : (i) l'introduction de nouveaux protocoles de routage adaptés à l'environnement, comme c'est par exemple le cas d'OLSR qui a été conçu pour pallier les insuffisances de DSDV, (ii) l'enrichissement des protocoles de routage en prenant en compte des informations géographiques, (iii) l'optimisation et l'adaptation des protocoles de routage aux spécificités du réseau. L'optimisation des protocoles de routage a notamment conduit à l'introduction de caches sur les nœuds du réseau, pour stocker les informations de routage gérées par les protocoles

réactifs. Notons ici qu'il n'est pas pertinent de gérer la mise en cache des informations de routage des protocoles proactifs puisque ces derniers mettent à jour les tables de routage de manière continue. En revanche, certains protocoles réactifs, comme AODV et DSR, offrent la possibilité d'utiliser des caches stockant des routes du réseau, afin d'optimiser les performances du protocole. L'utilisation de caches pour stocker des routes permet de limiter le trafic généré par le protocole de routage et de diminuer le temps d'attente induit par la recherche d'une route. En effet, les protocoles de routage réactifs intégrant une gestion de cache, ont en commun le fait qu'avant d'envoyer des données vers un terminal, *B*, un terminal, *A*, diffuse dans le réseau une requête de localisation de *B*, sauf si *A* stocke dans son cache la route menant vers *B*. Dans ce dernier cas, *A* envoie directement les données à *B via* la route stockée dans le cache, ce qui réduit à la fois le trafic généré (puisque aucune requête n'est diffusée) et le temps de réponse. Dans le cas où *A* doit diffuser la requête de localisation de *B*, les nœuds intermédiaires (c'est-à-dire, faisant suivre la requête de localisation) ont la possibilité d'extraire de leurs caches ladite route et donc de limiter la diffusion de la requête.

Trois techniques permettent d'augmenter les performances des caches des protocoles réactifs, c'est-à-dire, de réduire l'inondation du réseau résultant de la diffusion de requêtes de localisation. Ces techniques visent à :

- stocker le plus grand nombre de routes dans les caches,
- maintenir le plus longtemps possible les routes valides dans le cache,
- expulser le plus rapidement possible du cache les routes devenues invalides.

Nous détaillons chacune de ces techniques dans ce qui suit.

Afin d'augmenter le nombre de routes stockées dans chaque cache et de privilégier la diffusion des routes valides dans le réseau, le protocole TORA applique la technique suivante [62] : la destination répond à toutes les requêtes de route qu'elle reçoit. Ainsi, la source connaît un certain nombre de routes alternatives et peut calculer la route la plus courte vers la destination. L'aspect fondamental de cette technique est que tous les nœuds intermédiaires recevant une réponse de la part de la destination, peuvent mettre à jour leurs caches. Notons que la route cachée peut être ultérieurement utilisée lors de la localisation d'autres routes dont elle est un sur-ensemble.

Les protocoles de routage utilisent les informations transitant sur le réseau afin de tenir à jour leurs caches. Ces informations proviennent des en-têtes des messages des requêtes de

localisation et de leurs réponses, et des messages de données. Ainsi, les routes valides sont constamment insérées dans le cache. Doshi *et al.* [26] proposent que les nœuds intermédiaires écoutent les messages qui ne leur sont pas destinés. Si ces nœuds possèdent dans leur cache une route plus courte vers la destination, ils envoient un message « *gratuitous repl y »* à la destination, qui cache alors cette information pour une futur utilisation.

Deux techniques peuvent être utilisées afin d'expulser les éléments d'un cache : (i) soit le protocole affecte à chaque route une durée de vie, (ii) soit le protocole notifie qu'une route n'est plus valide aux caches stockant ladite route. AODV utilise la première technique et associe un champ TTL (*Time To Live*) correspondant à la durée de validité de l'élément, à chaque élément contenu dans le cache. Une route est expulsée du cache lorsqu'elle n'a pas été utilisée ou rafraîchie durant cette période de temps. DSR et TORA utilisent la seconde approche. Une route contenue dans le cache d'un nœud est valide tant que cette dernière ne reçoit pas de notification de son invalidité, c'est-à-dire, tant qu'aucun message d'erreur n'indique que la route a été rompue lors de l'acheminement d'un message de données. Avec DSR, si un lien faisant partie de la route utilisée pour acheminer une donnée est brisé, alors un message d'erreur est renvoyé vers la source. Ainsi, les nœuds intermédiaires, correspondant aux nœuds ayant préalablement acheminé le message de données et faisant actuellement transiter le message, se voient notifier cette cassure. Avec TORA, une cassure de lien n'est notifiée que dans l'entourage où elle survient. Dans les deux cas, un nœud ayant reçu une notification de cassure, expulse de son cache, la route et les portions de routes correspondant à la cassure. Deux principaux problèmes résultent de cette technique :

- Les liens cassés ne sont perçus que si un message de données chemine par la route concernée. Marina *et al.* [55] proposent d'utiliser un mécanisme plus réactif, en associant un TTL à chaque route, afin d'expulser les portions de route non utilisées pendant une période égale au TTL. La valeur du TTL est fixée en fonction du temps moyen entre deux cassures pour ladite route.

- La notification d'erreur est partielle puisque le message d'erreur n'est propagé qu'à un nombre restreint de nœuds, c'est-à-dire, aux nœuds ayant fait transiter le message dans le cas de DSR, et aux nœuds de l'entourage dans le cas de TORA. Les conséquences de cette *non-notification* sont multiples et significatives. Les nœuds intermédiaires non alertés par la cassure du lien et cachant la route, continuent d'utiliser cette route et ne cessent

d'indiquer de fausses routes aux nœuds sources cherchant à localiser un nœud dont le chemin d'accès incorpore le lien cassé. De plus, les nœuds sources envoient des messages de données qui vont eux-mêmes contenir des routes erronées. Or, ces informations erronées sont utilisées pour modifier les tables de routage. Finalement, même les nœuds ayant déjà reçu le message de notification de rupture de lien vont réincorporer la route erronée dans leur cache.

Afin d'améliorer le processus de notification d'erreurs, Marina *et al.* [55] proposent de diffuser les messages d'erreur à tous les nœuds contenant la route désormais erronée. Plus précisément, les nœuds intermédiaires font suivre le message d'erreur, si et seulement si, ils cachent des routes contenant le lien de cassure. Panchal *et al.* [61] proposent que le nœud source alerté par un message d'erreur, inclut dans sa nouvelle recherche de route, la route précédemment rompue. Ainsi, les nœuds n'ayant pas reçu la notification de la rupture d'un lien, ne renvoient pas la route invalide à la source. Par conséquent, la dissémination de routes erronées est évitée. Afin de notifier aux autres nœuds qu'un lien a été rompu, AODV maintient une liste des nœuds voisins qui utilisent une entrée du cache. Ainsi, quand le lien entre un nœud A et un de ses voisins est rompu, A envoie un message d'erreur à la liste des nœuds voisins utilisant ce lien pour acheminer l'information. Les nœuds recevant ce message d'erreur le renvoient aux nœuds connus utilisant ce lien, et ainsi de suite.

Afin de limiter l'impact des ruptures de liens lors de l'acheminement des données, deux approches ont été proposées. La première approche préconise que les nœuds faisant transiter un message de données puissent modifier la route contenue dans l'en-tête du message si cette dernière contient un lien rompu [92]. Ainsi, le processus de localisation du nœud n'est pas réitéré à chaque fois qu'une route est rompue. Wang *et al.* [88] utilisent prioritairement les routes stables pour acheminer les messages de données avec le protocole AODV. Pour cela, chaque nœud diffuse à chaque période T un message *hello* sur un saut. Chaque nœud calcule ensuite la stabilité de ses liens voisins en fonction du nombre de messages *hello* reçus de chaque voisin depuis $n \times T$, $n > 0$. Afin de connaître les routes stables, les nœuds diffusent périodiquement de faux messages de requêtes de routes (afin de garder la compatibilité avec AODV). Ces messages sont propagés prioritairement sur les liens stables. Pour cela, la durée de vie du message, définie par le champ TTL dans l'en-tête du message, est décrémentée seulement lorsque le message traverse un lien instable et non lorsqu'il chemine à travers un

lien stable. Ainsi, les routes stables sont souvent utilisées pour le routage des messages de données puisqu'elles sont présentes dans les caches des nœuds intermédiaires.

2.2.1.6 Synthèse

Comme nous l'avons vu, dans un réseau *ad hoc*, un certain nombre de protocoles de différentes natures (réactifs, proactifs, hybrides) et appartenant à différentes catégories (orientés topologie ou destination, hiérarchique ou non), peuvent être utilisés pour déterminer la localisation des terminaux. Selon le contexte d'utilisation, ces protocoles offrent des performances différentes. Le tableau 2-1 propose un récapitulatif des protocoles de routage que nous avons présentés, indiquant leurs principales propriétés (type et catégorie), le contexte d'utilisation dans lequel ils sont particulièrement performants, leur état d'avancement dans le processus de normalisation, le niveau d'implication des nœuds dans le routage, et ceux utilisant des caches.

Notons que tous les protocoles de routage (multi-sauts) que nous avons introduits ont en commun de se baser sur une *approche collaborative* pour déterminer la localisation des terminaux, et supposent que:

- chaque terminal est intéressé par les informations relatives au routage puisque ces dernières sont indispensables à chaque nœud voulant communiquer avec un nœud du réseau,
- les messages de contrôle nécessaires à la dissémination de la topologie du réseau ont une taille relativement faible puisque les informations véhiculées correspondent principalement à des listes d'adresses IP.

Les critères servant de base à la conception des protocoles de routage pour les réseaux *ad hoc* sont ainsi l'intérêt des nœuds à collaborer et à partager des informations de routage, et le faible coût induit par l'échange d'informations relatives à la localisation de terminaux. Ces critères ne sont pas toujours vérifiés dans le cadre général de la localisation de ressources dans un environnement mobile. En effet, dans le cas d'un réseau *ad hoc* spécialisé, tel qu'un réseau de capteurs, on peut supposer que tous les nœuds du réseau sont intéressés par le même type de ressources et d'informations puisqu'ils partagent un but commun. En revanche, une telle hypothèse est difficilement vérifiable dans un réseau *ad hoc* hétérogène ouvert, intégrant différents types de terminaux sans fil (par exemple, PDA, téléphone, PC portable) appartenant éventuellement à différentes catégories d'utilisateurs (par exemple, réseau *ad hoc* établi dans la maison communicante ou réseau *ad hoc* établi dans une galerie marchande ou un moyen de

transport public). Par ailleurs, la diffusion, la réplication ou la distribution d'informations relatives aux ressources sont difficiles à réaliser dans un réseau *ad hoc*, car elles ont un impact direct et important sur le trafic généré, en raison de la taille des données échangées. Par conséquent, les approches utilisées pour la localisation de ressources dans un réseau *ad hoc* diffèrent de celles utilisées pour la location de terminaux. Nous les présentons dans la section suivante.

Tableau 2-1 Propriétés des protocoles de routage

Protocole	Type	Catégorie	Implication des nœuds	Caractéristique du réseau	Normalisation IETF	Utilisation de caches
DSDV [64]	Proactif	Orienté destination	Uniforme	Faible mobilité faible densité	--	--
OLSR [23]	Proactif	Orienté topologie	Non-uniforme	Trafic de données important	RFC	--
TBRPF [60]	Proactif	Orienté topologie	Non-uniforme	Trafic de données important	RFC	--
TORA [62]	Réactif	Orienté destination	Uniforme	Forte mobilité	Draft expiré	X
ABR [78]	Réactif	Orienté destination	Uniforme	Forte mobilité	--	--
CBRP [47]	Réactif	Hiérarchique de groupe	Non-uniforme	Faible mobilité	Draft expiré	--
AODV [63]	Réactif	Orienté destination	Uniforme	Trafic de données faible	RFC	X
DSR [48]	Réactif	Orienté topologie	Uniforme	Trafic de données faible	RFC	X
SHARP [68]	Hybride	Orienté destination	Uniforme	Zone de trafic de données important	--	X
ZRP [35]	Hybride	--	Non-uniforme	--	Draft expiré	--
DREAM [8]	Géographique, Proactif	Orienté topologie	Uniforme	Forte mobilité faible densité	--	--
ZHLS [58]	géographique, hybride	Hiérarchique	Non-uniforme	--	--	--
LAR [50]	géographique, réactif	Orienté Topologie	Uniforme	Faible mobilité	--	

2.2.2 Localisation de ressources

Dans les réseaux filaires, une approche couramment utilisée afin de localiser les ressources est de se baser sur un annuaire. Cet annuaire centralise les informations relatives aux ressources (lorsqu'il s'agit de services ou de ressources matérielles) ou les contenus, comme par exemple des fichiers [104], des tuples [90], et ce dans le but de répondre aux requêtes des clients. Cette approche est aussi bien utilisée dans des réseaux à large échelle comme, par exemple, par UDDI[6] (*Universal Description, Discovery and Integration*) pour la découverte de services de l'Internet, que dans des réseaux locaux comme, par exemple, par Jini[7] et SLP (*Service Location Protocol*) [34]. Notons que SLP propose deux modes de fonctionnement, l'un étant centralisé et donc adapté aux réseaux de taille importante et l'autre entièrement décentralisé, c'est-à-dire sans annuaire, conçu pour des réseaux de petite taille.

La localisation centralisée des ressources n'est pas directement exploitable dans un réseau *ad hoc*. En effet, comparé à un réseau filaire, les déconnexions dans un réseau *ad hoc* sont extrêmement fréquentes. Or, si un annuaire est déconnecté, la localisation de ressources ne peut avoir lieu. Ainsi, un annuaire correspond à un point de cassure unique. Par ailleurs, cet annuaire constitue un goulet d'étranglement. La concentration du trafic provenant de l'ensemble du réseau au niveau de l'annuaire est en outre à l'origine d'une consommation énergétique élevée pour les terminaux proches de ce dernier. Finalement, un réseau *ad hoc*, par essence même, ne dispose pas d'administrateur réseau gérant le déploiement d'annuaire(s). Pour ces raisons, les solutions à la localisation de ressources dans un réseau *ad hoc* se basent sur une approche décentralisée. Notons que cette approche décentralisée est aussi proposée pour des réseaux de petite taille car elle permet de s'affranchir d'une intervention humaine, requise pour assurer la gestion de l'annuaire dans le cas centralisé. Nous proposons ci-après un tour d'horizon des approches suivies pour la localisation de ressources dans les réseaux de petite et large échelle.

Nous identifions deux modèles pour la localisation de ressources, suivant que le processus soit piloté par les fournisseurs ou les consommateurs des ressources du réseau. Nous parlons

[6] http://www.uddi.org
[7] http://www.jini.org

de modèle *push* dans le premier cas et de modèle *pull* dans le second. Plus précisément, le principe du modèle *push* consiste à envoyer aux utilisateurs, de manière proactive, des informations relatives aux ressources du réseau au moyen de messages de publicité, sans que ceux-ci n'en fassent la demande. Cette approche est par exemple adoptée par le protocole DEAPspace [59,41] pour la localisation de ressources dans les réseaux *ad hoc* à un saut. Un terminal mettant à disposition une ressource diffuse ainsi périodiquement des identifiants des ressources qu'il offre, ainsi que la liste des identifiants des ressources dont il a connaissance, auprès des terminaux clients. Cette technique de *push* induit une forte consommation de bande passante puisqu'elle se base sur une diffusion périodique de publicités de ressources. Avec le modèle *pull*, les utilisateurs du réseau initient de façon réactive la localisation de la ressource qu'ils souhaitent accéder, au moyen de requêtes spécifiques. Par exemple, le protocole de découverte de Salutation[8] suppose qu'un client, s'il souhaite découvrir une ressource, interagit avec le gestionnaire Salutation (ou SLM pour *SaLutation Manager*) local. Ce dernier diffuse alors un message aux autres SLMs afin de définir si ceux-ci disposent de la ressource demandé. Notons que le SLM se trouve typiquement sur le terminal du client et du fournisseur de ressources. Afin de réduire le coût provenant de la diffusion de ces requêtes sur l'ensemble du réseau, des solutions alternatives ont été proposées. Elles visent à répartir les informations relatives aux ressources (c'est-à-dire à leurs localisation/descriptions, ou aux données qu'elles fournissent ou encore aux résultats renvoyés suite à leurs accès) sur le réseau afin de les rapprocher des consommateurs. Nous décrivons de façon plus précise cette approche dans la section 2.2.2.3. Notons que les modèles *pull* et *push* sont applicables à des protocoles de localisation décentralisés aussi bien que centralisés. Dans le premier cas, les échanges de messages de requêtes/publicités se font directement entre les consommateurs et fournisseurs de ressources. Dans le second cas, les messages transitent par le(s) annuaire(s) déployé(s) dans le réseau.

Afin de réduire le trafic généré par l'envoi périodique des messages de publicité de ressources par l'approche *push*, ou la diffusion des requêtes de localisation induite par l'approche *pull*, plusieurs méthodes ont été utilisées. Elle visent à :

- contrôler la dissémination des messages, c'est-à-dire à restreindre à la fois la diffusion des messages en ne les propageant qu'aux terminaux intéressés, et la quantité de messages diffusée en ne propageant pas les messages non révélateurs,

[8] http://www.salutation.org/

- limiter la quantité de messages et d'informations envoyés, en agrégeant les messages envoyés et en limitant la taille des informations incluses dans les messages,
- limiter la portée, c'est-à-dire, le nombre de sauts, des requêtes de localisation envoyées en répliquant les informations relatives aux ressources sur des terminaux proches des utilisateurs.

Une première technique visant à restreindre le trafic généré par les protocoles de localisation de ressources est basée sur la diffusion *multicast*, qui permet de cibler les terminaux affectés par les messages générés. Un protocole *multicast* permet en effet de transmettre des paquets à un groupe d'utilisateurs (ou abonnés virtuels) identifiés par une seule adresse IP. Un tel protocole est généralement utilisé pour gérer la diffusion d'informations vers des groupes dont les membres collaborent fortement que ce soit entre eux, ou dans un même but. Nous décrivons plus précisément l'exploitation de la diffusion *multicast* dans la localisation de ressources dans les réseaux *ad hoc*, dans la section suivante.

2.2.2.1 Localisation de ressources basée sur la diffusion *multicast*

La diffusion dans un environnement *ad hoc* se base sur les techniques de diffusion *multicast*, afin de limiter le trafic généré par la diffusion. L'inondation (*flooding*) et le routage basé sur des arbres (*tree-based routing*) sont les deux principales techniques utilisées par les protocoles de diffusion *multicast*.

La diffusion *multicast* par inondation est l'approche basique générant peu de messages de contrôle mais induisant un fort trafic lié à l'acheminement des données. Un tel protocole *multicast* est notamment utilisé pour offrir un service DNS (*Domain Name System*) [22] dans un réseau *ad hoc*, alors qu'aucun serveur DNS centralisé n'est présent dans le réseau. La traduction des noms des ressources en adresses IP (et *vice versa*) ne peut en effet pas être réalisée par un unique serveur DNS dans un réseau *ad hoc*, ce qui conduit à partager cette charge entre les terminaux du réseau. Concrètement, un protocole *multicast* est utilisé pour éviter l'affectation de noms identiques à des ressources distinctes du réseau. Une diffusion *multicast* par inondation est particulièrement adaptée dans ce cas car la diffusion des requêtes des utilisateurs et des réponses du service DNS distribué fournit une détection passive des conflits. De plus, chaque terminal étant pourvu d'un cache, une réponse *multicast* permet de modifier les informations stockées dans le cache et de limiter le nombre de requêtes envoyées sur le réseau. Cependant, notons que chaque requête est diffusée dans l'ensemble du réseau,

même si un terminal voisin connaît la réponse. Ce choix est légitimé par le fait que la fonction essentielle d'un service DNS est de détecter des conflits de noms. Une approche similaire a été retenue pour effectuer la localisation de ressources dans un réseau *ad hoc* [15]. Elle suppose que pour chaque requête de localisation, tous les terminaux du réseau reçoivent la requête (même si un terminal proche possède la réponse). Par ailleurs, tous les terminaux stockant des informations relatives aux ressources retardent l'envoi de leur réponse (en fonction de la période de temps durant laquelle les informations relatives aux ressources ont été stockées), ce qui conduit à un temps d'attente accru pour l'utilisateur. D'autre part, chaque client répondant à la requête diffuse la description de la ressource par *multicast*, s'il n'a pas déjà reçu une réponse à cette requête. Cela suppose, par conséquent, que pour chaque terminal souhaitant accéder à la ressource, tous les terminaux du réseau reçoivent au moins une fois cette requête, et que chaque terminal du réseau reçoive au minimum une réponse envoyée par p (p >0) terminaux répondant à la requête.

Afin d'éviter l'inondation du réseau, une autre approche pour la diffusion *multicast* se base sur une gestion de groupes, qui permet de limiter la diffusion aux seuls nœuds intéressés. Cette approche génère un trafic relativement faible pour l'acheminement des données, mais la maintenance de la structure de diffusion sous forme d'arbre, induit un fort trafic de contrôle. Le défi posé par la conception de protocoles *multicast* réside ainsi dans l'efficacité de la maintenance de l'appartenance à un groupe et de la création de la structure de diffusion *multicast*. Le protocole SLP (*Service Location Protocol*) [34] se base sur des groupes de diffusion pour la découverte/localisation de services suivant un modèle *pull* décentralisé. Chaque nœud SLP (c'est-à-dire, un client ou un fournisseur de services) joint ainsi le(s) groupe(s) *multicast* associé(s) au protocole. Puis, un client souhaitant accéder à un service inclut dans sa requête de localisation, la description du service qu'il souhaite accéder. Cette requête est ensuite diffusée au(x) groupe(s) *multicast* auquel appartient le client. Les fournisseurs de services écoutant continuellement le réseau, ceux offrant le service demandé répondent au client en lui envoyant un message *unicast*. Le protocole Konark [40] utilise pour sa part une diffusion *multicast* pour implémenter un modèle *push* décentralisé; les terminaux mettant à disposition des ressources envoient périodiquement les descriptions de ces dernières aux membres du groupe *multicast*. Pour localiser une ressource, un client ayant nouvellement joint le groupe *multicast* peut néanmoins envoyer une requête vers le groupe *multicast* suivant le modèle *pull*. Cette approche permet de limiter la fréquence d'envoi des informations et le temps d'attente d'un utilisateur voulant accéder à une ressource. Le protocole SSDP (*Simple*

Service Discovery Protocol) [31] est semblable au protocole Kornak. Toutefois, il diffère de ce dernier au sens où il est de type événementiel (*event-driven*). Plus précisément, les fournisseurs de ressources diffusent au moyen d'un protocole *multicast,* une description des ressources offertes, seulement lorsque des changements surviennent (mise à disposition ou retrait d'une ressource), et non de façon périodique. Cela s'explique par le fait que le protocole SSDP est conçu pour des réseaux relativement peu dynamiques dans lesquels les déconnexions sont majoritairement volontaires et/ou peu fréquentes.

Cheng et *al.*[21] se basent sur le protocole *multicast* ODMRP (*On-DeMand Routing Protocol*) [51] pour assurer la localisation de ressources dans un environnement *ad hoc*. Le protocole ODMRP crée l'arbre *multicast* uniquement lorsqu'un terminal (appelé *source*) souhaite diffuser des données sur le réseau. Pour cela, la source initie la création d'un arbre *multicast* en diffusant sur tout le réseau une requête d'adhésion (*join query*). Chaque nœud recevant cette requête renvoie à la source un message *join reply*. De cette façon, la source connaît tous les nœuds faisant partie du groupe *multicast* et crée donc l'équivalent d'une liste de diffusion qui est utilisée pour diffuser les informations relatives aux ressources. Ce protocole de diffusion tire profit de ODMRP lors de la création de l'arbre de diffusion : chaque fournisseur mettant à disposition des ressources (c'est-à-dire, source) insère des informations relatives à ces dernières au message de *join query* envoyé sur le réseau. Seuls les membres du groupe intéressés par la ressource envoient une réponse au fournisseur, ce qui leur permet de créer pour chaque fournisseur un arbre *multicast* composé des terminaux intéressés par la ressource offerte. De la même façon, un client souhaitant accéder à une ressource envoie une requête dans un paquet *join query*. Les terminaux membres du groupe attendent un temps aléatoire avant de répondre, ce qui permet de vérifier qu'aucune réponse n'a été envoyé par un autre nœud. Les fournisseurs répondent; quant à eux, systématiquement.

D'autres solutions, non basées sur des protocoles de diffusion *multicast*, ont été utilisées afin de contrôler la surcharge du réseau *ad hoc* qui est induite par les protocoles de localisation de ressources. Ces approches visent à réduire la taille des informations échangées. Elles ont été particulièrement utilisées pour les réseaux de capteurs, qui sont des réseaux *ad hoc* spécialisés, composés de capteurs réalisant la même tâche et collaborant pour atteindre un objectif commun. Nous présentons ces techniques dans la section suivante.

2.2.2.2 Localisation de ressources dans les réseaux de capteurs

Typiquement, un réseau *ad hoc* de capteurs réalise une action (par exemple, un prélèvement de température) pour le compte d'un client qui est souvent un utilisateur final. Le but d'un tel réseau est en général de fournir un résultat agrégé à partir des données fournies par les différents capteurs le composant. Nous appelons contenu les données, agrégées ou non, délivrées par le réseau de capteurs. Dans ce contexte, plusieurs techniques visant à restreindre la dissémination des contenus ou à restreindre la taille des contenus échangés dans le réseau, ont été utilisées. Le protocole SPIN (*Sensor Protocol for Information via Negociation*) [38] propose que chaque nœud négocie avec ses voisins avant de transmettre un contenu. Ainsi, seules les informations importantes, c'est-à-dire, utiles aux capteurs et n'ayant pas déjà été transmises, sont disséminées dans le réseau. Plus précisément, un nœud souhaitant diffuser un contenu en envoie une publicité à ses voisins. Un nœud voisin recevant cette publicité renvoie alors une requête pour le contenu s'il ne le stocke pas et s'il n'a pas déjà reçu une publicité similaire. Les contenus sont ainsi disséminés en utilisant ce procédé.

D'autres approches se basent sur le filtrage des contenus reçus, ou encore sur la création de résumés générés à partir des contenus envoyés par les capteurs, ou finalement en calculant des sommes, moyennes, compteurs, minima ou maxima à partir des contenus reçues. L'agrégation de contenus est notamment utilisée par les protocoles PEGASIS (*Power Efficient Gathering in Sensor Information Systems*) [53] et LEACH (*Low Energy Adaptive Clustering Hierarchy*) [39]. Le protocole PEGASIS propose que chaque contenu reçu (ou généré par le capteur lui-même) soit fusionné avec les contenus préalablement reçus. Le contenu résultant de cette fusion est ensuite rediffusé dans le réseau avant d'atteindre ultimement la station de base qui collecte ces contenus. Le protocole LEACH diffère du protocole PEGASIS en réduisant le nombre de capteurs communiquant avec la station de base et utilisés pour collecter les informations fournies par le réseau de capteurs. Pour cela, le protocole divise le réseau de capteurs en groupes. Chaque groupe est régi par un nœud coordinateur, responsable de la collecte et de la fusion des contenus fournis par les membres du groupe. Ce coordinateur envoie ensuite les contenus fusionnés à la station de base.

Les techniques de minimisation de la taille des contenus échangés dans un réseau *ad hoc,* que

nous avons présentées, sont particulièrement adaptées aux réseaux spécialisés homogènes. Elle le sont moins pour des réseaux composés de terminaux hétérogènes, où les utilisateurs ne partageant pas nécessairement les mêmes centres d'intérêts. Ainsi, d'autres approches visant à accroître les performances des protocoles de localisation de ressources dans les réseaux *ad hoc* hétérogènes ont été proposées, comme illustré dans la section suivante.

2.2.2.3 Localisation de ressources et réplication

La disponibilité des ressources dans un réseau *ad hoc* est inférieure à celle d'un environnement filaire. En effet, le réseau peut facilement se diviser en plusieurs sous-réseaux non connectés, en raison notamment de la mobilité des utilisateurs. Par conséquent, un utilisateur ne peut accéder à une ressource qui était préalablement offerte par un terminal désormais déconnecté, ou, ayant migré vers un autre réseau *ad hoc*. Des caches, déployés sur les terminaux du réseau *ad hoc* et conservant les ressources, sous la forme d'une réplique des ressources, peuvent toutefois être utilisés, pour accroître la disponibilité des ressources du réseau.

L'interrogation d'un cache local permet d'une part de réduire le nombre de requêtes de localisation initiées par l'utilisateur et ainsi d'augmenter la vitesse d'accès à la ressource lors d'accès répétés, comme cela est également utilisé par les protocoles de routage réactifs. Notons que cette technique est largement utilisée pour le modèle *push* où les terminaux cachent les informations portant sur les ressources, diffusées de façon proactive par les terminaux fournisseurs de ressources. En ce qui concerne le modèle *pull*, les informations cachées correspondent à celles portant sur les ressources ayant été accédés par l'utilisateur.

Plus généralement, le trafic et le temps d'attente liés à l'obtention locale ou à partir d'un terminal proche, des informations relatives aux ressources, sont moins importants que dans le cas de l'obtention à partir d'un fournisseur de ressource plus éloigné. La performance de l'accès à une ressource peut ainsi être quantifiée en termes de :

- diminution du temps d'attente de l'utilisateur,
- diminution du trafic généré par le rapatriement des informations relatives aux ressources du réseau à partir d'un nœud proche ayant précédemment répliquée la ressource ou les informations y afférant, comparé à l'accès à la ressource à partir du nœud fournissant originalement cette ressource.

Finalement, les ressources accédés par un nombre important de clients ne peuvent être gérées par un seul terminal (ou serveur) sans que ce dernier ne devienne un goulet d'étranglement et participe à l'augmentation du temps d'attente de l'utilisateur. Le coût de réplication des informations relatives à la ressource, sur un autre terminal peut être en outre amorti si la popularité de la ressource est suffisante, c'est-à-dire, si la ressource est fréquemment accédée à partir du terminal cachant la réplique. Deux critères principaux doivent être respectés par un processus de réplication. Le critère le plus important est une réplication transparente pour l'utilisateur. Le second critère décisif est la maintenance de la cohérence des informations stockées. Précisément, la réplication des informations relatives aux ressources dans un environnement *ad hoc* repose sur :

- la définition des critères de réplication, et en particulier la sélection des hôtes pour les répliques.
- la définition des techniques utilisées pour maintenir la cohérence ces informations.

Nous précisons ces aspects dans ce qui suit.

Réplication des ressources dans le réseau

Hara [37] propose que chaque terminal stocke (ou cache) une copie des informations relatives aux ressources dont la fréquence d'accès est la plus élevée. Cette approche est fructueuse si les utilisateurs ne partagent pas les mêmes centres d'intérêts ou s'ils accèdent un nombre important de ressources différentes. En revanche, si les utilisateurs accèdent les mêmes ressources, alors les informations sont répliquées sur un nombre important de terminaux, conduisant à une éventuelle consommation significative de la mémoire. Pour résoudre ce problème, Hara propose deux approches complémentaires. La première vise à éviter de stocker les mêmes informations sur des terminaux voisins. Pour cela, chaque terminal diffuse son identifiant et sa fréquence d'accès aux différentes ressources dont il cache des informations s'y rapportant. A partir de ces données, si deux terminaux voisins stockent les mêmes informations, alors seul celui accédant le plus fréquemment les ressources correspondantes continue à stocker ces informations. Afin de supporter un nombre plus important de terminaux, la seconde approche consiste à décomposer le réseau en groupes. Dans ce cas, la fréquence d'accès à une ressource est évaluée par rapport au groupe, par un des terminaux le composant (celui ayant le plus petit identifiant). Chaque information relative à une ressource est ensuite affectée au terminal ayant la plus forte fréquence d'accès.

Cao *et al.* [18] proposent une technique de *caching* pour un réseau *ad hoc* dans le cas où les ressources sont spécifiquement du contenu. La solution proposée est basée sur la conservation/réplication du contenu ou du chemin d'accès au contenu. Dans le premier cas, les nœuds intermédiaires, faisant suivre les messages véhiculant le contenu, conservent une copie de ce dernier pour répondre à de futures requêtes provenant d'autres utilisateurs. Toutefois, un contenu n'est conservé que s'il est suffisamment populaire (pour l'utilisateur ou pour les nœuds faisant partie du réseau). De plus, afin que les contenus ne soient pas répliqués par tous les nœuds du réseau, ce qui résulterait en une utilisation peu efficace des ressources, les répliques ne sont conservées que si elles sont demandées par des nœuds distincts. L'approche alternative proposée se base sur des caches conservant les chemins d'accès aux contenus. Un nœud, A, ayant fait suivre un message relatif à l'accès à un contenu $\mathcal{C}$, qui peut être le message pour la localisation de $\mathcal{C}$, sait : quel terminal, S, met à disposition le contenu $\mathcal{C}$ et quel nœud, D, a demandé l'accès à ce contenu. Le nœud A peut donc cacher le chemin de la source S au nœud D, l'identifiant de $\mathcal{C}$, et le nombre de sauts le séparant de S et D. Ensuite, lorsque A reçoit une requête de localisation pour $\mathcal{C}$, il redirige la requête directement vers S ou D, en fonction de leur localisation respective. Notons que si la requête est redirigée vers D, la réplique de $\mathcal{C}$ peut, entre temps, avoir été expulsées du cache de D. Le nœud A doit alors rediriger la requête vers le fournisseur de la ressource S, ce qui engendre un coût d'accès important. Par conséquent, un nœud intermédiaire A ne dirige une requête vers D que lorsque D est très proche.

Thanedar *et al.* [82] proposent une approche où la répartition des informations relatives à une ressource n'est pas négociée entre les terminaux utilisateurs mais est déterminée par le terminal offrant la ressource. Dans un premier temps, ce dernier définit quelles sont les ressources fréquemment accédées par les utilisateurs. Pour cela, le fournisseur surveille les requêtes envoyées par les utilisateurs pour chaque ressource qu'il met à disposition. Lorsque la fréquence des requêtes (calculée sur une période de temps T) pour une ressource $\mathcal{R}$ dépasse un seuil donné, alors le fournisseur de la ressource diffuse sur le réseau un message *Probe* dans son N-voisinage, avec N définissant la zone dans laquelle les ressources (ou les informations s'y rapportant) seront répliquées. A la réception de ce messages, les nœuds ayant assez d'espace libre pour stocker des répliques de d et pouvant être atteints en (i = 2, .., 2j , .. ,

N) sauts renvoient un message d'agrément au serveur. Le serveur sélectionne ensuite le (ou les) nœud(s) sur lesquels sont répliqués les informations sur la base des réponses reçues. Périodiquement, chaque nœud envoie à ses 1-voisins les identifiants des ressources auxquelles il souhaite avoir accès. Les voisins cachant des informations relatives à ces ressources requises renvoient une réponse au nœud. Si la requête n'est pas satisfaite pendant une certaine période de temps, le nœud envoie une requête au fournisseur de la ressource. Le problème posé par cette solution est qu'elle n'assure pas une distribution uniforme des informations relatives aux ressources sur les nœuds accessibles en *i* sauts. Le fournisseur ne peut distribuer les répliques de façon uniforme parmi ces nœuds puisqu'il n'a pas, *a priori*, d'informations relatives à leur positionnement. La figure 2-3 illustre cette problématique. Supposons qu'un nœud *A* souhaite répliquer les informations portant sur une ressource. Les nœuds *B* (pour i = 2) et *C* (pour *i* = 4) sont choisis pour stocker la réplique. Le nœud *D*, accessible en trois sauts à partir de *A* doit, s'il souhaite accéder une ressource, contacter le nœud *A* et non le nœud *C* ou *B* accessibles en respectivement quatre et six sauts.

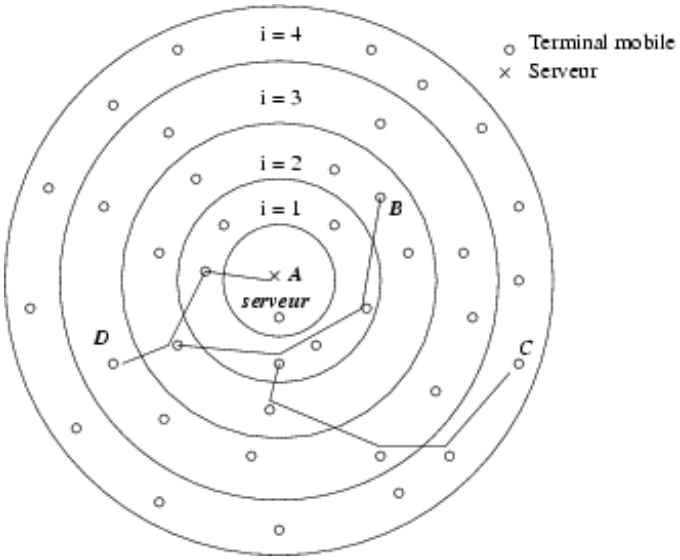


Figure 2-3 Distribution et accès aux ressources

Gestion de la cohérence des répliques

Afin de maintenir la cohérence des répliques des informations relatives aux ressources, trois approches principales ont été proposées : la première est basée sur la durée de vie des informations, la deuxième sur les rapports de validation, et la dernière sur des rapports

d'invalidation. La première approche vise à créer un faible couplage entre les fournisseurs de ressources et les entités conservant les informations y afférant. Le fournisseur affecte ainsi à chaque contenu une durée de validité. Cette approche est majoritairement choisie dans les réseaux *ad hoc* car elle n'induit pas de trafic supplémentaire. L'approche par validation propose de vérifier la validité des informations relatives aux ressources avant d'y accéder. Typiquement, cette vérification est assurée périodiquement par l'entité stockant ces informations. L'approche alternative est basée sur des rapports d'invalidation (*IR* pour *Invalidation Report*). Dans ce cas, un fournisseur envoie périodiquement des rapports d'invalidation (IR, *Invalidation Report*) dans lesquels sont inclus les informations ayant changées depuis le dernier rapport d'invalidation envoyé. Cette dernière solution est mieux adaptée pour des ressources caractérisées par un taux de modification peu important. Pour les ressources fréquemment utilisées, Cao [17] propose que des rapports appelés UIR (*Updated Invalidation Report*) correspondant à une petite fraction des informations essentielles soient envoyés plus souvent à l'utilisateur. Ces rapports correspondent aux ressources modifiées depuis le dernier rapport (ou UIR).

2.2.2.4 Synthèse

Nous avons présenté les protocoles de localisation de ressources proposés pour les réseaux *ad hoc* sans fil. Ces ressources peuvent aussi bien correspondre à des contenus qu'à des services logiciels ou ressources matérielles. Nous avons étudié ces protocoles en mettant en valeur les mécanismes proposés afin de réduire le trafic qu'ils génèrent. Ce critère d'évaluation est particulièrement important car la bande passante disponible dans les réseaux *ad hoc* est limitée. C'est pourquoi ce critère est généralement celui retenu pour la conception des protocoles pour les réseaux *ad hoc*. Toutefois, l'énergie consommée par les terminaux est aussi un facteur primordial dans l'évaluation de l'efficacité des solutions proposées. Nous présentons donc, dans la section suivante, une analyse de la consommation énergétique générée par les communications au-dessus des réseaux sans fil, qui inclut notamment l'accès au canal, l'acheminement de messages, et la localisation des terminaux et ressources associées. De plus, nous introduisons les travaux de recherche qui présentent des solutions visant à la réduire la déperdition énergétique des terminaux lors de cette communication.

2.3 Consommation énergétique induite par la communication

Les réseaux sans fil sont composés de terminaux utilisant des batteries comme source d'énergie. Le principal défi posé par cet environnement consiste à accroître l'autonomie des terminaux, c'est-à-dire, à diminuer leur consommation d'énergie tout en augmentant la puissance de leur(s) batterie(s). Cet objectif est d'autant plus difficile à atteindre que : (i) la miniaturisation des composants conduit à une miniaturisation des batteries et (ii) les nouvelles générations de terminaux intègrent des composants sophistiqués offrant des performances plus importantes mais, en contrepartie, consommant plus d'énergie. Par conséquent, l'énergie nécessaire au fonctionnement des terminaux a augmenté de 90% entre 1997 et 1999, alors que la capacité des batteries n'augmente que de 5 à 10 % par an [45,*46*]. La communication est la source prédominante de déperdition énergétique d'un terminal sans fil. Par exemple, le traitement de 3 millions d'instructions conduit à une consommation d'énergie équivalente à celle générée par l'envoi de 1Kb de données [66]. La gestion des communications est donc une source non négligeable d'économie d'énergie. Dans les sections suivantes, nous présentons les techniques utilisées afin de réduire la déperdition énergétique au niveau des communications. Plus précisément, dans la section 2.3.1, nous décrivons les techniques utilisées afin de réduire la consommation énergétique au niveau de l'accès au canal. Puis, dans la section 2.3.2, nous présentons les solutions proposées afin réduire les déperditions d'énergie liées au routage et au transport de messages.

2.3.1 Optimisation lors de l'accès au canal

Dans un réseau sans fil, qu'il soit basé sur une infrastructure ou *ad hoc*, l'une des principales sources de déperdition d'énergie au niveau de la gestion des communications est attribuée à la perte de paquets. Cette dernière est due à l'atténuation du signal sur les obstacles, à la réflexion du signal et à la réduction de la puissance du signal en fonction de la distance. Selon le mode de transmission utilisé, infrastructure ou *ad hoc*, les autres sources de consommation d'énergie diffèrent du fait que les terminaux constituant un réseau à base d'infrastructures communiquent en mode synchrone, alors qu'ils communiquent en mode asynchrone dans un réseau *ad hoc*. Un terminal se trouvant en mode infrastructure utilise un point d'accès fixe (*access point*) assurant la synchronisation des communications avec les terminaux. Cette synchronisation constitue la pierre angulaire permettant de limiter la consommation énergétique en mode infrastructure. En effet, grâce à ce mécanisme, un terminal reste en mode veille (et économise de l'énergie) et bascule en mode réception de façon périodique pour recevoir un message de balise envoyé par le point d'accès. Le point d'accès stocke les

messages reçus destinés aux terminaux qu'il couvre, en attendant de pouvoir les leur envoyer. Ainsi, les terminaux avertis par les messages de synchronisation demeurent éveillés seulement durant la période de temps nécessaire pour que la transmission des données puisse avoir lieu avec la station de base. Par ailleurs, cette coordination, assurée entre le point d'accès et les terminaux, et négociée par le point d'accès auprès des terminaux, permet de limiter les collisions. Un tel mécanisme ne peut toutefois pas être appliqué directement à un réseau *ad hoc* puisqu'aucune entité du réseau ne coordonne la communication. Par conséquent, en mode *ad hoc*, les sources de déperdition énergétique liées à l'utilisation d'un mécanisme de communication asynchrone proviennent :

- des collisions des paquets, ces derniers étant envoyés par plusieurs expéditeurs sur le même canal,
- de l'écoute quasi continue du canal de transmission,
- de l'envoi de messages de contrôle,
- de la surconsommation énergétique induite par le changement fréquent de mode de fonctionnement.

Nous examinons les différentes techniques visant à minimiser la consommation énergétique pour les cas que nous venons de définir.

Minimisation du taux de collisions

Afin de minimiser le nombre de collisions au niveau du trafic point-à-point, et la consommation énergétique en résultant, les protocoles MAC IEEE 802.11[7] et SMAC (Sensor-MAC) [91] proposent un mécanisme d'évitement de collision (*collision avoidance*). Avant d'envoyer un paquet, l'expéditeur écoute le canal, et si celui-ci est occupé, l'envoi est différé.

Pour réduire la probabilité de voir deux terminaux transmettre simultanément (puisqu'ils ne peuvent détecter les transmissions simultanées), les protocoles IEEE 802.11 (a, b et g) proposent un mécanisme dit de *virtual carrier sense*. Avant toute transmission, l'expéditeur diffuse un message de contrôle RTS (*Request To Send*), qui détermine quel est le destinataire et la taille des données échangées, cette dernière étant utilisée pour calculer le temps nécessaire à l'envoi du paquet. Le destinataire du message RTS répond en renvoyant un message de contrôle CTS (*Clear to Send*) indiquant si le canal est libre et donc si l'expéditeur peut envoyer les données. Lorsque l'expéditeur reçoit le message CTS, il envoie les données. Finalement, quand le destinataire a reçu le message proprement dit, il envoie un accusé de

réception sous la forme d'un paquet ACK (*ACKnowledgement*). Le tableau 2-2Tableau 2-2 fournit le coût énergétique pour l'expéditeur et le destinataire en fonction de la taille des messages. Avec le protocole IEEE 802.11, l'énergie consommée pour envoyer ou recevoir un paquet est donnée par l'équation linéaire suivante [75]: $\varepsilon = m \times taille + p$, où *taille* correspond à la taille du message, et m (respectivement p) détermine le coût énergétique incrémental (respectivement fixé) du message. Pour l'expéditeur, le coût élevé de m_{exp} est dû à l'émission du message de données. Le coût fixe p_{exp} résulte de la réception des deux messages de contrôle (CTS et ACK) et de l'émission du paquet RTS. Pour le destinataire A, le coût fixe p_{dest} est dû à l'émission de deux messages de contrôle (CTS et ACK) et par la réception du message RTS. La valeur m_{dest} pour A provient de la réception des données.

Contrairement aux protocoles MAC que nous venons de décrire, le protocole PAMAS (Power Aware Multi-AccesS) [77] utilise deux canaux séparés pour l'échange de messages de contrôle (c'est-à-dire, les messages RTS et CTS) et des messages de données. L'utilisation de deux canaux séparés limite ainsi le nombre de collisions entre les messages de données et de contrôle. Par conséquent, le nombre de collisions est réduit puisqu'un terminal peut à la fois recevoir ou émettre des données tout en interdisant aux autres terminaux d'utiliser le canal de transmission de données. Concrètement, quand un terminal A transmet un paquet à un terminal B, A envoie en parallèle un message RTS à B. Cependant, si un voisin de A ou B (ou des deux) reçoit ou transmet un paquet, il envoie un *busy tone* avec le message CTS envoyé en réponse. Ainsi, A ne transmet pas le paquet. Afin de réduire le nombre de collisions résultant de transmissions simultanées, le protocole EC-MAC (Energy Conserving MAC) [16] élit un coordinateur jouant le rôle de station de base. Les transmissions sont alors organisées par le coordinateur, en cycles, eux-mêmes divisés en plusieurs phases (voir figure 2-4). Au début d'un cycle, le coordinateur transmet un message de synchronisation de cycle. Ce message contient un certain nombre d'informations, comme des informations de synchronisation et d'ordonnancement de transmission. Dans la phase de *requête et de modification*, les terminaux transmettent des requêtes pour de nouvelles transmissions. La phase *nouvel utilisateur* permet aux nouveaux terminaux de s'enregistrer auprès du coordinateur. La phase de *synchronisation* est utilisée par le coordinateur pour diffuser un message de synchronisation contenant les intervalles de temps pendant lesquels un terminal peut échanger des données avec le coordinateur dans la *phase de données* suivante, chaque permission identifiant le terminal pouvant transmettre ou recevoir.

Tableau 2-2 Consommation énergétique liée à une communication point-à-point

Terminal	Consommation énergétique ($\mu W.sec$)	m ($\mu W.sec$)	p ($\mu W.sec$)
expéditeur X	$\varepsilon_{exp}=m_{exp} \times taille+ p_{exp}$	m_{exp}=1.9	p_{exp}=454
destinataire A	$\varepsilon_{dest}=m_{dest} \times taille+ p_{dest}$	m_{dest}=0.5	p_{dest}=356
Terminal non destinataire à portée de transmission			
à portée de transmission de l'expéditeur X et du destinataire A	$\varepsilon_{AX} = m_{AX} \times taille+ p_{AX}$	m_{AX}=-0.22	p_{AX}=210
à portée de transmission de l'expéditeur X	$\varepsilon_{X} = m_{X} \times taille + p_{X}$	m_{X}=-0.04	p_{X}=90
à portée de transmission du destinataire A	$\varepsilon_{A} = m_{A} \times taille+ p_{A}$	m_{A}=0	p_{A}=119

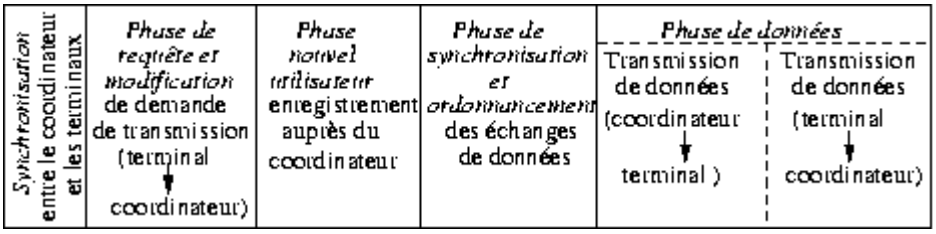


Figure 2-4 Phases de transmission du protocole EC-MAC

Minimisation du temps dévolu à l'écoute du canal de transmission

Afin de réduire la perte d'énergie, le temps dédié à l'écoute du canal doit être réduit au minimum. Bien que le protocole IEEE 802.11 réponde indirectement au problème de l'optimisation de l'énergie, la consommation d'énergie n'a pas été un critère fondamental lors de sa conception. Avec IEEE 802.11, les terminaux écoutent la plupart du temps le canal et restent dans le mode écoute (*idle*). En effet, seuls les terminaux à portée de communication

d'un expéditeur peuvent changer le mode de fonctionnement de leur interface, c'est-à-dire, faire passer l'interface réseau du mode écoute au mode veille durant le temps alloué à la transmission. Le tableau 2-2 présente le coût énergétique, pour un nœud qui n'est pas le destinataire du message du protocole IEEE 802.11, en fonction de la taille des messages. Les terminaux situés à portée d'émission d'un expéditeur reçoivent un message RTS et peuvent entrer dans un mode de conservation d'énergie durant la transmission. Ceci explique les valeurs négatives de m_{AX} et m_X, puisque l'énergie consommée est inférieure à celle du mode *idle*. Le coût fixe, p_{AX}, pour les terminaux non destinataires du message mais à portée d'émission de l'expéditeur est plus important que p_X, avec p_X correspondant au coût associé aux nœuds non-destinataires se trouvant seulement à portée de transmission de l'expéditeur. En effet, ces derniers ne reçoivent pas les messages CTS et ACK. Finalement, les terminaux, se trouvant seulement à portée du destinataire, reçoivent les messages CTS et ACK, mais ne reçoivent pas de message RTS. Ils ne peuvent donc pas entrer en mode de réduction de consommation d'énergie. Ceci explique pourquoi le coût incrémental m_A est nul. Par conséquent, tous les terminaux écoutent le canal de communication, à l'exception de ceux se trouvant à portée d'émission de l'expéditeur et du destinataire. Seuls ces derniers peuvent économiser de l'énergie en passant en mode veille. Le protocole PAMAS inclut la taille des données à la fois dans les paquets CTS et RTS. Les nœuds à portée de communication de l'expéditeur et/ou du destinataire peuvent par conséquent éteindre leur interface de communication pendant le temps de transmission. Contrairement au protocole IEEE 802.11, le protocole EC-Mac est basé sur une combinaison de processus de réservation et d'ordonnancement. Ainsi, les terminaux ne surveillent pas de façon continue le canal car ils connaissent la période de temps nécessaire pour transmettre et recevoir les données, ainsi que celle nécessaire pour se coordonner avec la station de base. L'approche alternative proposée par le protocole SMAC est basée sur la synchronisation des terminaux. Chaque terminal se trouve en mode veille et se réveille seulement afin de vérifier qu'aucun terminal ne désire communiquer avec lui. Cette solution nécessite une synchronisation périodique entre les terminaux voisins. Le protocole SMAC essaie de limiter la déperdition énergétique en acceptant une latence induite par le fait que les expéditeurs aient à attendre le réveil des destinataires avant d'émettre. Les concepteurs de SMAC justifient ce choix par le fait que ce délai peut être supporté par certains types d'applications et que par ailleurs, les capteurs d'un réseau sont caractérisés par de longues périodes d'inactivité durant lesquelles aucun événement ne se produit. Une caractéristique non négligeable des protocoles IEEE 802.11 et PAMAS est que les mécanismes utilisés pour limiter la consommation d'énergie n'affectent

pas leurs performances. Par ailleurs, EC-MAC respecte les conditions requises par un trafic multimédia.

Minimisation de la fréquence de changement de mode

Un terminal peut se trouver en plusieurs modes: éteint, veille, réception, émission. Le passage d'un mode à un autre induit un accroissement momentané de la consommation énergétique. Par ailleurs, ce passage requiert un certain laps de temps. Par exemple, le passage du mode transmission au mode réception prend de 6 à 30 ms, le passage du mode veille au mode émission prend 250ms. Par conséquent, la fréquence du passage d'un mode à un autre doit être minimisée, tout en assurant que les périodes de temps dévolues au mode veille soient maximisées.

Minimisation du nombre de messages de contrôles échangés

La dernière source de consommation d'énergie est attribuée à la réception puis à la destruction de paquets non utilisés, comme les paquets de contrôle et de coordination, par les nœuds qui n'en sont pas les destinataires et qui reçoivent ces paquets en raison de leur localisation. Cependant, ces messages sont nécessaires afin de limiter au maximum le nombre de collisions. Ce coût énergétique ne peut donc être évité.

2.3.2 Optimisation lors de l'acheminement d'informations

Deux étapes sont nécessaires pour acheminer un message entre deux terminaux distants. Dans un premier temps, le protocole de routage fournit le chemin entre la source et la destination. Puis, le message est transporté par les nœuds intermédiaires suivant les indications fournies par le protocole de routage. De la même façon, on distingue deux étapes fondamentales lors de la diffusion multi-points, c'est-à-dire, lors de l'émission d'un message d'une source vers plusieurs destinataires. La première correspond à la création de l'arbre de diffusion, et la seconde à l'acheminement du message de la source vers les terminaux appartenant à l'arbre de diffusion. Dans cette section, nous étudions l'énergie consommée et les techniques utilisées afin de réduire la déperdition énergétique lors de l'acheminement de messages point-à-point et multi-points, en décomposant cette étude selon les étapes que nous avons présentées: le routage, le transport et la diffusion.

Routage

D'après Cano *et al.* [16], l'énergie consommée par un protocole proactif est stable au regard du trafic. En contre partie, même si le réseau demeure inactif, en terme de trafic, les protocoles proactifs induisent une consommation énergétique continue. Une comparaison de la consommation énergétique des protocoles de routage réactifs est réalisée par Cano *et al.* [16] suivant différentes caractéristiques du réseau (taux de données échangées, nombre de terminaux, superficie du réseau). Ils observent qu'un nombre accru de sources conduit à une augmentation du trafic de contrôle du protocole de routage. Toutefois, l'énergie consommée suit une augmentation moindre que celle du nombre de terminaux. Ce comportement est dû au fait que les protocoles de routage réactifs apprennent les nouvelles routes à partir des paquets déjà envoyés.

Les protocoles de routage, qu'ils soient réactifs, proactifs ou hybrides, utilisent en général le nombre de sauts comme mesure du coût d'une route. Or, cette mesure est inadaptée puisque l'énergie consommée par les terminaux n'augmente pas proportionnellement avec le nombre de sauts. Pour cette raison, un certain nombre de solutions ont été proposées afin de réduire la consommation énergétique induite par le protocole de routage en ne sélectionnant pas obligatoirement le chemin le plus court. Les protocoles proposés configurent dynamiquement la puissance allouée à la transmission de telle façon que l'énergie consommée lors de la réception involontaire de messages par les terminaux non destinataire soit minimale. Pour cela, ils tirent partie du fait qu'un nombre plus important de sauts (mais de courtes distances) peut générer moins de déperdition énergétique qu'un nombre moins élevé de sauts (mais de longues distances). Doshi *et al.* [26] proposent de transmettre les messages en leur affectant le minimum d'énergie nécessaire pour qu'ils atteignent la destination. Ensuite, la route choisie est celle qui consomme le moins d'énergie pour arriver à destination. Cette solution est basée sur des protocoles existants tels que DSR ou AODV. La puissance minimum pour accéder au prochain saut de transmission est fournie par la couche MAC. La route sélectionnée est ensuite incluse dans le paquet de demande de route avec l'identification du terminal et est rediffusée sur un saut jusqu'à ce que le message atteigne le destinataire. Le destinataire inverse alors l'ordre de la route incluse dans l'en-tête de la requête de localisation, puis l'incorpore avec les valeurs les puissances d'émission relatives, dans le message renvoyé à la source. Ainsi, la source obtient la puissance d'émission nécessaire pour chaque saut à partir de la réponse, et calcule le coût énergétique pour ladite route.

Rodoplu *et al.* [71] supposent que chaque terminal possède un système de positionnement

GPS. La puissance de transmission associée à l'envoi est définie dynamiquement en fonction de la distance séparant l'expéditeur et le destinataire qui est fournie par le GPS. Comme ce calcul nécessite la connaissance de la localisation des nœuds les plus proches, chaque nœud diffuse périodiquement sa position. Comme avec Doshi *et al.* [26], l'idée principale est de trouver le plus court chemin, en terme de puissance utilisée, vers le destinataire. Les nœuds intermédiaires calculent le coût énergétique à l'expédition du lien et l'expéditeur d'origine choisit la route en évaluant la plus faible somme de consommations énergétiques. Par ailleurs, Rodoplu *et al.* réduisent l'énergie consommée en synchronisant les terminaux. Chaque terminal se réveille périodiquement pour recevoir ou émettre des messages et retourne en mode veille pour conserver son énergie. Cette synchronisation est rendue possible grâce au GPS qui fournit un système de temps globale TUC (*Temps Universel Coordonné*).

Une approche alternative ne nécessitant pas de récepteur GPS a été proposée dans [20]. Cette approche se base sur un algorithme aléatoire distribué, visant à minimiser le nombre de routeurs (c'est-à-dire de nœuds intermédiaires faisant suivre les messages), et mettant en mode veille les terminaux ne transmettant pas de messages. Pour ce faire, chaque terminal diffuse, sur un saut, un message *Hello* contenant l'état du terminal (routeur ou non-routeur). A partir de ce message, chaque terminal construit une liste des routeurs utilisés pour transmettre les messages, et des non-routeurs se trouvant en mode veille. Ainsi, le nombre de terminaux se trouvant en mode veille et participant involontairement à la communication (c'est-à-dire, recevant des messages qui ne leur sont pas destinés) est minimisé. Par conséquent, la durée de vie de tout le réseau est prolongée.

L'utilisation d'algorithmes de routage basés sur la sélection du lien optimal (en fonction du nombre de sauts ou de l'énergie dépensée par paquet) a toutefois des effets négatifs sur la durée de vie du réseau. L'utilisation accrue d'un petit nombre de terminaux pour acheminer les paquets au détriment de leur état énergétique conduit à vider leur batterie de leur énergie, et peut à terme induire la rupture d'un lien, et dans le pire des cas, générer un partitionnement du réseau. Pour résoudre ce problème, Chen *et al.* [20] proposent de vérifier que les terminaux partagent équitablement la charge de routage, et ainsi de changer de routeurs au cours du temps. Périodiquement, un terminal non-routeur détermine s'il doit devenir routeur; S'il s'aperçoit que deux terminaux ne peuvent s'atteindre directement ou *via* un ou plusieurs routeurs, il change d'état et passe à l'état de routeur. Afin de partager la charge de routage entre les terminaux, Shah *et al.* [73] affectent de façon récurrente différentes routes. Le

protocole de routage ne trouve pas une seule route mais plusieurs entre une source et une destination, et affecte à chaque route la probabilité d'être choisie. Ainsi, le protocole de routage réalise une rotation entre les différentes routes au lieu d'utiliser les mêmes terminaux se trouvant sur une route optimale.

Afin de réduire la disparité entre les niveaux d'énergie restant sur les terminaux, et donc de maximiser le temps à partir duquel une batterie sera épuisée, Gupta *et al.* [33] classifient les nœuds en trois zones: *normal*, *avertissement*, *danger,* qui correspondent respectivement à plus de 20 %, entre 10-20 % et moins de 10 %, de l'énergie maximale. Ce protocole affecte un coût d'utilisation énergétique à chaque zone. Par exemple, le coût correspondant au routage de l'information *via* un nœud se trouvant dans la zone danger ou avertissement est plus important que celui correspondant au routage de l'information *via* un nœud puissant en zone normale. L'idée principale est d'acheminer les informations à travers des nœuds ayant de bonnes capacités énergétiques restantes. Enfin, afin de limiter l'expansion de la consommation d'énergie induite par transmission d'un paquet à travers un nœud surchargé, Singh *et al.* [76] proposent une procédure qui envoie le trafic vers les nœuds ayant le moins de données à transmettre.

Transport

Lorsqu'une route a été découverte, le transfert des données est assuré par un protocole de transport. Un protocole de transport dit fiable est responsable de la vérification de la délivrance des données depuis la source vers la destination. Le protocole TCP (*Transmission Control Protocol*) [65] est le plus communément utilisé pour réaliser cette tâche. Ce protocole, créé pour les réseaux filaires, offre un niveau de performance relativement faible dans les réseaux sans fil. Plusieurs études utilisant des simulations [95,7], ou expérimentales [2], quantifient la consommation d'énergie de différentes implémentations du protocole TCP. Les résultats obtenus par Agrawal *et al.* [2] montrent que les faibles performances de TCP dans un réseau *ad hoc* multi-sauts sont principalement dues :

- aux cassures fréquentes des routes,
- aux délais excessifs lors de l'établissement des routes par les protocoles réactifs,
- à la surcharge du réseau induite par le trafic de contrôle des protocoles de routage.

Par ailleurs, Ahuja *et al.* [3] observent que les performances du protocole TCP varient en fonction du protocole multi-sauts utilisé. De façon générale, l'utilisation couplée de TCP et d'un protocole proactif offre de meilleures performances comparé à un protocole réactif. Toutefois, les faibles performances de TCP sont principalement dues au fait que TCP réagit de façon inappropriée aux pertes de messages [94]. Les réseaux sans fil souffrent de pertes de paquets dues à un taux d'erreur élevé, résultant des collisions et déconnexions, alors les réseaux filaires sont caractérisés par des pertes moins nombreuses et majoritairement attribuées à la congestion des liens. Il s'ensuit que dans ces deux types de réseaux, le protocole TCP réagit en faisant appel à des algorithmes de contrôle de congestion inadaptés aux réseaux sans fil. Pour résoudre ce problème, une solution vise à différencier les erreurs dues aux congestions des autres. Holland *et al.* [42] réalise cette distinction en utilisant un mécanisme ELN (*Explicit Loss Notification*) gérant l'envoi de notifications explicites. Ainsi, l'expéditeur n'invoque pas systématiquement des algorithmes de contrôle de congestion. Zorzi *et al.* [93] proposent se suspendre la transmission des paquets quand les conditions de communication deviennent difficiles (par exemple, lorsque le taux d'erreur ou le taux de bruit au niveau du canal physique atteint un certain niveau). Ensuite, les transmissions reprennent quand les conditions de communication sont de nouveau satisfaisantes.

D'autres solutions proposent d'adapter le comportement de TCP aux contraintes des environnements sans fil. Plusieurs techniques ont été proposées afin d'améliorer le mécanisme de recouvrement d'erreur. Elles se basent sur une ou plusieurs des approches suivantes :

- l'ajustement dynamique de la taille des fenêtres d'émission de l'expéditeur en fonction des conditions,
- la retransmission rapide des segments manquants,
- le réajustement dynamique du délai d'attente spécifiant quand un ensemble de paquets doit être renvoyé.

Dans le premier cas, Tsaoussidis *et al.* [80] observent que le contrôle de la congestion de TCP n'augmente pas assez rapidement la taille des fenêtres de transmission. En fait, la diminution de cette taille est faite rapidement par TCP alors que l'augmentation s'avère être particulièrement lente. Cette technique est efficace dans les réseaux filaires puisqu'elle permet de ne pas aggraver la congestion d'un routeur surchargé. Pour pallier ce problème dans un réseau sans fil, une approche judicieuse vise à augmenter rapidement la taille des fenêtres dès qu'une phase caractérisée par un fort taux d'erreurs prend fin [81]. Dans le deuxième cas, la méthode des SACKs (*Selective ACKnowledgment*) s'avère la plus adaptée au réseaux sans fil

et réduit efficacement la déperdition énergétique [74]. En effet, cette méthode permet d'identifier de façon précise quels sont les segments de données manquants (non reçus) et limite donc le nombre de retransmissions non nécessaires. Concrètement, le destinataire envoie un SACK correspondant à un accusé de réception dupliqué (c'est-à-dire, envoyé en plus des ACK classiques) lorsqu'il identifie qu'un segment de donnée est manquant. Cet accusé de réception indique quel est le dernier segment de donnée reçu avant le ou les segments manquants, et le segment immédiatement reçu après le segment manquant. Cette technique permet de notifier rapidement à l'expéditeur que des segments de données sont manquants avant que toutes les données contenues dans une fenêtre ne soient envoyées.

Diffusion d'informations relatives aux ressources

Partant du constat que des terminaux formant un réseau *ad hoc* ont des capacités énergétiques limitées et que ces derniers font souvent appel à la diffusion (par exemple, pour localiser des ressources ou des terminaux), un certain nombre de travaux se sont intéressés à la conception de protocoles de diffusion efficaces en terme de consommation énergétique. Les algorithmes en résultant visent à limiter l'énergie consommée par la diffusion de messages à partir d'une source. Pour cela, ils contrôlent la diffusion des messages en :

- paramétrant la puissance de transmission nécessaire à la source émettrice pour communiquer avec un terminal séparé par une distance de *d* mètres,
- définissant s'il est plus judicieux qu'une transmission entre deux terminaux distants soit effectuée en un seul saut (en augmentant la puissance de transmission) ou en plusieurs sauts (en réduisant la portée d'émission des nœuds intermédiaires).

Les algorithmes proposés se focalisent sur la création d'arbres de diffusion de type *broadcast* et/ou *multicast*, qui exploitent les propriétés de diffusion des réseaux sans fil tout en s'assurant que tous les nœuds à portée de communication de la source reçoivent le message. Nous décrivons dans ce paragraphe, les différentes approches utilisées afin de créer ces arbres de diffusion.

Afin de construire un arbre de diffusion, l'algorithme BIP (*Broadcast Incremental Power*) [89] se base sur l'algorithme *Prim* [67]. Il ajoute un à un les nœuds à cet arbre, comme décrit ci-après. Dans un premier temps, la source ajoute à l'arbre de diffusion le nœud qu'elle peut atteindre en dépensant le moins d'énergie pour transmettre. Cette opération est réitérée jusqu'à ce que tous les nœuds soient inclus dans l'arbre, c'est-à-dire, que la source détermine le nœud

non couvert dont l'ajout induit le coût incrémental minimum en terme d'énergie allouée à la transmission. Afin d'optimiser la procédure d'inclusion des nœuds dans l'arbre de diffusion, l'algorithme détermine quelles sont les transmissions redondantes, et donc non nécessaires. Concrètement, si un nœud E, appartenant à l'arbre de diffusion et ayant plusieurs parents P_1, .., P_i, .., P_n, est à portée d'émission d'un de ses parents P_i, alors P_i reçoit deux fois le même message. L'algorithme BIP diminue la puissance de transmission du nœud G qui est grand parent de E afin que P_i ne reçoive plus le message diffusé par G. Ainsi, le nœud P_i ne reçoit qu'un seul message : celui diffusé par E. Dans ce cas, l'arbre est modifié: P devient l'enfant de E. En d'autres termes, si un nœud E couvre un de ses parents P_i en re-émettant le message diffusé, alors, la puissance de transmission du grand parent de E peut être réduite afin que celui-ci ne couvre plus P_i. Ainsi, E reçoit le message d'un autre de ses parents, et P_i ne reçoit le message que de la part de E. Un certain nombre de propositions ont été faites afin d'optimiser l'algorithme BIP. Elles se basent sur le fait que BIP ne tire pas pleinement avantage des propriétés de diffusion d'un réseau sans fil en incorporant un à un les nœuds dans l'arbre de diffusion [86]. L'algorithme RBIP (*Reliable Broadcast Incremental Power*) [6] propose une version modifiée de BIP en ajoutant simultanément plusieurs nœuds à l'arbre de diffusion. Plus précisément, RBIP choisit une puissance de transmission suffisante pour atteindre les enfants. Dans les transmissions suivantes, la source choisit une puissance de transmission suffisante pour que tous les enfants n'ayant pas reçu le message (c'est-à-dire, ceux n'ayant pas renvoyé d'accusés de réception à la source) reçoivent le message. Cette étape est réitérée jusqu'à ce que le nombre d'accusés de réception reçu soit égal au nombre d'enfants. En parallèle, l'algorithme RBIP ajoute le coût minimum du lien (à l'ensemble des liens éligibles) de l'arbre existant. Cet ajout graduel de liens à l'arbre existant doit vérifier que le coût incrémental minimise le coût total de l'arbre.

L'algorithme IMBM (*Iterative Maximum-Branch Minimization*) [52] construit un arbre de diffusion BBT (*Basic Broadcast Tree*) basique, c'est-à-dire, ne prenant pas en compte l'énergie lors de la création de l'arbre. Ce n'est que lorsque l'arbre de diffusion est créé que l'algorithme modifie ce dernier afin d'exploiter les propriétés de diffusion des réseaux sans fil. Pour cela, il réduit la puissance de transmission de façon itérative tout en maintenant la connectivité (topologie) du réseau. Cet algorithme utilise alors les mêmes processus que ceux utilisés par l'algorithme BIP. Le processus MBR est utilisé pour définir si un nœud j doit être atteint en un seul ou deux sauts (*via* un nœud k) par la source i; C'est-à-dire si $d_{ik} + d_{kj} < d_{ij}$. Ensuite, l'algorithme vérifie la non *recouverture* de nœuds. Dans une deuxième étape, il

ajoute les nœuds minimaux. L'algorithme EWMA (*Embedded Wireless Multicast Advantage*) [14] crée un arbre MST (Minimum Spanning Tree) correspondant à un sous-ensemble de nœuds permettant de maintenir la connectivité du réseau. Cet arbre est obtenu en regroupant les nœuds proches et en limitant le nombre de liens entre les nœuds éloignées. A partir de cet arbre, l'algorithme définit un sous-arbre composé de nœuds dont la puissance de transmission minimise l'énergie totale résultant de la diffusion de messages dans l'arbre MST. L'algorithme RBOP (*RNG Broadcast Oriented Protocole*) [19] substitue le graphe MST au graphe RNG (*Relative Neighborhood Graph*) [79]. Ce choix est motivé par le fait que l'algorithme RNG ne nécessite pas d'informations globales relatives au réseau mais uniquement des informations locales correspondant à la distance séparant un nœud de ses voisins et la distance séparant ses voisins. Ces informations locales peuvent être obtenues grâce au GPS ou basées, par exemple, sur l'atténuation du signal entre deux nœuds. Finalement, lorsqu'un nœud reçoit un message d'un nœud voisin, il vérifie si le nœud fait partie du graphe RNG. Si c'est le cas, il renvoie le message aux nœuds n'ayant pas encore reçu le message en affectant à la transmission la puissance correspondante. Sinon, il définit un délai (puisque normalement, il devrait recevoir le même message d'un nœud faisant partie du graphe RNG). Puis, à l'expiration du délai, il rediffuse le message avec une puissance telle que les voisins, ayant déjà été atteint précédemment, ne reçoivent pas de nouveau le message.

2.4 Synthèse

Les réseaux *ad hoc* offrent une certaine flexibilité aux utilisateurs nomades, en leur permettant d'établir spontanément un réseau local sans recourir à une infrastructure existante. Dans un tel environnement, l'exploitation des ressources numériques du réseau nécessite deux étapes. La première consiste à découvrir quelles sont les ressources disponibles. La seconde correspond à interagir/accéder aux ressources et nécessite donc une localisation dynamique du terminal mettant à disposition la ressource, et ce, même si l'éloignement du terminal requiert une communication multi-sauts. Les protocoles de routage (multi-sauts) remplissent cette seconde tâche en gérant des tables de routage utilisées pour acheminer les messages vers les terminaux du réseau (voir § 2.2.1). La maintenance des informations de routage diffère selon le type du protocole. En effet, elle est effectuée de façon périodique pour les protocoles proactifs, est initiée sur demande pour les protocoles réactifs, et finalement dépend de la localisation des terminaux pour les protocoles hybrides. Il en résulte que les performances de

ces protocoles varient selon les caractéristiques du réseau *ad hoc* du point de vue de leur mobilité, densité, trafic, et superficie.

Les approches proposées afin de découvrir les ressources d'un réseau *ad hoc* adoptent quant à elles des modèles de communication pouvant être centralisé, de type *push* ou *pull*, ou encore basées sur une gestion de groupes (voir § 2.2.2). Une découverte centralisée est proposée dans les réseaux de capteurs, dans lesquels un (ou plusieurs) terminaux ayant des capacités suffisantes stockent les informations qu'il(s) collecte(nt) dans le réseau. Le principal défaut de cette solution est que la durée de vie des capteurs se trouvant dans l'entourage du terminal est réduite suite à l'afflux des messages collectés. De plus, le terminal qui collecte les informations constitue un unique point de cassure. Dans une approche de type *push*, les terminaux mettant à disposition des ressources, diffusent de façon proactive (et donc périodiquement et sans que les clients n'en fassent la demande) des informations relatives à leurs ressources. Dans une approche de type *pull*, les clients souhaitant découvrir les ressources offertes, diffusent sur le réseau une requête à laquelle répondent les terminaux concernés. Ces deux modèles de communication particulièrement populaires se basent sur des techniques des diffusions coûteuses. Elles conduisent donc à l'inondation du réseau si le nombre de clients ou ressources disponibles croit. Afin de contrôler la dissémination des messages, les modèles de communication proposés se basent sur des groupes constitués des clients et fournisseurs de ressources. Concrètement, un tel groupe correspond à un arbre dont la structure est utilisée pour propager les requêtes des consommateurs et les réponses des fournisseurs. En contre partie, la maintenance de cet arbre est particulièrement coûteuse puisque l'apparition, la disparition ou le déplacement d'un membre du groupe conduit à la modification de l'arbre de diffusion. Bien que des techniques complémentaires aient été proposées, notamment afin de réduire la quantité de messages transmis en les agrégeant, ou en répliquant les informations relatives aux ressources, il apparaît clairement que les modèles de communication utilisés par ces solutions à la localisation de ressources s'avèrent inadaptés. En effet, ils sont tout d'abord particulièrement coûteux en terme de trafic généré. Cela conduit d'une part à encombrer un réseau *ad hoc* déjà caractérisé par une bande passante particulièrement étroite, et d'autre part à augmenter significativement la consommation énergétique des terminaux. L'autonomie des terminaux et plus largement du réseau est donc diminuée.

Les travaux portant sur la conservation de l'énergie des terminaux sans fil ont principalement portés sur les couches basses du modèle OSI (voir § 2.3). Il s'ensuit qu'au niveau de la couche applicative, dans laquelle nous incluons notamment les protocoles de localisation de ressources et les systèmes assurant l'accès et la gestion des ressources, les solutions n'ont pas pris en compte les capacités limitées des terminaux. Ainsi, les capacités mais aussi l'hétérogénéité des terminaux ne sont pas exploitées pour adapter les stratégies de communication et de gestion des ressources des terminaux. Par ailleurs, les réseaux sans fil et notamment les réseaux *ad hoc* offrent des contextes d'exécutions particulièrement riches suite à la présence d'autres réseaux auxquels ils sont connectés par des passerelles. Or, les solutions à la localisation de ressources présentées ne permettent pas aux utilisateurs d'exploiter l'ensemble des ressources de l'environnement, ce qui inclut celles présentes sur d'autres types de réseaux.

Au regard des limitations des solutions existantes à la localisation de ressources dans un réseau *ad hoc*, nous avons étudié de nouvelles solutions adaptée à la structure des réseaux *ad hoc*. En particulier, la localisation de ressources n'est pas gérée de façon centralisée mais selon un modèle distribué de type collaboratif. De plus, la localisation prend en compte les capacités disparates des terminaux et leurs faibles ressources énergétiques, tout en exploitant la richesse du contexte d'exécution. Nous présentons dans la section 2.4.1 le contexte d'exécution qui nous intéresse, ce qui inclut une description précise du réseau sous-jacent. Puis, nous introduisons notre contribution qui est décrite plus avant dans les chapitres suivants.

2.4.1 Contexte d'exécution

Les avancées des technologies de communication sans fil et l'évolution des besoins des utilisateurs font que les terminaux sans fil (téléphones cellulaires, PDAs ou ordinateurs portables) disposent progressivement de plusieurs interfaces de communication offrant une connectivité de type IP. Concrètement, cela signifie qu'un utilisateur peut utiliser ces différentes interfaces afin d'exploiter les réseaux basés sur IP coexistant dans son environnement. L'ensemble de ces réseaux homogènes[9], coexistant dans un lieu donné forme un réseau composite que nous appelons *réseau hybride* (figure 2-5). Un utilisateur muni d'un terminal sans fil ayant une seule interface de communication connectée au réseau *ad hoc* peut

[9] http://standards.ieee.org.

également accéder aux ressources accessibles sur d'autres réseaux (sans fil, filaires, basés sur des infrastructures) en interagissant avec des terminaux appelés *passerelles*. Plus précisément, ces terminaux passerelles sont munis de plusieurs interfaces de communication et offrent une connectivité aux autres terminaux vers les réseaux disponibles.

Nous considérons ce type de structure de réseau dans notre étude pour les raisons suivantes. D'une part, le réseau considéré répond aux exigences des utilisateurs nomades en mettant à leur disposition un ensemble de sources d'information et de ressources numériques, à la fois plus important et plus varié. D'autre part, la présence de réseaux *ad hoc* dans ces réseaux hybrides permet aux utilisateurs nomades de pouvoir exploiter à faible coût (du point de vue financier) les ressources locales disponibles, et ce sans nécessiter d'infrastructures. Par ailleurs, afin de permettre aux utilisateurs, et ce malgré leur mobilité, d'accéder durablement aux ressources du réseau *ad hoc*, nous supposons que ces derniers disposent d'une communication multi-sauts en supportant un protocole de routage adéquat, qu'il soit proactif, réactif ou hybride, et géographique ou non. Toutefois, il est intéressant de noter que le choix d'un protocole de routage dépend directement du contexte d'application visé. Par exemple, si le nombre d'utilisateurs et de trafic transitant est faible, l'utilisation d'un protocole réactif est plus indiquée. En revanche, si le réseau est dense et le trafic important, un protocole proactif ou hybride est préconisé. La même problématique est partagée par les protocoles de localisation de ressources. En effet, les modèles de communications utilisés afin de gérer la découverte des ressources doivent être adaptés aux caractéristiques du réseau *ad hoc*, et cela afin de limiter le trafic généré par cette localisation. Ceci nous conduit notamment à aborder distinctement la localisation de ressources dans les réseaux *ad hoc* à petite et large échelle, comme motivé dans les deux sections suivantes.

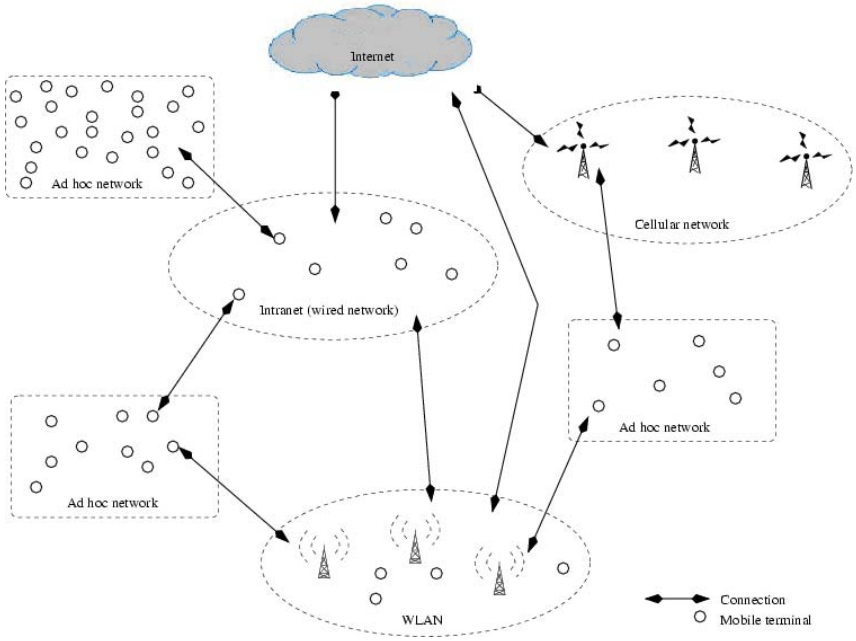


Figure 2-5 Réseau hybride

2.4.2 Localisation de ressources dans les réseaux à petite échelle

L'évolution rapide et la popularité des technologies sans fil auprès des consommateurs conduit à l'omniprésence d'appareils numériques sans fil (PDAs, ordinateurs portables, téléphones, capteurs) dans leur environnement quotidien. Cela nous pousse à envisager plusieurs scénarios d'utilisation des réseaux *ad hoc* de petite taille, c'est-à-dire, de la taille d'un réseau local. Considérons en particulier un utilisateur disposant d'un terminal numérique sans fil et souhaitant accéder aux ressources offertes par son environnement numérique proche, incluant les contenus mis à disposition. Cet environnement numérique correspond, par exemple, à celui d'un musée offrant à l'utilisateur une description de ses œuvres exposées, d'une maison communicante combinant différents équipements numériques comme des capteurs ou équipements grand public, d'un campus universitaire. Un autre exemple de scénario est celui de personnes collaborant de façon spontanée pour échanger des contenus, personnelles ou publiques, comme cela peut survenir lors de réunions de travail, ou de manifestation culturelles ou sportives pour échanger des contenus capturés au moyen des nouveaux dispositif multimédias communicants.

Dans les scénarios énoncés, l'utilisateur doit pouvoir exploiter les différentes ressources offertes par le réseau *ad hoc* local, que les ressources soient logicielles comme un contenu

numérique ou matérielles comme une imprimante. L'utilisateur doit également pouvoir profiter de la connectivité fournie par d'autres types de réseaux pour accéder à un ensemble plus vaste de ressource ou encore à des données personnelles qui ont une localisation fixe non nécessairement proche. Notre contribution vise à permettre à l'utilisateur de bénéficier de ces opportunités en introduisant un protocole de localisation des ressources numériques pour réseau *ad hoc* de petite taille, exploitant la présence de réseaux interconnectés (voir chapitre 3). Contrairement aux solutions existantes, notre approche ne se base ni sur une entité centrale pouvant être inaccessible et représentant un goulet d'étranglement, ni sur une inondation du réseau ou une gestion de groupe coûteuse lorsque les terminaux sont mobiles. Au contraire, nous proposons une solution entièrement distribuée qui limite le trafic généré par les requêtes de localisation en s'appuyant sur une collaboration entre des terminaux proches et ayant été sélectionnés au préalable. Par ailleurs, notre protocole de localisation de ressources tire partie de la présence d'infrastructures de communication sans toutefois y avoir recourt systématiquement. Plus précisément, afin de limiter et contrôler le trafic généré sur le réseau *ad hoc*, notre protocole réalise un compromis entre le trafic généré par la collaboration entre les terminaux du réseau *ad hoc* et celui généré par l'interaction avec les infrastructures accessibles à partir du réseau *ad hoc*. Notre solution est validée par une évaluation de l'énergie dépensée par les terminaux.

Par ailleurs, nous intégrons des caches au niveau du protocole de localisation de ressources pour le cas particulier où les ressources correspondent à des contenus. L'introduction de caches permet de réduire la surcharge du réseau puisqu'ils diminuent le nombre de messages de localisation et d'échange de contenus. De plus, l'introduction de cache atténue la charge des terminaux mettant à disposition des contenus puisqu'ils sont moins souvent sollicités. Elle diminue également le temps d'attente de l'utilisateur puisque les contenus deviennent accessibles localement. Finalement, l'intégration de caches accroît la disponibilité des contenus puisque ces derniers restent accessibles même si leur fournisseur ne l'est plus. Nous définissons en outre une stratégie de remplacement des éléments stockés dans le cache. Cette politique influe directement sur l'efficacité des caches lorsque ces derniers sont saturés. Les approches classiquement proposées (telles que LRU, LFU, FIFO) se basent sur le comportement de l'utilisateur afin de définir les éléments à sauvegarder dans le cache. Au contraire, nous proposons une stratégie hybride adaptée aux réseaux *ad hoc*, c'est-à-dire prenant en compte aussi bien le coût d'accès au contenu répliqué, que les caractéristiques de l'élément stocké, ou bien le comportement de l'utilisateur. Par ailleurs, nous proposons de

réduire le temps d'attente de l'utilisateur en préchargeant les contenus. Cela permet d'offrir à l'utilisateur un accès performant aux contenus numériques, c'est-à-dire rapide, adapté à ses besoins et aux ressources dont dispose le terminal, et ce malgré les déconnexions fréquentes et la faible bande passante disponible dans un réseau *ad hoc*.

Notre contribution comprend donc: un protocole de localisation de ressources dans un réseau *ad hoc* de petite taille, et une stratégie de gestion de caches et de préchargement pour optimiser l'accès au contenu dans les réseaux *ad hoc*.

2.4.3 Localisation de ressources dans les réseaux à large échelle

Les réseaux *ad hoc* sont utilisés pour des applications militaires et de secours en raison de leur déploiement rapide. Dans ce contexte d'application, les utilisateurs (tels que les secouristes) sont équipés de terminaux (portables, PDA). Ils ont besoin de localiser des sources d'informations numériques relatives, par exemple, à l'état du sinistre (foyers de destruction, blessés). Ils doivent également localiser les terminaux afin de connaître le déploiement actuel des effectifs (positionnement des équipes, secouristes, soldats, matériels). De plus, la découverte des ressources matérielles et des services numériques de l'environnement leur permet potentiellement de communiquer à l'extérieur du réseau. L'utilisation des ressources nécessite au préalable leur localisation dans le réseau *ad hoc* de grande échelle ainsi créé, c'est-à-dire, un réseau caractérisé par une superficie importante et une forte densité des terminaux et du trafic. L'exploitation effective du réseau *ad hoc* est difficile à réaliser en raison de ces caractéristiques intrinsèques comme les interférences radio, la mobilité des utilisateurs, et la faible bande passante. Par ailleurs, le contexte d'application considéré suppose en général la présence de peu d'infrastructures (ces dernières ayant été détruites).

Dans un réseau *ad hoc* de grande taille et de densité importante, une localisation efficace des ressources requiert de limiter au maximum le trafic généré sur le réseau. Nous proposons à cet effet un protocole de localisation de ressources semi-distribué (voir chapitre 4). Plus précisément, un ensemble de répertoires distribués dynamiquement et de façon homogène dans le réseau *ad hoc* gère la localisation des ressources pour les utilisateurs. Cela permet de limiter le nombre de terminaux coopérant dans le but de localiser les ressources. Indirectement, cela réduit le trafic généré en limitant la quantité de requêtes envoyées et de réponses renvoyées. De plus, cette localisation de ressources ne se base pas sur un seul

répertoire, pouvant être déconnecté et représentant un point de cassure. Par ailleurs, ces répertoires collaborent afin de cibler précisément l'envoi des requêtes de localisation vers les répertoires conservant les contenus demandés. Cela permet de localiser de façon efficace (c'est-à-dire sans recourir à des techniques de diffusion) les ressources disponibles. Nous proposons également une solution qui gère aussi bien la localisation de ressources au sein du réseau *ad hoc* que dans le réseau hybride. Elle peut, de plus, être facilement déployée, à la fois dans un réseau *ad hoc* et dans un réseau basé sur des infrastructures. Il en résulte que cette approche permet une localisation de ressources adaptée à un environnement hybride constitué de réseaux *ad hoc* ayant une forte densité et une faible connectivité.

Nous avons validé notre solution en réalisant d'une part une simulation du protocole nous permettant d'évaluer ses performances dans un réseau à large échelle. Par ailleurs, nous introduisons une mise en œuvre de notre protocole pour des ressources correspondant à des services Web mobiles, qui peut notamment être exploitée pour des applications de l'intelligence ambiante du fait du caractère diffus du Web (voir chapitre 5). A cet effet, nous présentons un logiciel libre sous licence LGPL, implémentant un prototype d'intergiciel assurant la localisation et l'accès aux services Web, éventuellement hébergés par des terminaux sans fil, dans un réseau hybride. Nous évaluons par ailleurs les performances de l'intergiciel par expérimentations.

Chapitre 3 Localisation distribuée pour réseaux ad hoc à petite échelle

3.1 Introduction

Dans ce chapitre, nous introduisons un protocole de localisation de ressources adapté aux réseaux *ad hoc* de petite taille. Notre but est de permettre aux utilisateurs de pouvoir localiser de façon dynamique et transparente les ressources disponibles sur le réseau en exploitant les infrastructures existantes et ce, sans nécessiter une configuration du terminal impliquant l'intervention de l'utilisateur. De plus, l'un des objectifs fondamental de ce protocole de localisation de ressources est de réduire le temps d'attente de l'utilisateur, malgré la bande passante restreinte et les déconnexions fréquentes, tout en limitant la consommation énergétique associée au processus de localisation. Pour atteindre ces objectifs, nous proposons un protocole de localisation de ressources distribué adapté aux réseaux *ad hoc*. En effet, de par sa structure distribuée, ce protocole permet de localiser les ressources disponibles, sans créer de goulet d'étranglement. Pour effectuer la localisation des ressources, les terminaux collaborent ensemble. Cette localisation tire parti des infrastructures disponibles en interagissant efficacement avec ces dernières, tout en pouvant être effectuée lorsqu'aucune infrastructure n'est déployée.

Afin de masquer les déconnexions et réduire le temps d'attente de l'utilisateur, nous proposons d'utiliser des caches ainsi que des techniques de préchargement pour le cas particulier où les terminaux accèdent à des contenus volumineux. L'utilisation de caches réduit le temps de localisation des contenus ayant été accédés récemment, tandis que le préchargement identifie les contenus non accédés récemment par l'utilisateur et susceptibles de l'intéresser. En raison des capacités limitées des terminaux et du réseau (en terme de stockage de données et de bande passante), la gestion des caches proposée intègre une stratégie d'identification des contenus devant être chargées ou remplacées. Le préchargement se base sur un algorithme de prédiction anticipant les futurs accès aux contenus. Il permet ainsi d'homogénéiser le trafic engendré par l'envoi des requêtes en les répartissant dans le temps.

A titre d'illustration de l'application de notre protocole, nous considérons son utilisation dans le contexte d'une visite interactive d'un site historique tel que le château de Versailles et ses environs. Des PDAs sont mis à disposition des visiteurs afin d'accéder aux ressources disponibles durant la visite. Dans cet exemple, les ressources correspondent à des applications interactives ou à des contenus tels que des pages statiques ou dynamiques résultant de l'interrogation de bases de données, ou encore des documents multimédias. En raison des faibles capacités de stockage des terminaux, seul un nombre restreint de ressources peut être conservé sur les terminaux. Or, l'interactivité et l'intérêt de la visite dépendent du volume, de la richesse et de la diversité des ressources mises à disposition. Dans ce contexte, l'utilisation de stations de base permet d'offrir aux terminaux un accès vers un nombre important de ressources. Or, en raison des contraintes financières et esthétiques, seul un petit nombre de stations de base sera déployé à des endroits clés de la visite. Par conséquent, les utilisateurs ne peuvent accéder aux ressources que lorsqu'ils se trouvent à portée de communication d'une station de base. De plus, le nombre restreint de stations de base conduit à la surcharge de ces dernières et à un délai d'attente accru pour les utilisateurs. Au contraire, notre protocole de localisation de ressources permet aux utilisateurs de pouvoir exploiter à la fois les ressources se trouvant dans un réseau *ad hoc* et celles accessibles à partir des stations de base. De plus, il limite la surcharge des stations de base en localisant les ressources se trouvant dans le réseau *ad hoc* sans recourir systématiquement aux stations de base pour accéder aux ressources. Enfin l'utilisation de caches et de techniques de préchargement permet de réduire le temps d'attente de l'utilisateur et la surcharge du réseau puisqu'ils diminuent le nombre de messages de localisation et d'échange de contenus.

La suite de ce chapitre se décompose de la façon suivante. Tout d'abord, dans la section 3.2, nous introduisons les principes de localisation sur lesquels s'appuie notre protocole de localisation de ressources. Puis, dans la section 3.3, nous définissons la notion de profils exploités dans le processus de localisation. Nous présentons ensuite dans la section 3.4 les politiques de gestion du cache et dans la section 3.5 les stratégies de préchargement. Enfin, nous proposons dans la section 3.6 une évaluation de la solution apportée avant de conclure dans la section 3.7 par un résumé de notre contribution.

3.2 Protocole de localisation

Nous présentons dans cette section une description de notre protocole de localisation de ressources. Celui-ci se base sur l'envoi de requêtes de localisation qui perment d'atteindre le terminal mettant à disposition la ressource recherchée. Ces requêtes contiennent le minimum d'informations nécessaires à la localisation de la ressource sur le réseau, comme par exemple l'identifiant de l'expéditeur et la description de la ressource recherchée. L'objectif principal de notre protocole est de réduire le trafic généré par l'envoi de ces requêtes puisque celles-ci représentent un coût important pour les terminaux, en terme de dépense énergétique et de bande passante du réseau. Notre protocole se base sur un système de coopération entre les terminaux appartenant au réseau *ad hoc*, et peut exploiter, lorsqu'elles sont présentes, les infrastructures réseau existantes (e.g., stations de base).Le terminal souhaitant accéder à une ressource, commence par la rechercher auprès des terminaux appartenant à son *N*-voisinage[10]. Pour cela, le terminal diffuse la requête de localisation à tous ces terminaux, puis étend sa recherche à l'extérieur de cette zone si nécessaire. Concentrer dans un premier temps la recherche aux seuls terminaux du voisinage permet de réduire le trafic généré. De plus, une diffusion est peu coûteuse (en terme de trafic généré) si elle se base sur un algorithme de diffusion adaptés aux propriétés radio des réseaux *ad hoc* [72], comme par exemple ceux des protocoles proactifs (e.g., OLSR - cf § 2.2.1.1) ou hybrides (e.g., ZRP- cf §2.2.1.3). Puis, afin de limiter le trafic associé à la communication avec des terminaux distants, c'est-à-dire se trouvant à l'extérieur de ce *N*-voisinage, seul un nombre restreint de terminaux est interrogé de façon ciblée (point à point). Ainsi, aucune requête n'est diffusée dans l'ensemble du réseau et seuls les terminaux partageant les mêmes centres d'intérêts, c'est-à-dire ayant des profils utilisateurs similaires (voir § 3.3) que le demandeur sont interrogés. Plus précisément, la communication point-à-point n'a lieu qu'avec les terminaux plus proches que la station de base. Cela permet d'exploiter les infrastructures du réseau disponibles afin de limiter le trafic résultant de la localisation sur le réseau *ad hoc*.

[10] La notion de *N*-voisinage utilisée par ce protocole, a été décrite dans la section 2.2.1. Elle fait référence à l'ensemble des terminaux accessibles en 1, .., N sauts, un saut étant défini par l'expédition d'un message à un terminal se trouvant à portée de transmission radio.

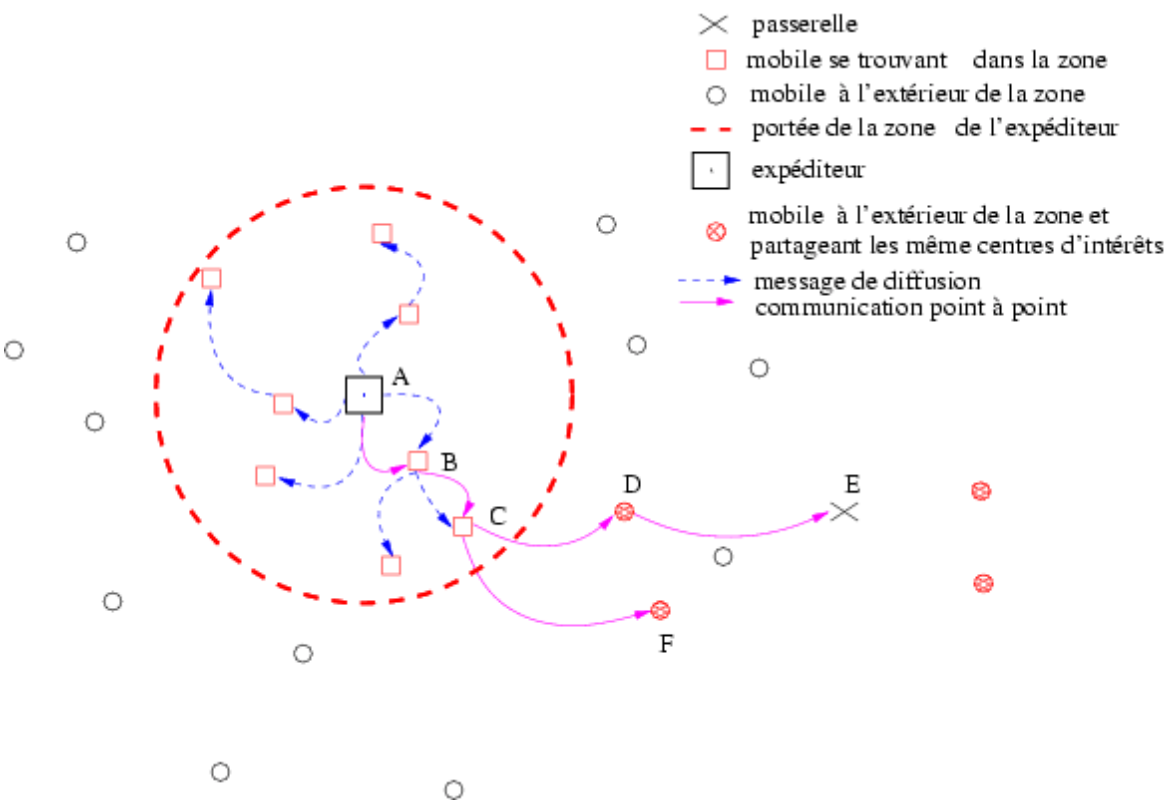


Figure 3-1 Localisation de ressources

La figure 3-1 illustre ce processus de localisation et de coopération entre les terminaux. Dans cet exemple, afin de présenter l'ensemble du processus de localisation, nous supposons que la ressource que souhaite accéder le terminal *A* n'est pas présente dans le réseau *ad hoc*. De plus, nous déterminons que la station de base la plus proche est accessible en *M* sauts (*M*=4), et que le *N*-voisinage est fixé à $N = 2$ sauts.

Pour commencer, l'ensemble des terminaux situés à une distance inférieure à *N* sont interrogés. Ensuite, les terminaux distants *D* et *F* sont contactés car ils partagent les mêmes centres d'intérêts (cf § 3.3) que *A*, et sont plus proches de *A* que ne l'est la station de base (c'est-à-dire que le nombre de sauts les séparant de *A* est inférieur à *M*). Finalement, la station de base *E* disponible dans le réseau est contactée.

Nous obtenons le protocole de communication suivant, pour accéder à une ressource *R* demandée par un terminal *A*.

- Si une station de base ou un terminal passerelle donnant accès à la station de base se trouve dans le *N*-voisinage de *A*, alors *A* lui envoie une requête. Sinon, *A* diffuse une requête pour *R* aux terminaux se trouvant dans le *N*-voisinage de *A*.
- Si aucun terminal se trouvant dans la *N*-voisinage de *A,* ne met à disposition la ressource *R*, alors, les terminaux interrogés sont ceux qui partagent des centres d'intérêts communs avec le terminal *A* et qui se trouvent à une distance inférieure à celle de la station de base la plus proche.
- La requête pour *R* est finalement acheminée vers la station de base ou la passerelle la plus proche.

Le coût associé à la localisation, en termes de dépense d'énergie et d'utilisation de la bande passante est minimisé, puisque lors de la localisation de la ressource *R*, la diffusion est restreinte au voisinage proche du terminal source. D'autre part, la communication point-à-point n'a lieu qu'avec les terminaux plus proches que la station de base et partageant les mêmes centres d'intérêts que le demandeur. Ces centres d'intérêts sont définis à partir d'un profil utilisateur et matériel, que nous définissons dans la section suivante.

3.3 Profils matériels et utilisateurs

Le protocole utilise deux types de profils, le *profil matériel* décrivant les caractéristiques physiques d'un terminal et le *profil utilisateur* définissant les centres d'intérêt de l'utilisateur du terminal, les centres d'intérêts d'un utilisateur étant définis par rapport aux ressources qu'il accède.

Les terminaux constituant un réseau *ad hoc* ont en commun le fait qu'ils utilisent une batterie comme source d'énergie. L'énergie disponible sur le terminal est donc un critère prépondérant puisqu'elle a un impacte direct sur l'autonomie des terminaux. Le profil matériel d'un terminal est constitué de l'énergie disponible sur l'hôte. Les profils matériels sont utilisés pour identifier le terminal, possédant des ressources énergétiques suffisantes, à partir duquel la ressource est accédée. Cela permet de maximiser la durée de vie du réseau en n'interagissant pas avec des terminaux dont les ressources énergétiques sont sur le point de s'épuiser.

Les profils utilisateurs sont utilisés afin de sélectionner les terminaux interrogés de façon ciblée, c'est-à-dire suivant une communication de type point-à-point. Nous exprimons ces profils sous une forme structurée et compacte permettant que (i) leur comparaison et que leur mise à jour soit effectuée de façon rapide et à faible coût (en terme de traitement nécessaire), et que (ii) leur dissémination soit peu coûteuse en terme de bande passante utilisée. Cela rend l'utilisation de profils adaptée aux contraintes des réseaux *ad hoc*. Nous définissons de quelle façon les profils utilisateurs sont générés dans la section 3.3.1. Puis, nous présentons comment les profils sont mis à jour dans la section 3.3.2, puis échangés dans la section 3.3.3. Enfin, dans la section 3.3.4, nous définissons les critères pris en compte pour (i) déterminer les terminaux interrogés de façon ciblée et (ii) sélectionner le terminal à partir duquel est accédé la ressource demandée.

3.3.1 Définition de profils utilisateurs

Le profil d'un utilisateur correspond à un ensemble de centres d'intérêt définis par rapport au contexte d'application du réseau *ad hoc* [103, 105]. Il est représenté par $P = <I_1, .. , I_r>$. Chaque centre d'intérêt est caractérisé par un ensemble de mots clés provenant d'un vocabulaire. Un ensemble de mots clés pour un centre d'intérêt I_j est noté $<m_{1,j}, .. , m_{nj,j}>$, avec n_j correspondant au nombre de mots clés composant I_j. Les vocabulaires proviennent d'ontologies dont la structure est définie selon le standard OWL[11] proposée par le W3C[12].

Le profil utilisateur est défini de façon à ce que l'importance relative (ou degré d'importance) d'un centre d'intérêt soit spécifiée. Cela est effectué en pondérant chaque mot clé caractérisant un centre d'intérêt (voir ci-après pour le calcul des pondérations). Le centre d'intérêt I_j du profil utilisateur P est représenté par la pondération de chaque mot clé le caractérisant :

$$I_j = <p_{1,j}, \ldots , p_{k,j, \ldots} , p_{nj,j}>,$$

avec $p_{k,j}$ désignant la pondération du mot clé $m_{k,j}$ (figure 3-2).
Deux paramètres sont utilisés afin de définir la pondération $p_{k,j}$ d'un mot clé $m_{k,j}$ de I_j [103] :

- la fréquence d'apparition $f_{k,j}^{d_i}$ du mot clé $m_{k,j}$ dans les méta-données d_i décrivant chaque ressource i, avec i ∈ {1, .., *No*}, *No* désignant le nombre de ressources ayant été accédés.

[11] http://www.w3.org/TR/2004/REC-owl-features-20040210/.
[12] http://www.w3.org.

- La fréquence de présence $df_{k,j}$ du mot clé $m_{k,j}$ parmi l'ensemble des méta-données caractérisant les ressources accédées. Cette fréquence est égale au rapport entre le nombre de ressources caractérisées par des méta-données dans lesquelles le mot clé $m_{k,j}$ est présent, et le nombre total de ressources accédées, noté *No*.

Plus précisément, la pondération $p_{k,j}$ du profil d'un utilisateur vérifie :

$$p_{k,j} = \sum_{i=1}^{No} f_{k,j}^{d_i} \times log(1/df_{k,j}).$$

Le facteur $log(1/df_{k,j})$ est défini de manière à mettre en évidence le fait qu'un mot clé, apparaissant souvent dans les méta-données d'un ensemble restreint de ressources, ne soit pas pénalisé par le fait qu'il n'apparaisse pas dans les méta-données de toutes les ressources accédées.

$$P = \begin{pmatrix} I_1 & \cdots & I_p \end{pmatrix} = \begin{pmatrix} p_{1,1} & \cdots & p_{1,r} \\ \vdots & & \vdots \\ p_{k,j} & & \\ \vdots & & \\ p_{n1,1} & \cdots & p_{nr,r} \end{pmatrix}$$

Figure 3-2 Profil utilisateur

3.3.2 Gestion des profils

Les centres d'intérêt d'un utilisateur évoluent au cours du temps, de nouveau centres d'intérêts apparaissent alors que d'autres disparaissent. Cette évolution des centres d'intérêts est prise en compte grâce à mises à jour du profil utilisateur effectuées lors de l'accès à une ressource. Les pondérations du profil sont augmentées suite à l'analyse des méta-données caractérisant les ressources accédées. Toutefois, cette mise à jour ne prend pas en compte la disparition des centres d'intérêt de l'utilisateur. Pour que le profil d'un utilisateur reflète ses centres d'intérêts actuels, et cela sans nécessiter le recalcule du profil, nous proposons de décrémenter graduellement et régulièrement les pondérations du profil. Par exemple, cette mise à jour peut être effectuée dès que *D* ressources ont été accédées. Lors de cette mise à jour, nous prenons en compte l'augmentation moyenne Δw des pondérations du profil résultant de l'accès aux ressources, depuis la dernière mise à jour, par rapport au nombre, noté *s*, de pondérations non

nulles du profil, et cela dans le but de diminuer graduellement d'autant l'ensemble des pondérations du profil. Plus précisément, chaque pondération $w_{j,k}$ du profil est décrémentée de la façon suivante:

$$w_{j,k} = w_{j,k} - \frac{\Delta w}{s \times \mathrm{D}}$$

Notons que si suite à cette diminution une pondération est négative, alors sa valeur est initialisée à zéro. Ainsi, au cours du temps, les centres d'intérêt passés de l'utilisateur, et qui n'en sont plus, ne sont plus représentés dans son profil. Ces profils à jour sont ensuite échangés entre les terminaux.

3.3.3 Échange des profils

Afin de pouvoir cibler les requêtes de localisation vers les terminaux partageant les mêmes centres d'intérêt, un terminal se base sur les profils utilisateurs qu'il conserve. A cette fin, chaque terminal du réseau conserve son profil utilisateur et des profils utilisateurs de terminaux partageant ou non les mêmes centres d'intérêt. Nous présentons ci-dessous de quelle façon *A* obtient ces profils. Nous distinguons deux cas, soit le terminal *A* est nouvellement connecté au réseau, soit il appartient déjà à ce réseau.

Un terminal *A* nouvellement connecté au réseau ne connaît pas encore les centres d'intérêts des autres utilisateurs. Il envoie donc son profil, *via* un message point-à-point s'il connaît un terminal proche, ou sinon *via* un message de diffusion aux terminaux *N*-voisins. Le (ou les) terminaux recevant le message stocke(nt) le profil de *A* puis lui réponde(nt) en renvoyant la liste des terminaux partageant les mêmes centres d'intérêt que lui, ces terminaux sont identifiés suivant la méthode présentée dans la section 3.3.4. Le terminal *A* conserve alors cette liste. Notons que si un seul terminal est interrogé, et si ce dernier ne connaît pas de terminaux partageant les mêmes centres d'intérêts que *A*, alors *A* diffuse ensuite le même message contenant son profil aux terminaux *N*-voisins. Ensuite, afin de limiter le trafic généré, les informations relatives aux profils sont incluses dans les messages échangés correspondant aux requêtes de localisation et aux réponses envoyées en échange. Cette inclusion dans les messages envoyés permet de limiter le nombre de messages de contrôle spécifiques envoyés dans le réseau afin de garantir la dispersion des informations relatives aux profils.

Lorsqu'un terminal *A* envoie une requête de localisation *via* un message de diffusion ou point-à-point (étant donné le protocole présenté dans la section 3.2), il inclut périodiquement dans le

message des informations relatives aux profils, qu'il stocke. Ces informations correspondent à son profil utilisateur mis à jour, à une liste des profils utilisateur des terminaux connus partageant les mêmes centres d'intérêts et une liste de profils utilisateur connus caractérisés par des centres d'intérêts différents. Ces derniers sont sélectionnés aléatoirement parmi la liste des profils d'utilisateur ne partageant pas les mêmes centres d'intérêts que *A*, et conservés par ce dernier. Un terminal *B* recevant le message stocke ou met à jour le profil de *A* selon le cas, les profils des terminaux partageant ou non les mêmes centres d'intérêts que *A*. Il en résulte que cette approche permet de garantir aussi bien la dispersion des informations relatives aux centres d'intérêts de *A*, que celles non partagées par *A*. Elle permet aussi de maintenir à jour les informations relatives aux profils. Enfin, le terminal *B* dispose d'une liste de terminaux partageant les centres d'intérêt de *A* lui étant utile s'il partage les mêmes centres d'intérêts que *A*. Plus précisément, lorsque cette requête est diffusée par *A*, les informations relatives aux profils sont incluses périodiquement. Ainsi, les terminaux interrogés peuvent mettre à jour le profil de *A* et cela limite la taille des messages envoyés. Lorsqu'il s'agit d'une requête de type point-à-point, ces informations sont incluses lorsque le terminal destinataire est interrogé pour la première fois, puis ensuite de façon périodique, et cela pour les mêmes raisons. Puis, le terminal *B* inclut dans sa réponse les mêmes informations, à savoir son profil, une liste des profils des terminaux partageant les mêmes centres d'intérêts que lui et des profils de terminaux choisis aléatoirement et ne partageant pas les mêmes centres d'intérêts. De la même façon cette inclusion est réalisée périodiquement. Quand *A* reçoit ces informations, il met à jour les informations relatives aux profils, qu'il stocke.

3.3.4 Utilisation des profils dans le processus de coopération

Un terminal *A* peut déterminer les terminaux partageant les mêmes centres d'intérêt à partir des profils utilisateurs qui lui ont été envoyés, et qu'il conserve (voir la section précédente). Cette détermination nécessite de comparer le profil *P* de *A* avec les autres profils utilisateur qu'il conserve, ces derniers étant notés P'_t (t = 1, …, *Nb),* avec *Nb* désignant le nombre de profils de terminaux que *A* conserve. En raison de la structure des profils utilisateur, cette comparaison peut être effectuée à faible coût, en terme de traitement (voir ci-après). La comparaison de *P* et P'_t consiste à générer un résultat définissant la similarité *Sim* (P, P'_t) existant entre les centres d'intérêts définis dans *P* et ceux dans P'_t).

Nous présentons dans l'équation 3.1 et ci-dessous de quelle façon le calcul de cette similarité

est effectué. Tout d'abord, étant donnés les centres d'intérêt I_j du profil P et I_j' du profil P'_t, avec (j=1, ...,r), pour chaque mot clé $m_{i,j}$ (i=1, ..., n_j) caractérisant I_j et I'_j, les pondérations $p_{i,j}$ de P et $p'_{i,j}$ de P'_t correspondantes sont multipliées. Puis, les résultats obtenus pour chaque mot clé sont additionnés ensembles. Ce résultat intermédiaire ($\sum_{i=1}^{nj} p_{i,j} \times p'_{i,j}$) indique s'il est grand, que les deux utilisateurs accèdent fréquemment tous les deux des ressources relatives au même centre d'intérêt, donné par I_j pour P et I_j' pour P'_t. Ensuite, les résultats obtenus pour chaque centre d'intérêt sont additionnés ensemble. Ainsi, plus les deux utilisateurs possèdent de centres d'intérêts communs et plus la valeur de la similarité est forte.

La similarité Sim (P, P'_t) entre deux profils P et P'_t correspond à la trace, notée TR, résultant du produit de la transposée de P, noté P^T, avec le profil P'_t et vérifie donc :

$$Sim(P, P'_t) = TR(P^T \times P'_t) = \sum_{i=1}^{r} \sum_{i=1}^{nj} p_{i,j} \times p'_{i,j}$$

Équation 3-1

Suite à la comparaison de son profils utilisateur avec celui de tous les terminaux dont il dispose, le terminal A peut classer les terminaux en fonction de la similitude existant au niveau des centres d'intérêts, entre son utilisateur et ceux des terminaux considérés. A partir de ce classement des terminaux, le terminal A peut donc efficacement déterminer quels sont les terminaux ayant les centres d'intérêts les plus proches. Afin d'augmenter l'efficacité de la sélection des terminaux, il se base en outre sur des statistiques qui indiquent :

- le nombre de fois où le terminal a été interrogé alors qu'il mettait à disposition la ressource demandée par A, par rapport au nombre total de fois ou il a été interrogé par A,
- le nombre de fois où le terminal désigné a interrogé A et auquel A a répondu de façon affirmative, par rapport au nombre total d'interrogations de A.

Les terminaux t se trouvant à l'extérieur du N-voisinage et plus proche que la station de base sont ainsi pondérés par une valeur

$$F = (Sim(P, P'_t) \times statiques) \div nbSauts,$$

avec *nbSauts* correspondant au nombre de sauts et obtenu à partir des tables de routage. Les terminaux pour lesquels la valeur de F est la plus grande sont prioritairement interrogés. Ce

processus est réitéré jusqu'à ce qu'un message stipulant que le terminal interrogé mette à disposition la ressource demandée, soit reçu, ou bien qu'aucun autre terminal ne soit éligible pour la requête.

En ignorant le cas où une station de base est accessible dans le *N*-voisinage de *A*, et étant donné le protocole de coopération que nous avons présenté dans la section précédente, une requête de localisation pour une ressource *R* qui n'est pas stockée localement est gérée de la façon suivante: (i) *A* diffuse une requête de localisation pour la ressource *R* dans son *N*-voisinage, (ii) si nécessaire, c'est-à-dire si la ressource *R* n'est pas mise à disposition par l'un des terminaux appartenant au *N*-voisinage, alors *A* itère l'envoi des requêtes pour *R* vers les terminaux se trouvant à l'extérieur de ce *N*-voisinage et qui sont plus proches que la station de base, ces terminaux étant ceux maximisant *F*. Finalement, la station de base est contactée en dernier ressort.

Examinons maintenant ce qui se passe lorsqu'un terminal *T* reçoit une requête de la part de *A* concernant la ressource *R*. Avant toute chose, le terminal commence par incrémenter le compteur d'accès de *A* utilisé ensuite lors du calcul de statistiques. Ensuite, s'il met à disposition la ressource *R,* et s'il accepte de coopérer, le terminal *T* renvoie une réponse à destination de *A*. Cette réponse contient les informations nécessaires à *A* pour effectuer la sélection finale du terminal à partir duquel la ressource sera accédée. Ces informations concernent d'une part un champ *TTL* (Time To Live) qui permet de fixer la durée de mise à disposition de la ressource, et d'autre part le profil matériel de l'hôte mettant à disposition la ressource. Ce profil matériel est constitué d'une valeur *Capacite* désignant l'énergie disponible sur le terminal. Lorsque le terminal ne met pas à disposition la ressource demandée, ou encore s'il refuse de coopérer, alors aucun message n'est renvoyé à *A* afin de ne pas surcharger le réseau. Nous n'utilisons pas de messages d'échec (de type *miss*) pour notifier qu'un terminal ne met pas à disposition la ressource demandée. Le terminal *A* fixe un délai au delà duquel l'absence de la ressource est avérée, et cela afin de limiter la surcharge du réseau et la consommation énergétique. La valeur de ce délai est fixée en fonction du plus grand nombre de sauts nécessaires pour interagir avec les terminaux auprès desquels la ressource a été demandée.

Lors de la réception d'un message de réponse, *A* met à jour les statistiques d'accès et les

profils correspondants. Le choix du terminal avec lequel *A* interagit pour accéder à la ressource s'effectue lors de l'expiration du délai que *A* a fixé. Pour cela, *A* sélectionne, parmi tous les terminaux ayant répondu à la requête, celui qui maximise la fonction[13] suivante:

$$Q = \frac{(\lambda \times TTL + \beta \times Capacite)}{\mu \times sauts}$$

Équation 3-2

Les facteurs λ, β et μ sont des constantes appartenant à l'intervalle [0 1]. Leurs valeurs sont fixées afin de favoriser l'accès à une ressource soit caractérisée par une durée de mise à disposition importante, ou une mise à disposition par un terminal proche, ou encore par un terminal ayant des capacités énergétiques suffisantes, et sont telles que $\lambda + \beta + \mu = 1$. Cette fonction, ainsi définie permet de prendre en compte les principaux critères de sélection d'un terminal, c'est-à-dire, sa capacité, la période de mise à disposition (ou période de validité) de la ressource, ainsi que sa proximité. Cependant, les métriques exprimant les capacités, la proximité et la période de validité n'étant a priori pas comparables, une nouvelle mesure basée sur un calcul de variance et de l'espérance est définie. Celle-ci correspond à une mesure unifiée utilisant la variance en temps que mesure de dispersion.

Nous illustrons le processus de calcul de cette mesure en nous limitant aux paramètres de la période de validité et de la proximité, la généralisation avec plus de deux paramètres étant triviale. Considérons que le terminal *A* reçoive *n* messages de réponse pour la ressource *R*. Pour illustration, la figure 3-3 donne un ensemble de valeurs de TTLs et de valeurs correspondant aux nombres de sauts, en considérant la réception de messages de réponse envoyés par les terminaux {*B*,..,*G*} au terminal *A*.

Le terminal *A* calcule l'espérance m_{TTL} (respectivement m_{saut}) du TTL (respectivement nombre de sauts) et la variance σ_{TTL} (respectivement σ_{saut}):

$$m_{TTL} = \frac{1}{n} \sum_{i=1}^{n} TTL_{mobilei}, \quad \sigma_{TTL} = \frac{1}{n} \sum_{i=1}^{n} [TTL_{mobilei} - m_{TTL}]^2$$

$$m_{saut} = \frac{1}{n} \sum_{i=1}^{N} saut_{mobilei}, \quad \sigma_{saut} = \frac{1}{n} \sum_{i=1}^{n} [saut_{mobilei} - m_{saut}]^2$$

[13] Dans le cas ou le terminal sélectionné n'est plus accessible, en raison, par exemple, de l'application d'une politique d'économie d'énergie, la requête est envoyée au prochain terminal éligible et cela jusqu'à ce que la ressource soit localisée.

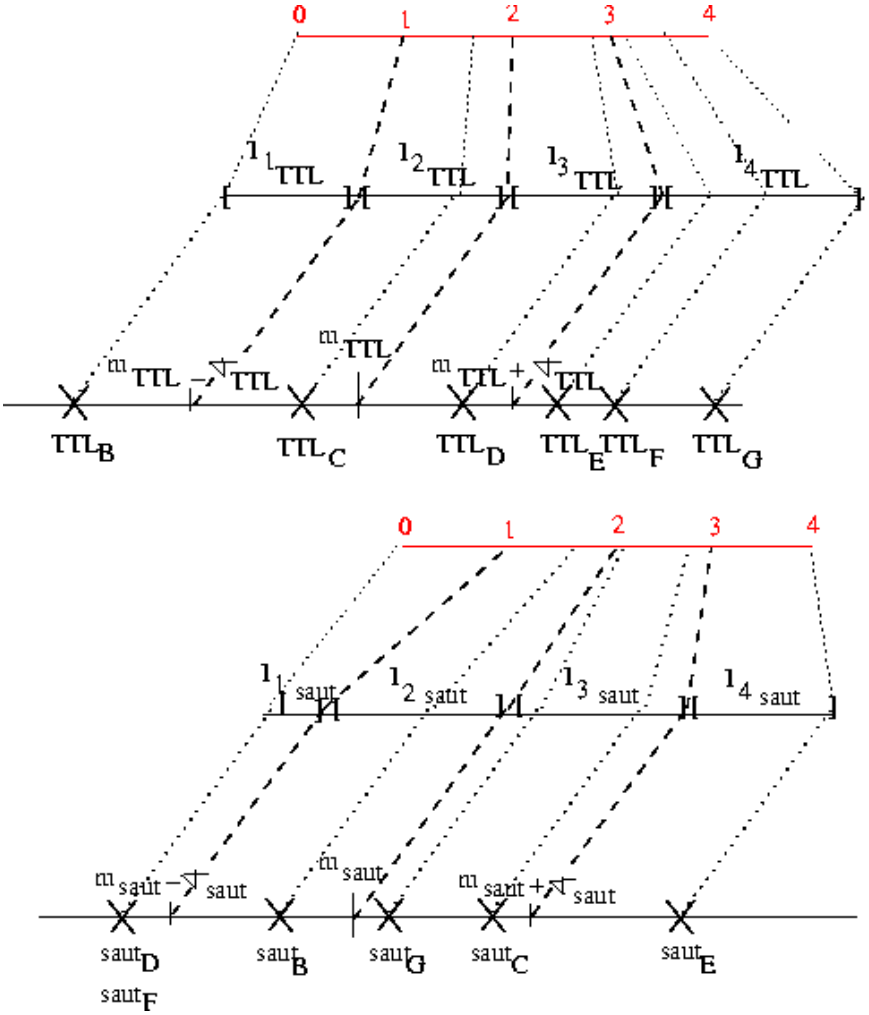


Figure 3-3 Distribution des valeurs

Puis, il définit quatre intervalles à partir de la valeur moyenne et de la variance des TTLs :

- $I_{1TTL} =]-\infty, m_{TTL}-\sigma_{TTL}]$,
- $I_{2TTL} = [m_{TTL}-\sigma_{TTL}, m_{TTL}]$,
- $I_{3TTL} =[m_{TTL}, m_{TTL}+\sigma_{TTL}]$,
- $I_{4TTL} = [m_{TTL}+\sigma_{TTL}, +\infty]$.

De la même, les quatre intervalles suivants sont spécifiées pour les sauts :

- $I_{1saut} =]-\infty, m_{saut}-\sigma_{saut}]$,
- $I_{2saut} = [m_{saut}-\sigma_{saut}, m_{saut}]$,
- $I_{3saut} =[m_{saut}, m_{saut}+\sigma_{saut}]$,
- $I_{4saut} = [m_{saut}+\sigma_{saut}, +\infty]$.

La valeur de chaque TTL fournie par les terminaux *B*,..,*G* est alors répartie sur quatre intervalles [0..1], [1..2], [2..3] *et* [3..4] (figure 3-3) suivant l'intervalle (I_{1TTL}, I_{2TTL}, I_{3TTL} ou I_{4TTL}) auquel chacun appartient. La même opération est effectuée afin de déterminer la valeur de chaque *saut* fournie par les terminaux *B*,..,*G*. A titre d'exemple, le tableau 3-1 donne des valeurs initiales de TTL et de nombres de sauts, ainsi que les valeurs obtenues avec la métrique unifiée, considérant la distribution donnée par la figure 3-3.

Les valeurs résultant de ce calcul sont celles utilisées dans le calcul de *Q* (voir sa définition dans l'équation 3.2). Ces valeurs sont donc non plus exprimées en fonction d'une métrique de temps ou de sauts, mais en fonction d'une métrique unifiée exprimée en fonction de la valeur moyenne et de la variance. La figure 3-4 présente la valeur de *Q* sans prendre en compte la valeur capacite ($\beta = 0$) en accord avec les valeurs de la figure 3-3, en fonction de la valeur de λ prenant des valeurs allant de 1 (*i.e.*, μ=0, et le coût de communication est ignoré) à 0 (*i.e.*, μ=1, et le coût de communication demeure le seul facteur sélectif).

Lors de l'expiration du délai que le terminal *A* a fixé, si des messages de réponse ont été reçus, la ressource est accédée à partir du terminal maximisant *Q*. Sinon, la prochaine itération du protocole de coopération est initiée.

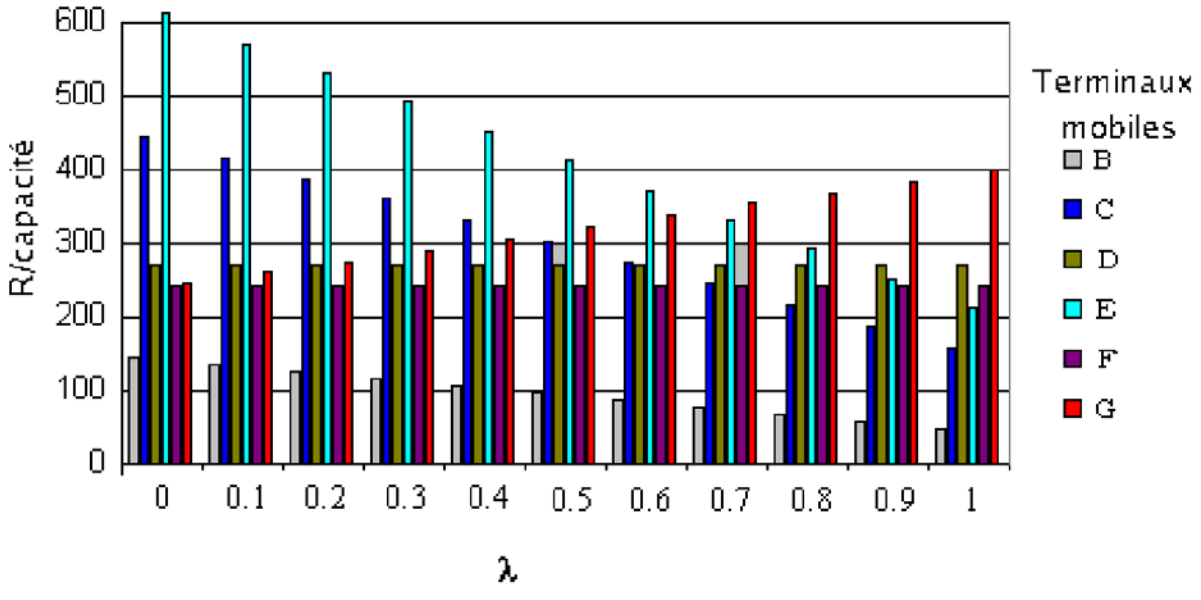


Figure 3-4 Fonction de répartitive Q

Tableau 3-1 Métrique de TTL et du nombre de sauts

terminal	TTL	TTL	métrique de TTL	nombre de sauts	métrique du nombre de sauts
b	½	$(m_{TTL}-\sigma_{TTL})$ -1/2σ_{TTL}	1-1/2= ½	4	1+1/5
c	4/7	m_{TTL}-3/7σ_{TTL}	2-3/7=11/7	8	25/7+2/5
d	27/10	$(m_{TTL}+\sigma_{TTL})$ -3/10σ_{TTL}	3-3/10 = 27/10	2	27/10
e	13/4	$(m_{TTL}+\sigma_{TTL})$+1/4σ_{TTL}	17/8	10	49/8
f	23/6	$(m_{TTL}+\sigma_{TTL})$+5/6σ_{TTL}	29/12	2	29/12
g	5	$(m_{TTL}+\sigma_{TTL})$+2σ_{TTL}	4	6	2+1/25

3.4 Intégration de caches utilisateurs

Nous enrichissons le protocole de localisation de ressources introduit dans la section précédente, afin qu'il intègre des caches utilisés dans le cas particulier où les ressources correspondent à des contenus volumineux. Ces caches conservent les contenus accédés par l'utilisateur. L'efficacité de ces caches dépend directement de la politique de gestion utilisée. Cette politique est constituée d'une stratégie de placement et de remplacement des contenus dans le cache. Plus précisément, lorsqu'un contenu précédemment sollicité est reçu par le terminal, l'algorithme de placement définit les critères devant être vérifiés par le contenu pour qu'il puisse être stocké dans le cache. Si le contenu doit être conservé et que sa taille excède la taille de l'espace libre du cache, alors l'algorithme de remplacement détermine le (ou les) contenu(s) devant être supprimés du cache afin de libérer l'espace nécessaire. L'objectif primordial d'un cache est de conserver les contenus qui seront réaccédés dans le futur. Cela implique d'autoriser leur stockage et de ne pas les supprimer du cache au profit d'autres contenus. D'autres critères peuvent être aussi utilisés (voir ci-dessous).

Nous définissons ci-dessous une politique de placement et de remplacement pour la gestion des caches dédiées à notre protocole de localisation. La politique de remplacement se base sur une fonction afin d'évaluer les contenus devant être expulsés du cache. Les critères que nous définissons sont plus particulièrement adaptés à l'environnement dynamique du réseau *ad hoc*.

Une politique de placement a pour but de filtrer les contenus rapatriés en n'autorisant la sauvegarde dans le cache que de certains d'entre eux. Les filtrages que nous appliquons sont les suivants :

- **Le filtrage en fonction de la taille du contenu** permet d'éviter la suppression d'un nombre important de contenus de petites tailles stockés pour les remplacer par un seul contenu de taille volumineuse. Dans ce cas, un seuil correspondant à la taille maximale des contenus pouvant être conservés temporairement, est fixé.
- **Le filtrage en fonction du type de contenu** est notamment utilisé pour ne pas stocker certain type de contenus. Les contenus particulièrement dynamiques et ayant donc une durée de validité très faible ne sont pas conservés. De plus, l'utilisateur définit les types et les caractéristiques des contenus ne pouvant être stockés dans le cache,
- **Le filtrage en fonction de la localisation du terminal** interdit de sauvegarder des contenus provenant de certains terminaux. Il est utilisé dans le but de ne pas stocker des contenus sauvegardés par des terminaux *N*-voisins.

Les techniques de filtrage utilisées visent à limiter le chargement de contenus dans le cache et donc à réduire le nombre de contenus devant être expulsés de ce dernier. Lorsque le cache est saturé, elles sont couplées avec des techniques de remplacement.

Les politiques de remplacement de caches ont faits l'objet de recherches approfondies. Il en résulte donc de nombreux travaux. Plusieurs critères ont été utilisés pour définir quand un objet doit être expulsé du cache. Ces critères sont:

- le caractère récent de l'objet,
- la fréquence d'accès à l'objet, exprimée en fonction du nombre de requêtes à l'objet,
- la taille de l'objet,
- le coût induit par le rapatriement de l'objet depuis sont serveur d'origine,
- le temps écoulé depuis la dernière modification de l'objet,
- la date d'expiration correspondant au temps à partir duquel l'objet est périmé.

Trois types de stratégies de remplacement, utilisant les critères ceux énoncés ci-avant, ont été proposées : (i) les stratégies aléatoires (ii) les stratégies se basant sur un seule critère, (iii) les stratégies hybrides se basant sur plusieurs critères. Les stratégies dites aléatoires choisissent aléatoirement le (ou les) objet(s) devant être expulsé(s) du cache. Le désavantage des stratégies aléatoires est que ces dernières sont difficiles à évaluer. En effet, une même simulation réalisée à partir d'une même trace, donne des résultats différents à chaque fois qu'elle est réalisée. Deux politiques de gestion de cache particulièrement populaires et basées sur un critère unique sont la stratégie LRU (*Least Recently Used*) et LFU (*Least Frequently*

Used). La stratégie LRU vise à expulser prioritairement les objets les moins récemment utilisés. La stratégie LFU a pour objectif d'expulser les éléments ayant été le moins souvent accédés afin de conserver les éléments les plus populaires dans le cache. La simplicité des stratégies basées sur un seul critère est à l'origine de leur principal inconvénient. Elles n'intègrent pas tous les paramètres importants susceptibles d'influencer les performances du cache. Afin d'enrichir ces stratégies, des stratégies dites hybrides ont été proposées. Il en résulte un nombre important de variantes se basant sur l'algorithme LRU et LFU. Les stratégies hybrides ont en commun le fait qu'elles définissent un critère primordial (e.g., la taille du contenu) pour sélectionner les objets à expulser. Puis, si plusieurs objets sont jugés équivalent par rapport au critère primordial défini, alors elles utilisent un second critère pour les départager. En d'autres termes, ce type de stratégie définit une hiérarchie de critères ordonnés afin de sélectionner les objets à expulser. Cette hiérarchie et son ordonnancement sont fixes et ne peuvent être changés.

A l'inverse, notre stratégie de remplacement définit une fonction de coût flexible se basant sur plusieurs critères, l'importance de ces critères pouvant être changée par l'utilisateur au cours du temps suivant sa situation physique. Cette stratégie de remplacement se base sur la pondération des contenus stockés. Ces derniers sont ensuite expulsés du cache en fonction de leur pondération respective. Plus précisément, ceux ayant la pondération la plus faible sont expulsés prioritairement du cache. Précisément, chaque contenu stocké dans le cache est pondéré en fonction des critères suivants :

- **Popularité**. La valeur *Popularite* est utilisée afin d'obtenir une approximation de la probabilité d'un accès futur, à la fois par le terminal lui-même mais aussi par les autres terminaux composant le réseau. Cette probabilité est exprimée en fonction du nombre de fois où un contenu a été accédé depuis qu'il est conservé dans le cache.
- **Coût d'accès**. La valeur *CoutAcces* sert à estimer le coût énergétique associé à l'obtention du contenu à partir d'un terminal distant, si ce dernier est expulsé du cache. Ce coût d'accès dépend de : la présence (ou non) d'une passerelle accessible dans le *N*-voisinage du terminal, de celle d'un terminal situé dans son *N*-voisinage conservant le contenu demandé, du fait qu'il soit nécessaire de communiquer avec un terminal distant, pour obtenir le contenu. Dans les deux premiers cas, le coût est relativement peu important, alors qu'il peut être très important dans le dernier cas. En effet, la valeur du coût d'accès est calculée en fonction de la consommation énergétique

associée à la communication dans le *N*-voisinage ou à l'extérieur de ce *N*-voisinage (comme cela est détaillé dans la section § 3.6).

- **Cohérence**. Un contenu est valide pendant une période de temps limitée. Cette dernière est connue grâce au champ *TTL*, défini par l'expéditeur du contenu et conservé dans le cache. Cependant, quand l'énergie restante sur le terminal est faible, il est plus judicieux de favoriser la sauvegarde de l'énergie au détriment de la cohérence du contenu. Ainsi, la valeur exprimant la cohérence est égale à $\nu \times TTL$, avec ν augmentant lorsque l'énergie diminue.
- La taille du contenu est prise en compte car l'ajout d'un contenu nécessite un espace de stockage important et conduit à la suppression d'un nombre important d'autres contenus plus petits.

Nous obtenons la fonction suivante pour pondérer un contenu du cache :

$$W = \frac{\alpha \times Popularite + \gamma \times Coherence}{\beta \times CoutAcces + \delta \times Taille}$$

Les valeurs de α, β, γ et δ sont fixées en fonction de l'importance respectives des critères associés à *CoutAcces*, *Popularite*, *Taille*, et *Coherence* dans le maintien d'un contenu dans le cache. La mesure utilisée pour combiner les différents paramètres est calculée de la même façon que pour la métrique utilisée pour calculer la fonction *Q* de la section 3.3.

L'utilisation d'un cache permet de réduire ostensiblement le temps d'attente de l'utilisateur puisque ce dernier dispose sur son terminal, d'un ensemble de contenus conservés dans le cache. L'efficacité du cache dépend d'une part de la stratégie adoptée pour le gérer et d'autre part de la taille allouée au cache. Afin d'augmenter le taux de *hit* du cache, et ce malgré leur taille limitée lorsque les terminaux ont des capacités réduites, nous proposons d'introduire une politique de préchargement des contenus dans le cache.

3.5 Intégration d'une technique de préchargement

Il existe deux types de préchargement. Avec le préchargement informé ou hors ligne (*off-line prefetching*), l'utilisateur informe le système quant à ses futurs accès. Avec le préchargement en ligne (*on-line prefetching*), le système se base sur les accès passés de l'utilisateur afin de

prédire ses accès futurs. Le préchargement hors ligne a particulièrement été étudié pour les environnements mobiles. Il consiste à mettre à la disposition de l'utilisateur une interface graphique lui permettant de sélectionner parmi un ensemble de contenus ceux qu'il souhaite précharger. Le préchargement en ligne a pour sa part été étudié dans le contexte des réseaux filaires. Il est soit orienté serveur, ou orienté client. Plus précisément, lorsque le préchargement est orienté serveur, ce dernier se base sur la popularité des contenus afin de définir ceux devant être préchargés. L'algorithme de préchargement orienté client se base sur l'historique des accès de l'utilisateur afin de générer un arbre de dépendance à partir duquel le système détermine les contenus à précharger. Le principal problème posé par ces deux approches [106,107,108] est que les contenus sont préchargés, mais sans prendre en compte le fait que le préchargement peut être effectué inutilement puisque : en raison de leur durée de validité limitée, les contenus peuvent ne plus être valides lorsqu'ils seront accédés par le client.

Nous proposons dans cette section une stratégie de préchargement en ligne. Cette dernière prend en compte les spécificités du terminal en terme de ressources disponibles et le trafic en généré par le préchargement. Notre stratégie de préchargement en ligne est de type client. Elle se base sur un algorithme de prédiction qui exploite un historique des contenus ayant été accédés précédemment. Cet algorithme de prédiction étudie le comportement des utilisateurs et la structure du contenu afin de mettre en évidence les dépendances existant entre les contenus et leurs ordres d'accès. Ces informations sont ensuite utilisées afin de prédire les contenus les plus à mêmes d'être accédés ultérieurement à un temps t. La stratégie que nous proposons diffère de celles existantes au sens où elle prend en compte à la fois la durée de vie du contenu (et donc son caractère dynamique), et les périodes de temps séparant les accès aux contenus (et donc le comportement usuel de l'utilisateur) afin de définir à quel moment un contenu doit être préchargé. Ainsi, les contenus ne sont pas préchargés s'ils ne sont plus valides lors de leur accès. De plus, afin de limiter à la fois la déperdition d'énergie des terminaux lorsque leurs réserves sont limitées et de ne pas augmenter le trafic sur un réseau déjà surchargé, cette stratégie a lieu seulement lorsque le terminal dispose d'une énergie suffisante (c'est-à-dire si cette dernière est supérieure à un seuil fixé), et lorsque le trafic[14] sur le réseau est faible. Cet algorithme de prédiction se base sur un graphe de dépendance

[14] perçu par le terminal de l'utilisateur.

établissant quels sont les contenus devant être préchargés. Nous définissons ci-dessous le processus de génération d'un graphe de dépendance.

Graphe de dépendance

Nous modélisons les séquences d'accès d'un utilisateur aux contenus sous la forme d'un processus (ou chaîne) de Markov d'ordre *k*. Ces séquences d'accès sont représentées sous la forme d'un graphe orienté et pondéré (figure 3-5), appelé graphe de dépendance, dont un nœud[15] correspond à un contenu et un arc détermine la relation de dépendance (définie par des accès consécutifs) existant entre les deux contenus liés *S* et *D*. Finalement, la pondération d'un arc correspond à un couple de valeurs désignant la fréquence d'accès au nœud (caractérisant) *D* et la durée moyenne entre l'accès au nœud *S* et au nœud *D*. La fréquence d'accès au nœud *D* sert à définir la probabilité que l'utilisateur accède le contenu *D* suite au contenu *S*. La durée entre l'accès au nœud *S* et au nœud *D* est utilisée afin de déterminer quelle est la période de temps pendant laquelle le contenu peut être préchargé. Ce graphe de dépendance est généré à partir de l'historique conservant les informations relatives aux accès de l'utilisateur sous la forme d'une liste de champs constitués chacun d'un identifiant du contenu accédé, de la date de son accès utilisée, afin de définir la période de temps entre les différents accès, et de sa durée de validité. Cette séquence d'accès est notée S =<A_1, .., A_n>.

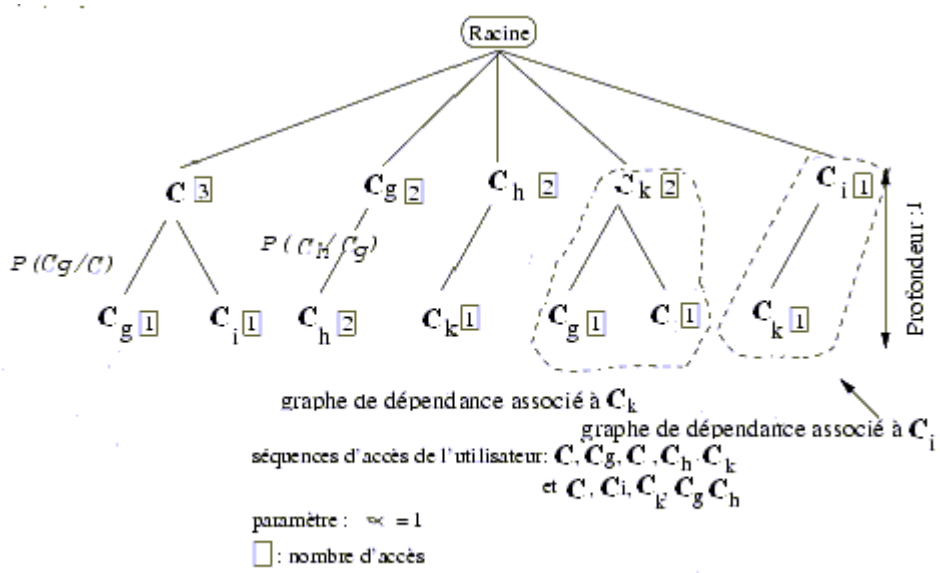


Figure 3-5 Graphe de dépendance de niveau 1

[15] Hormis le nœud racine.

La probabilité conditionnelle qu'une séquence d'accès $<A_{i+1}, ... A_n>$ ait lieu consécutivement à la séquence d'accès $<A_1, ..., A_i>$ vérifie :

$$P(A_{i+1}, ..., A_n / A_1, ..., A_i) = \frac{P(A_1, ..., A_n)}{P(A_1, ..., A_i)}, \text{ car } P(A_1, ..., A_i \cap A_{i+1}, ..., A_n) = P(A_1, ..., A_n)$$

Si la profondeur α du graphe de dépendance est inférieur à n, alors $P(A_{i+1}, ..., A_n / A_1, ..., A_i)$ est évaluée de façon approximative par :

$$\frac{P(A_1, ..., A_\alpha) \times P(A_\alpha, ..., A_n)}{P(A_1, ..., A_\alpha) \times P(A_\alpha, ..., A_i)} \text{ si } \alpha < i,$$

et est égale à :

$$\frac{P(A_1, ..., A_\alpha) \times P(A_\alpha, ..., A_n)}{P(A_1, ..., A_i)} \text{ si } i \leq \alpha < n.$$

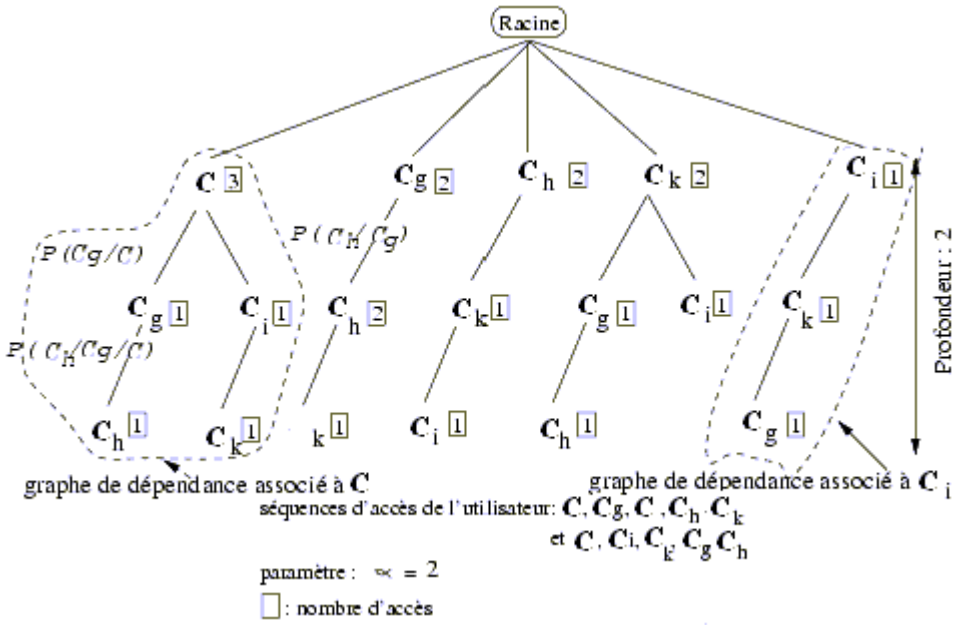


Figure 3-6 Graphe de dépendance de niveau 2

Les figures 3-5 et 3-6 illustrent ce cas en présentant des graphes de dépendance que nous appelons graphe *de niveau un* (α = 1), et *deux* (α = 2). Considérons dans un premier temps le graphe de niveau un et la séquence d'accès de l'utilisateur $<Ci, Ck>$. Le graphe de niveau un associé à *Ck* (c'est-à-dire dont la racine est *Ck*) fournit une probabilité d'accès identique et égale à 0.5 pour les contenus *Ci* et *Cg* suite à l'accès à *Ck* (d'après la figure 3-5). En effet, le graphe ne prend pas en compte le fait que l'accès suit un accès à *Ci* et donc que $P(Ci \cap Ck \cap Ci)$ est nulle (figure 3-6). Maintenir un graphe de niveau deux assure une prédiction plus efficace puisque cela fournit la valeur exacte de $P(Cg/Ck/Ci)$. Plus le niveau du graphe augmente et plus la prédiction est sure. Toutefois, concrètement, selon le contexte

d'application, l'efficacité du préchargement peut décroître lorsque la profondeur du graphe dépasse un certain niveau. La raison est que le nombre de contenus candidats au préchargement diminue fortement, ce qui aboutie à ne plus précharger de contenus. C'est le cas par exemple dans le contexte du Web. D'après des études expérimentales [24], il a été prouvé que le niveau du graphe de dépendance idéal est fixé à 3.

Dans le contexte des réseaux mobiles, certains terminaux ont des ressources de stockage limitées. Pour ce type de terminaux, le niveau de l'arbre est de 2. En effet, l'espace de stockage nécessaire dépend directement de la profondeur du graphe.

A partir du graphe de dépendance de niveau α, l'algorithme de préchargement définit quel contenu doit être préchargé. Le tableau 3-2 présente l'algorithme de préchargement. Le préchargement n'a lieu que si l'énergie disponible sur le terminal est suffisante (ligne 2). Quand un contenu A_s est accédé suite à une séquence d'accès $<A_1, ..., A_{s-1}>$, l'algorithme de préchargement extrait de cette séquence une sous-séquence $<A_{s-\alpha+1}, ..., A_s>$, avec α correspondant au niveau du graphe de dépendance. Si la séquence $<A_{s-\alpha+1}, ..., A_s>$ est conservée dans le graphe de dépendance ayant pour racine $A_{s-\alpha+1}$, alors, l'algorithme en extrait la liste des contenus éligibles notées $G_1, ..., G_j, ...G_p$ correspondant à ceux susceptibles d'être accédés prochainement (ligne 4). Il utilise aussi les informations contenues dans le graphe de dépendance afin de déterminer la probabilité $P(G_j/A_s,..., A_{s-\alpha+1})$ avec $(j = 1,..., p)$, que le contenu G_j soit accédé. Puis, l'algorithme détermine le contenu X dont la probabilité d'être accédé suite à la séquence d'accès $<A_{s-\alpha+1}, ..., A_s>$ est la plus forte (ligne 5). Le préchargement du contenu X peut alors être effectué si sa probabilité d'accès pondérée dépasse un seuil fixé (ligne 6). Plus précisément, nous disposons d'une période de temps (ou fenêtre) définie par la durée moyenne entre l'accès à X et à A_s, notée $TmoyenAcces(A_s->X)$, pendant laquelle le contenu peut être préchargé. Un contenu X ne être préchargé uniquement s'il est toujours valide lorsqu'il sera accédé par l'utilisateur, c'est-à-dire au moins jusqu'à la période de temps estimée pour son accès (ligne 7-8-9). Le cache dispose ainsi d'une fenêtre de préchargement ΔT qu'il utilise pour précharger les contenus dès que les conditions réseaux sont favorables (ligne 7-10).

Étant donné :	
les accès passés	$<A_1, .. A_{s-1}>$,
le graphe de dépendance	*GrapheDep*,
le niveau du graphe de dépendance	α
le contenu à précharger	*X*.
Durée de validité du contenu X	DuréeValidité(X)
Durée moyenne entre l'accès aux contenus A_s et *P*	*TmoyenAcces*(A_s->*P*))
Fonction qui extrait les nœuds fils du nœud A_s, le nœud A_s se trouvant dans le graphe de dépendance ayant comme racine $A_{s-\alpha+1}$ et comme chemin $A_{s-\alpha+1}$, ..., A_s	*ExtraireSousGraphe* (A_s, $<A_{s-\alpha+1}$, ..., $A_{s-1}>$)

[1] Réception(Requête(A_s))

[2] *si* (EnergieHote> *MinEnergie*)

[3] *si* ($<A_{s-\alpha+1}$, ..., *As*-1> $\in$ *GrapheDep*)

[4] [G_1, ..., G_p] = *ExtraireSousGraphe*(A_s, $<A_{s-\alpha+1}$, ..., *As*-1>)

[5] X = {G $\in$ [G_1, .., G_p]) / p(G/A_s) = $max_{G \in [G1, .., Gp]}$ ($p(G/A_s)$)}

[6] *si ((p(X/As) > seuil)*

[7] *AttendreTantQue*(ConditionPréchargementNonFavorables ET DuréeValidité(X) < *TmoyenAcces*(A_s->*X*))

[8] *si* (DuréeValidité(*X*) < *TmoyenAcces*(A_s->*X*))

[9] EnvoieRequête(*X*)

[10] f*in si*

[11] f*in si*

[12] f*in si*

Tableau 3-2 Algorithme de préchargement

3.6 Évaluation

Dans un réseau filaire, les performances d'un protocole sont typiquement évaluées en terme de charge du réseau (nombre de messages envoyés, taille des messages envoyés, bande passante utilisée). Un tel critère peut être utilisé afin d'évaluer les performances d'un protocole dans un réseau *ad hoc*. Toutefois, il n'est pas suffisant. Dans un réseau *ad hoc*, l'énergie disponible au niveau d'un terminal est un critère primordial, puisqu'un terminal n'est plus connecté au réseau dès que sa batterie est vide. Par conséquent, nous utilisons l'énergie dépensée comme critère d'évaluation. Lors de cette évaluation de la consommation énergétique des terminaux, nous considérons un réseau de 50 terminaux, et dont la distribution est uniforme sur la surface S avec $S = 300\ [m] \times 300\ [m]$. Nous utilisons des terminaux munis d'interface de communication de type 2.4 Ghz DSSS Lucent IEEE 802.11 WaveLan PC Bronze 2Mbps. Finalement, nous considérons le protocole de routage ZRP (voir section 2.2.1.3) en tant que protocole de routage sous-jacent. Il n'existe pas à l'heure actuelle de traces disponibles portant sur les accès à des ressources dans un réseau *ad hoc* sans fil. Pour cette raison, nous ne pouvons présenter d'évaluation de la politique de remplacement du cache et de préchargement. De plus, dans l'évaluation théorique de la consommation énergétique que nous présentons, nous ne considérons pas le coût associé au traitement des messages et des données puisqu'il est négligeable comparé au coût de la communication. En effet, rappelons que l'énergie consommée pour transmettre 1 Kb est approximativement la même que cette consommée pour exécuter 3 millions d'instructions [66]. Nous présentons l'évaluation de la consommation énergétique associée à la communication de la façon suivante. Dans un premier temps, nous étudions de quelle façon sont déterminés les coûts associés à la communication. En nous basant sur ce calcul, nous déterminons l'énergie consommée par le protocole de routage ZRP, puis par le protocole de localisation de ressources et finalement par le rapatriement dans le cas particulier où une ressource correspond à un contenu.

3.6.1 Energie consommée au niveau de la communication

En nous focalisant sur le coût énergétique associé à la communication, le coût induit par l'envoi d'un message correspond à la somme du :

- coût d'émission pour l'expéditeur,
- coût de réception pour le destinataire,

- coût de réception et d'émission pour les nœuds intermédiaires faisant suivre le message,
- coût de réception au niveau des nœuds non-destinataires (désignant des terminaux recevant les messages en raison de leurs localisations par rapport à l'expéditeur, aux nœuds intermédiaires et au destinataire), bien qu'ils ne participent pas directement à l'acheminement du message.

Terminal	Surface	Nombre de	énergie totale
à portée de communication		Terminaux	pour une zone
expéditeur X & destinataire A	$S_{AX} = \theta (r^2 - a_1^2) + \beta (R^2 - a_2^2)$	$n_{AX} = \frac{N S_{AX}}{S}$	$n_{AX} \varepsilon_{AX}$
expéditeur X	$S_X = \pi r^2 - S_{AX}$	$n_X = \frac{N S_X}{S} = \pi r^2 - \theta r^2 \sin^2 \theta + \beta R^2 \sin^2 \beta$	$n_X \varepsilon_X$
destinataire A	$S_A = \pi R^2 - S_{AX}$	$n_A = \frac{N S_A}{S} = \pi R^2 - \theta r^2 \sin^2 \theta + \beta R^2 \sin^2 \beta$	$n_A \varepsilon_A$

Tableau 3-3 Consommation énergétique associée à une communication

Dans le cas d'une communication de type point-à-point, tous les nœuds se trouvant à portée de transmission de l'émetteur *X* seulement, consomment le même montant d'énergie, noté ε_X (figure 3-7). Cela est dû au fait qu'ils reçoivent les mêmes messages de contrôle, à savoir, les messages RTS (voir section 2.3.1). De même, selon que les terminaux se trouvent à portée de transmission du destinataire *A* seulement, ou à la fois de *X* et de destinataire *A*, les terminaux reçoivent les messages de contrôle CTS et ACK, et respectivement RTS, CTS et ACK, et consomment donc le même montant d'énergie, à savoir ε_A et respectivement ε_{AX}.

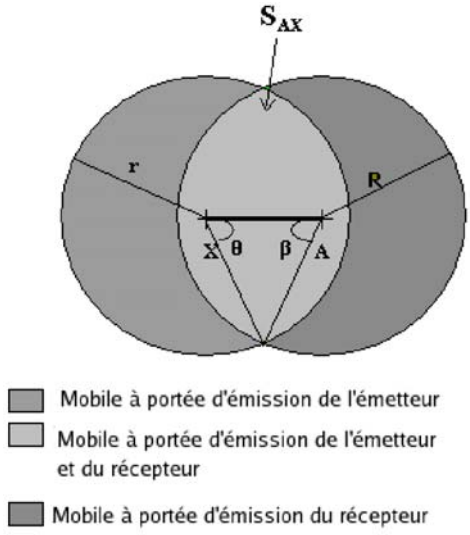


Figure 3-7 Terminaux à portée de transmission de l'expéditeur et du destinataire

Notons que l'énergie consommée associée aux communications point-à-point a déjà été introduite dans le tableau 2-2. L'énergie totale consommée par tous les terminaux suite à l'envoi d'un message de type point-à-point par l'expéditeur est égale à la somme de l'énergie consommée par chaque terminal selon sa situation géographique par rapport à l'expéditeur et au destinataire (figure 3-7). Ainsi, lors d'une communication de type point-à-point, l'énergie consommée par le réseau dans son ensemble correspond à la somme de l'énergie consommée par les mobiles environnant tous les expéditeurs et destinataires ayant participé à l'acheminement des messages (tableau 3-3).

Dans le cas d'une diffusion, tous les terminaux se trouvant à portée de transmission de l'expéditeur, ce qui inclut le destinataire et les nœuds non destinataires, consomment le même montant d'énergie (tableau 3-4).

Tableau 3-4 Consommation énergétique au niveau de l'expéditeur et du destinataire lors de la diffusion

Terminal	Consommation énergétique ($\mu W.sec$)	m ($\mu W.sec$)	p ($\mu W.sec$)
Expéditeur X	$\varepsilon_X = m_X \times \text{taille} + p_X$	$m_X = 1.9$	$p_X = 266$
Destinataire A & non destinataires à portée de communication de X	$\varepsilon_A = m_A \times \text{taille} + p_A$	$m_A = 0.5$	$p_A = 56$

La différence entre la consommation énergétique associée à la diffusion et à la communication point-à-point, est due à l'émission des messages de contrôle (RTS et CTS) et des accusés de réception (message ACK) dans le cas de la communication point-à-point. L'énergie dépensée par tous les terminaux du réseau suite à la diffusion d'un message est égale à la somme de l'énergie dépensée par les expéditeurs (incluant la source, et les terminaux intermédiaires) et de celle consommée par les terminaux se trouvant à portée de transmission de ces derniers (tableau 3-3). En se basant sur cette évaluation théorique de la consommation énergétique induite par la communication, nous présentons par la suite, la consommation énergétique induite par :

- l'envoi des messages de contrôle par le protocole de routage, afin de maintenir à jour les tables de routage sur chaque terminal en considérant plus spécifiquement le protocole ZRP (voir section 3.6.2),
- la localisation de ressources (section 3.6.3),
- le rapatriement des ressources lorsque ces dernières correspondent à des contenus (section (3.6.4).

3.6.2 Energie consommée par les messages de contrôle de ZRP

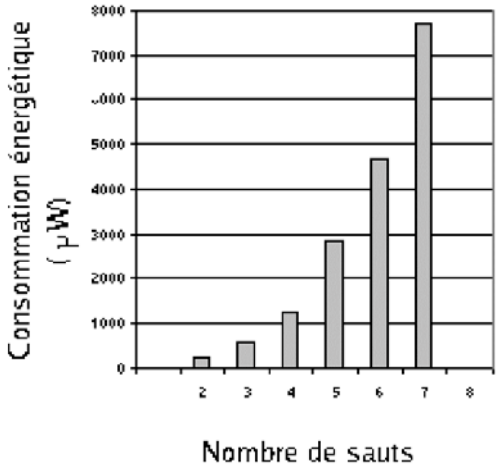


Figure 3-8 Consommation énergétique générée

par la réception du trafic provenant de ZRP dans un zone

Les performances de ZRP ont été évaluées dans [36,35] en estimant le trafic généré par ZRP. La figure 3-8 présente l'énergie consommée, dans l'ensemble du réseau considéré, par le trafic généré de façon proactive par le protocole ZRP en fonction du nombre de sauts définissant la taille de la zone proactive. Nous considérons que les messages de contrôle envoyés de façon

proactive afin de maintenir les tables de routages le sont avec une périodicité égale à 0.2/s. Étant donnée la consommation énergétique générée par ZRP, nous évaluons la consommation énergétique associée au protocole de localisation.

3.6.3 Consommation énergétique résultant de la localisation de ressources

Après avoir examiné la consommation énergétique associée au protocole de routage, nous étudions la consommation énergétique induite par notre protocole de localisation coopératif. Une requête de localisation pour une ressource inclut la diffusion de la requête (vers les terminaux se trouvant dans le *N*-voisinage de l'expéditeur) et la communication point-à-point (avec les terminaux partageant les mêmes centres d'intérêts). La Figure 3-9 fournit l'énergie consommée par le protocole de localisation coopérative en fonction du nombre de sauts définissant le *N*-voisinage. Plus précisément, la figure fournit le coût associé à la diffusion et la communication point-à-point avec un terminal situé à une distance qui ajoutant un saut par rapport au nombre de sauts définissant le *N*-voisinage. La consommation énergétique totale inclut la diffusion et la communication point-à-point. Celle-ci augmente avec le nombre de sauts définissant le *N*-voisinage (puisque plus de mobiles se trouvent dans le *N*-voisinage). Lorsque nous comparons la consommation énergétique provenant de la diffusion avec celle induite par la communication point-à-point, il en résulte que le coût ajouté par la diffusion est faible comparé au nombre de nœuds contactés.

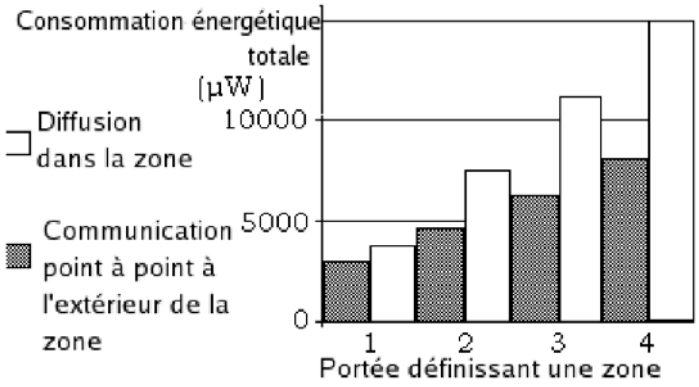


Figure 3-9 Consommation énergétique induite par la diffusion et la communication point-à-point

3.6.4 Consommation énergétique liée au rapatriement de contenus

Après avoir examiné la consommation énergétique associée à la localisation des ressources, nous étudions maintenant la consommation énergétique induite par le rapatriement d'une ressource correspondant à un contenu.

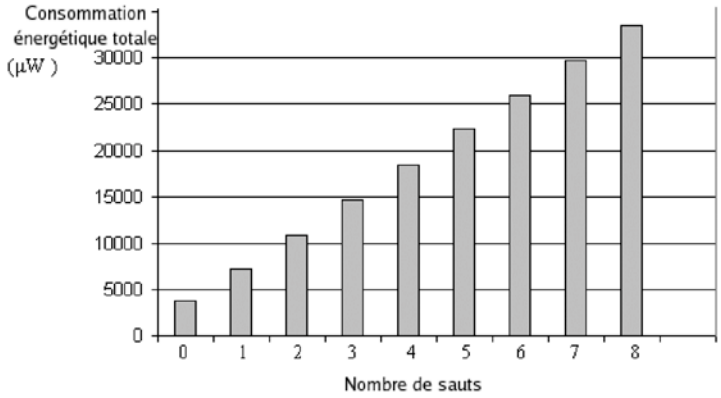


Figure 3-10 Consommation énergétique induite par le rapatriement des contenus

La figure 3-10 fournit la consommation énergétique associée à l'acheminement d'un message de 1 Kb vers un destinataire en fonction du nombre de sauts séparant la source du destinataire, ce qui correspond à la somme de l'énergie consommée par tous les nœuds participant directement ou indirectement à la communication. La figure démontre clairement que la consommation énergétique augmente proportionnellement avec le nombre de sauts. Cela provient de l'augmentation du nombre de réacheminement de messages et de la réception des messages de contrôle par les nœuds non destinataires.

Tableau 3-5 Énergie consommée en fonction de la densité des nœuds dans le réseau

Nombre de terminaux dans le réseau	50	70	90
Énergie consommée par les terminaux non destinataires (µ W)	1389	18959	2136
Ratio	9	7	6.11

Le tableau 3-5 évalue l'impact de la densité des nœuds sur la consommation énergétique. Pour un nombre constant de sauts égal à 3, nous voyons que l'augmentation de la densité des nœuds induit une consommation énergétique additionnelle pour les nœuds non destinataires.

Mais, le *ratio suivant :*

$$\frac{\varepsilon_{exp.} + \varepsilon_{dest.} + \varepsilon_{terminaux\ intermediaires.}}{\varepsilon_{totale\ non\text{-}destination}}$$

met en valeur le faible impact de la réception par les nœuds non destinataires, par rapport à la consommation totale d'énergie. Ainsi, pour 70 terminaux et un destinataire se trouvant à 3 sauts de l'expéditeur, l'énergie consommée par les nœuds non destinataires est environ 6 fois moins importante que celle consommée par l'expéditeur, le destinataire et les 3 nœuds faisant suivre le message.

3.7 Conclusion

L'évolution rapide et la popularité des technologies sans fil auprès des consommateurs conduit à l'omniprésence d'appareils numériques sans fil (PDAs, ordinateurs portables) dans le contexte quotidien d'une personne. Ces technologies sans fil et les réseaux *ad hoc* peuvent être exploités dans différents types de scénarios comme celui d'une visite guidée d'un site historique, ou encore celui d'un centre commercial ou même d'un campus universitaire dans lequel des magasins ou des départements mettraient à disposition des services numériques ou des contenus. L'exploitation des ressources dans un réseau *ad hoc* mobile de petite taille nécessite une localisation dynamique des ressources disponibles. Nous avons proposé dans ce chapitre un protocole de localisation de ressources exploitant la présence de réseaux inter-connectés. Contrairement aux solutions existantes, présentées dans le chapitre 2, notre approche ne se base pas sur une entité centrale pouvant être inaccessible et représentant un goulet d'étranglement. Au contraire, nous proposons une solution entièrement distribuée. Afin de limiter le trafic généré par la diffusion de requêtes de localisation dans tout le réseau, notre solution se base sur un système de coopération entre les terminaux. Plus précisément, un terminal souhaitant localiser une ressource collabore avec les terminaux proches en leur diffusant une requête. Puis, il dirige les requêtes vers les terminaux distants partageant les mêmes centres d'intérêts. Ainsi, la portée d'une diffusion est limitée et les interactions avec les terminaux distants sont effectuées de façon ciblée. De plus, ce protocole tire partie de la présence d'infrastructures de communication (sans toutefois y avoir recourt systématiquement) pour limiter et contrôler le trafic généré sur le réseau *ad hoc*. En effet, un compromis est réalisé entre le trafic généré par la collaboration entre les terminaux du réseau *ad hoc* et celui généré par l'interaction avec les infrastructures accessibles à partir du réseau *ad hoc*. Par ailleurs, si plusieurs terminaux interrogés mettent à disposition la ressource

demandée, le terminal souhaitant accéder à la ressource en sélectionne un à partir duquel il accède cette ressource. Ce choix se base sur plusieurs critères représentatifs des réseaux *ad hoc*, à savoir les capacités énergétiques du terminal, son éloignement (exprimé en nombre de sauts) et la durée de mise à disposition de la ressource.

Par ailleurs, deux autres contraintes des réseaux *ad hoc* doivent être considérées lors de la conception d'un protocole de localisation de ressources. Elles correspondent au manque de connectivité résultant notamment des changements de topologie du réseau et au temps d'attente de l'utilisateur. Lorsqu'une ressource correspond à un contenu volumineux, ces contraintes ont un impacte d'autant plus important puisque le temps d'attente de l'utilisateur induit par le rapatriement du contenu s'ajoute à la localisation du contenu.
A cet effet, nous avons intégré au protocole des caches conservant les contenus accédés et des techniques de préchargement. Ces caches réduisent la surcharge du réseau puisque moins de requêtes et de réponses sont émises. De plus, ils atténuent la charge des terminaux mettant à disposition des contenus puisque ces derniers sont moins souvent sollicités. Ils diminuent la latence de l'utilisateur puisque les contenus sont accessibles localement ou à partir de terminaux proches conservant dans leur caches ces contenus. Cela nous a conduit à proposer une stratégie de placement et de remplacement des contenus stockés dans le cache adaptée aux réseaux sans fil *ad hoc*. Cette politique influe directement sur l'efficacité des caches lorsque ces derniers sont saturés. Nous proposons une stratégie hybride adaptée aux réseaux *ad hoc*, c'est-à-dire prenant en compte aussi bien le coût d'accès à la ressource, que les caractéristiques du contenu, ou encore la probabilité de réaccès au contenu par l'utilisateur local ou par ceux du réseau avec lesquels il coopère. Finalement, nous proposons de précharger les contenus afin réduire le temps d'attente de l'utilisateur et de lui masquer les déconnexions intermittentes. Cette stratégie de préchargement prend en compte les capacités limitées des terminaux en adaptant le niveau du graphe de prédiction. Par ailleurs, elle permet d'offrir à l'utilisateur un accès performant aux contenus, c'est-à-dire rapide, adapté à ses besoins et aux ressources dont dispose le terminal de ce dernier, et ce malgré les déconnexions fréquentes et la faible bande passante disponible dans un réseau *ad hoc*.

Chapitre 4 Localisation semi-distribuée pour réseaux ad hoc à large échelle

4.1 Introduction

Dans ce chapitre, nous introduisons un protocole de localisation de ressources, adapté pour des réseaux *ad hoc* à large échelle. Les réseaux *ad hoc* à large échelle sont utilisés notamment pour des applications militaires et de secours en raison de leurs faibles coûts financiers et de leurs déploiements rapides, ne nécessitant pas d'infrastructures. Dans ce contexte d'application, les utilisateurs (tels que les secouristes) sont équipés de terminaux (portables, PDA). Ils ont besoin de localiser des ressources mettant à disposition des informations relatives, par exemple, à l'état du sinistre (foyers de destruction, blessés). Ils doivent également localiser aussi bien des ressources gérant le traitement et la génération de données relatives, par exemple au déploiement des effectifs (positionnement des équipes, secouristes, soldats, matériels, QG), que des ressources matérielles correspondant, par exemple, à des infrastructures de communication leur permettant de communiquer hors du réseau *ad hoc*. Toutefois, la localisation de ressources dans le réseau *ad hoc* est d'autant plus difficile à effectuer lorsque le réseau est caractérisé par une superficie importante et une forte densité des terminaux et du trafic. Ce contexte d'application à large échelle nécessite donc de fournir des mécanismes de localisation contrôlant le trafic généré par la localisation pour ne pas aggraver la surcharge du réseau. Une approche entièrement distribuée, basée sur la diffusion des requêtes et/ou des réponses dans l'ensemble du réseau, n'est en particulier pas envisageable techniquement puisque le trafic augmente de façon exponentielle avec le nombre de terminaux présents dans le réseau *ad hoc*. De plus, une localisation basée sur un groupe constitué des clients et fournisseurs de ressources, et dans lequel sont diffusées les requêtes et/ou réponses de localisation, n'est pas applicable en raison du nombre important de requêtes expédiées par les clients et de l'afflux des réponses renvoyées par les fournisseurs. En conséquence, nous proposons un protocole semi-distribué basé sur des répertoires correspondant à des terminaux gérant la localisation des ressources dans l'ensemble du réseau *ad hoc*. Cette approche est adaptée aux réseaux à large échelle car elle limite le nombre de

terminaux interrogés et réduit ainsi le trafic généré par la localisation. La distribution des répertoires garantit une localisation des ressources malgré le manque de connectivité, et une répartition homogène entre les répertoires du trafic induit par la recherche des ressources. Cette approche est en outre plus appropriée que celle basée sur un répertoire unique pouvant être déconnecté et représentant un goulet d'étranglement. Dans notre protocole, un client souhaitant accéder à une ressource interroge le répertoire voisin, celui-ci collabore alors avec les autres répertoires afin de localiser la ressource dans l'ensemble du réseau.

Par ailleurs, notre protocole garantit la localisation des ressources aussi bien dans le réseau *ad hoc* que dans l'ensemble des réseaux connectés à ce dernier. Il s'ensuit qu'un terminal bénéficiant d'une connectivité avec un réseau inter-connecté, peut accéder de façon transparente aux ressources présentes aussi bien dans le réseau *ad hoc* que dans les réseaux inter-connectés.

Nous exposons notre contribution de la façon suivante. Tout d'abord, dans la section 4.2, nous présentons les principes de localisation de notre protocole et décrivons précisément de quelle façon sont déployés les répertoires. Nous étudions de quelle manière ces répertoires assurent la localisation des ressources dans leur voisinage dans la section 4.3, puis dans l'ensemble du réseau *ad hoc* dans la section 4.4 et enfin dans le réseau hybride dans la section 4.5. Nous proposons ensuite une évaluation par simulation de notre protocole dans la section 4.6. Finalement, nous concluons par un résumé de notre contribution dans la section 4.7.

4.2 Principes de la localisation

Notre protocole de localisation de ressources se base sur un ensemble de répertoires distribués sur le réseau *ad hoc* et sur les réseaux inter-connectés, comme présenté dans la figure 4-1.

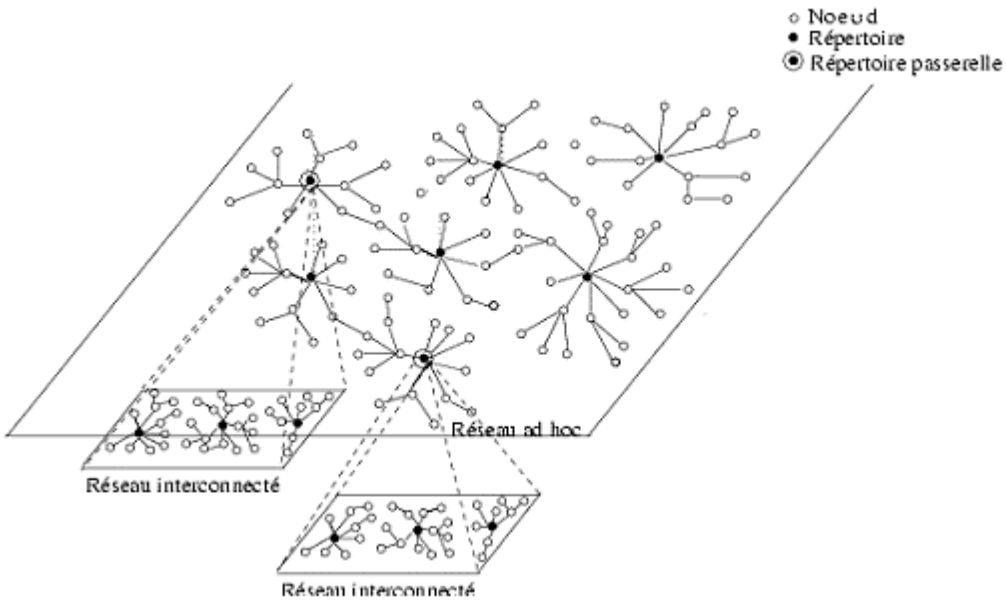


Figure 4-1 Architecture de localisation

Ces répertoires assurent la localisation des ressources pour les autres terminaux. Un terminal souhaitant accéder à une ressource, envoie une requête vers le répertoire le plus proche. Cela permet de limiter le trafic induit par la localisation puisque seul un terminal, à savoir le répertoire le plus proche, est interrogé par le client. Cette requête contient l'adresse IP de l'expéditeur et une identification de la ressource qu'il souhaite accéder. En échange, le répertoire interrogé lui renvoie les informations nécessaires pour que le client puisse accéder à la ressource, ce qui inclut notamment des informations concrètes portant sur la localisation de la (ou des) ressource(s) répondant aux exigences du client.

Par la suite, nous utilisons les termes de *zone de responsabilité* ou *zone de couverture* afin de faire référence au *N*-voisinage d'un répertoire. De même, le *rayon de couverture* ou le *rayon d'action* désignent la valeur prise par *N*.

Pour répondre aux requêtes des clients, chaque répertoire conserve les informations relatives aux ressources (identification et localisation) présentes dans son *N*-voisinage. Puis, lorsque ce répertoire est interrogé par un client, il collabore avec les autres répertoires afin de localiser la ressource dans l'ensemble du réseau, ce qui inclut le réseau *ad hoc* et les réseaux inter-connectés. Plus précisément, si la ressource demandée n'est pas présente dans le *N*-voisinage du répertoire, alors ce dernier fait suivre la requête vers un ensemble de répertoires conservant les informations concrètes relatives notamment à l'identification et à la localisation de la ressource recherchée. La sélection de cet ensemble de répertoires interrogés se base sur les

profils de ces derniers. Un profil correspond à une caractérisation de l'hôte et à un résumé compact de l'ensemble des informations abstraites stockées par un répertoire (voir § 4.4.1). Il en résulte que le trafic généré par l'interrogation est peu important puisque : (i) les requêtes ne sont envoyées qu'à un nombre restreint de répertoires sélectionnés, (ii) le nombre de réponses renvoyées est réduit du fait du nombre de répertoires interrogés.

En plus de sa zone de stockage, dans laquelle il conserve les informations relatives aux ressources se trouvant dans son *N*-voisinage, un répertoire dispose d'un cache conservant les informations relatives aux ressources se trouvant dans l'ensemble du réseau. L'utilisation de ces caches permet de répondre rapidement aux requêtes des clients et des autres répertoires. Ces caches réduisent aussi le trafic induit par la collaboration entre les répertoires, durant le processus de localisation d'une ressource, puisque les informations sont accessible à partir du cache du répertoire interrogé par le client. Ainsi, lorsqu'un répertoire reçoit une requête de localisation de ressources de la part d'un client ou d'un répertoire, il vérifie si les informations relatives à cette ressource sont stockées dans son cache, puis, si nécessaire, dans la zone de stockage.

Afin de garantir une localisation des ressources transparente, ne nécessitant pas l'interaction de l'utilisateur, les répertoires sont déployés dynamiquement par le protocole dans l'ensemble du réseau *ad hoc* (§ 4.2.1). De plus, pour garantir également une couverture complète du réseau par les répertoires et ce malgré les changements de topologie, le protocole assure un redéploiement dynamique des répertoires (§ 4.2.2).

4.2.1 Configuration du protocole

Afin d'assurer un déploiement homogène dans le réseau MANET, les répertoires sont élus dynamiquement par les terminaux. Ce processus d'élection se compose de deux étapes:

- l'étape d'*initialisation* correspond à la diffusion d'une requête d'élection et à la collecte des réponses,
- l'étape de *sélection* définit le processus de sélection, du (ou des) nœud(s) devant jouer le rôle de répertoire.

En raison de la bande passante réduite dans les réseaux *ad hoc*, l'objectif poursuivi, lors de la phase d'initialisation, est de limiter le trafic résultant de la diffusion de la requête d'élection.

Celui à atteindre lors de l'étape de sélection est d'identifier le terminal le plus à même de jouer le rôle de répertoire. Du fait des capacités limitées et de l'hétérogénéité des terminaux, le terminal élu comme répertoire doit disposer des capacités nécessaires pour gérer le traitement induit par la réception des requêtes de localisation. Nous considérons ici aussi bien les capacités de stockage, de traitement, que les ressources énergétiques. En effet, ces ressources énergétiques doivent être suffisantes afin que le terminal ne soit pas déconnecté prématurément. Par ailleurs, la situation géographique du terminal doit permettre de localiser des ressources pour des nœuds non encore couverts, c'est-à-dire n'étant pas déjà pris en charge par d'autres répertoires. Ces pré-requis sont nécessaires pour garantir une localisation de ressources efficace. Nous définissons dans les paragraphes suivant les étapes d'initialisation et de sélection d'un nœud devant jouer le rôle de répertoire tout en présentant de quelle façon ces objectifs (limitation du trafic généré et identification du terminal le plus à même de jouer le rôle de répertoire) sont atteints.

Initialisation de l'élection

Afin de se faire connaître auprès des clients et terminaux mettant à disposition des ressources, chaque répertoire diffuse une annonce à intervalle de temps régulier Δ_T dans son N-voisinage, la valeur de N étant affectée au répertoire à l'issue de son élection. Un terminal (à l'exception d'un répertoire) ne recevant pas ces annonces pendant une période de temps donnée par $\alpha \times \Delta_T$, avec $\alpha > 1$, est supposé ne pas avoir été couvert par un répertoire local. En d'autres termes, aucun répertoire, gérant la localisation des ressources pour ce terminal, n'est durablement présent.

Le terminal initie alors l'élection d'un répertoire. Afin que les terminaux, ne recevant pas cette annonce, ne commencent pas simultanément l'élection, chaque terminal fixe aléatoirement une valeur ε ($\varepsilon << \Delta_T$) à partir de laquelle il débute une élection. Toutefois, si deux terminaux initient ensemble un processus d'élection, alors, le terminal ayant l'adresse IP la plus petite continue le processus d'élection, alors que l'autre l'interrompt.

Le terminal débutant l'élection diffuse alors sur $E = 2 \times H$ sauts une requête d'élection et devient coordinateur de l'élection. La valeur de H est fixée comme décrit plus avant dans la section 4.6. Elle désigne la zone du réseau dans laquelle le répertoire sera élu. La requête est diffusée sur $E = 2 \times H$ afin de connaître le déploiement des répertoires se trouvant dans son

2*H*-voisinage. Cela est dû au fait que le terminal désigné comme répertoire aura, de part sa situation géographique, une zone de couverture pouvant atteindre des terminaux se trouvant dans le *E*-voisinage du coordinateur. Les informations concernant le déploiement seront utilisées afin de limiter le nombre de nœuds déjà couverts et aussi par le répertoire nouvellement élu.

Tous les terminaux ayant reçu une requête d'élection du coordinateur lui répondent en incluant une caractérisation de leur contexte réseau dans leur message. Le contexte réseau est utilisé afin de sélectionner le terminal ayant la position géographique la plus adaptée pour qu'il soit répertoire. Ce contexte inclut donc la liste des adresses IP des répertoires couvrant le terminal afin que le coordinateur sache si ce terminal doit être couvert (ou non) par le nouveau répertoire élu. Il inclut aussi la liste des terminaux 1-voisins à partir de laquelle le coordinateur définit la connectivité existant entre les terminaux.

Les terminaux *H*-voisins du coordinateur et susceptibles de devenir répertoire, répondent en incluant en outre dans leur message une caractérisation de l'hôte lui-même. La caractérisation de l'hôte est utilisée par le coordinateur afin de définir si le terminal a les capacités nécessaires pour gérer le rôle de répertoire. Elle correspond aux capacités de stockage, de traitement et énergétiques du terminal, cette dernière incluant le pourcentage d'énergie restant dans le terminal et le niveau de charge initial de la batterie. Puis, à partir des réponses renvoyées par les terminaux, le coordinateur sélectionne le terminal devant jouer le rôle de répertoire suivant l'algorithme défini ci-après.

Sélection du ou des répertoire(s)

Le tableau 4-1 , exprimé en pseudo code, présente le processus de sélection d'un répertoire par le coordinateur *e*. Le coordinateur se base sur les réponses envoyées par l'ensemble $N_E(e)$ des terminaux se trouvant dans son *E*-voisinage. A partir de ces réponses, le coordinateur *e* définit l'ensemble *C* des nœuds ayant les capacités suffisantes pour jouer le rôle de répertoire et se trouvant dans son *H*-voisinage (ligne 1). Plus précisément, il définit un niveau minimal, en terme de capacité de traitement, de stockage et de réserve énergétique, devant être vérifié par les terminaux. Notons que si ce niveau n'est atteint par aucun terminal, alors il réduit ces niveaux.

Puis, il détermine pour chaque nœud k susceptible de jouer le rôle de répertoire :

- L'ensemble $d_P^c(k)$ des nœuds non encore couverts par un répertoire, et, qui seront couverts par le nœud k si son rayon de couverture est égale à P, avec $P \in [0, H]$ (lignes 4, 6), (voir ci-après pour une description précise du calcul de $d_P^c(k)$)

- L'ensemble $c_P^c(k)$ des nœuds déjà couverts par un (ou plusieurs) répertoire(s) et couverts aussi par le nœud k si son rayon de couverture est égale à P, avec $P \in [0, H]$ (lignes 5, 7).

Le calcul de $d_P^c(k)$ (ligne 5) est effectué de façon récursive en se basant sur le fait que l'ensemble des nœuds appartenant au P-voisinage de k est constitué de l'ensemble des nœuds appartenant à son $(P-1)$-voisinage et de ceux accessibles en strictement P sauts. Dans notre cas cela signifie que l'ensemble $d_P^c(k)$ des nœuds, couverts seulement par k avec P comme rayon de couverture est constitué de l'ensemble des nœuds $d_{P-1}^c(k)$ couverts seulement par k avec P-1 comme rayon de couverture et de ceux non encore couverts et accessibles en strictement P sauts (cet ensemble de nœuds étant noté $d_p^{\boxdot}(k)$.

De même, le calcul de l'ensemble $c_H^c(k)$ se base sur le fait que l'ensemble des nœuds déjà couverts et couverts aussi par k avec P comme rayon de couverture est constitué de l'ensemble des nœuds déjà couverts et aussi couverts par k avec P-1 comme rayon de couverture et de ceux déjà couverts et accessibles en strictement P sauts (cet ensemble de nœuds étant noté $c_p^{\boxdot}(k)$.

Le coordinateur a donc déterminé l'ensemble des nœuds non encore couverts et déjà couverts, qui seront couverts par un nœud k s'il est élu répertoire et cela en fonction du rayon de couverture que le coordinateur lui assignera. Il réalise cette tâche afin de sélectionner le nœud offrant la meilleure couverture dans cette zone du réseau. En d'autres termes, le coordinateur vise à sélectionner le nœud qui, en raison de sa situation géographique, couvre le nombre maximal de nœuds non encore couverts par d'autres répertoires. En supposant que, plus le rayon d'action d'un nœud est important et plus le nombre de nœuds déjà couverts est important, nous choisissons de sélectionner en priorité, parmi les nœuds offrant la meilleure couverture, celui ayant le rayon d'action le plus faible.

Soit : e coordinateur ayant initié l'élection

E nombre de sauts sur lequel est diffusé la requête d'élection

N ensemble des nœuds du réseau *ad hoc*

$N_G(k)$ ensemble des nœuds pouvant être atteints en g sauts par le nœud k, $g \in [1,G]$

S ensemble des nœuds non couverts par un répertoire

ER ensemble des répertoires couvrant les nœuds participant à l'élection

$ER = \{ r \in N / \exists n \in N_E(e) \ni : n \in N_N(r) \}$

NC ensemble des nœuds couverts par un répertoire de ER

$NC = \{ n \in N_E(e) / \exists r \in ER \ni : n \in N_N(r) \}$

$|A|$ Cardinalité de l'ensemble A

max valeur maximale parmi une liste de valeurs

min valeur minimale parmi une liste de valeurs

extraire fonction qui extrait un élément d'un ensemble

[1] $C = \{ \forall i \in N_H(e) / e \in N_H \ et \ capacite(i) \geq minimum \}$

[2] Pour tout k appartenant à C

[3] Pour tout P appartenant à $\{0,\cdots,H\}$

[4] $d_P^c(k) = d_{P-1}^c(k) + d_p^{\boxdot}(k)$

[5] $c_P^c(k) = c_{P-1}^c(k) + c_p^{\boxdot}(k)$

[6] $avec \begin{cases} d_p^{\boxdot}(k) : \textit{nœuds non couverts atteignables en P sauts à partir de k} \\ d_p^{\boxdot}(k) = \{x \in d_P^c(k) / k \notin N_{P-1}(k)\} \\ d_P^c(k) : \textit{nœuds non couverts appartenant au P-voisinage de k} \\ d_P^c(k) = \{x \in S \cap N_P(k)\} \end{cases}$

[7] $et \begin{cases} c_p^{\boxdot}(k) : \textit{nœuds déjà couverts atteignables en P sauts à partir de k} \\ c_p^{\boxdot}(k) = \{x \in c_P^c(k) / k \notin N_{P-1}(k)\} \\ c_P^c(k) : \textit{nœuds déjà couverts appartenant au P-voisinage de k} \\ c_P^c(k) = \{x \in NC \cap N_P(k)\} \end{cases}$

[8] Fin pour tout

[9] Fin pour tout

// Calcul du nombre maximal de nœuds non couverts qui seront couverts

[10] $NbCouvert = max_{\,k \in C,\, h \in [0,\, H]} (|d_h^c(k)|)$

Détermination de l'ensemble des nœuds offrant une couverture optimale

[11] $G = \{ \forall (k, h) \in C \times [0, H] / \; d_h^c(k) = d \}$

// Calcul du rayon d'action minimal offrant une couverture optimale

[12] $MinR = min_{\,(k,h) \in G} (h)$

// Détermination de l'ensemble des nœuds offrant une couverture optimale

// et dont le rayon d'action est minimal

[13] $Gmin = \{ \forall (k, h) \in G / \; d_h^c(k) = MinR \}$

[14] $mincov = min_{\,(k,h) \in Gmin} (|c_h^c(k)|)$

[15] *Renvoyer* $N = MinR$ et $r = extraire (\{ \forall (k, h) \in Gmin / \; c_h^c(k) = mincov \}$

Tableau 4-1 Algorithme de sélection d'un répertoire

Pour cela, le coordinateur extrait parmi les nœuds pouvant jouer le rôle de répertoire celui offrant la meilleure couverture, (lignes 10--13), c'est-à-dire le nœud *r* ayant le rayon d'action *MinR* le plus faible (ligne 12,13) mais couvrant le nombre maximal *NbCouv* de nœuds non encore couverts par un autre répertoire (ligne 10). Par exemple, si deux nœuds *a* et *b* couvrent le même nombre de nœuds mais ont pour rayon d'action respectifs H_a et H_b avec $H_a < H_b$, alors *a* est sélectionné pour devenir répertoire. De plus, si plusieurs nœuds couvrent le nombre maximal *d* de nœuds non-couverts par un autre répertoire, tout en ayant le même rayon de couverture à *MinR*, le nœud couvrant le moins de nœuds déjà couverts est sélectionné (lignes 13-14) ; ce nombre de nœuds déjà couverts est donné par *mincov*. Ainsi, le nombre de nœuds couverts par plusieurs terminaux et recevant donc inutilement les annonces de ces répertoires, est réduit au minimum.

Lorsque le nœud *r* devant jouer le rôle de répertoire est sélectionné, le coordinateur lui envoie un message. Ce message contient la couverture *N* qu'il lui affecte.

La sélection du terminal appelé à être répertoire nécessite d'une part la diffusion sur un nombre fixé de sauts d'une requête d'élection par le coordinateur et la réception par ce dernier des réponses renvoyées par chaque terminal interrogé. Nous étudions dans le paragraphe suivant de quelle façon nous réduisons le trafic généré pour chacune de ces phases.

Réduction du trafic généré lors de l'élection

L'élection d'un répertoire se base sur : (i) la diffusion sur *H* sauts d'une requête d'élection puis (ii) sur le renvoi au coordinateur des réponses par les terminaux interrogés. La réduction du trafic généré se base donc sur une diffusion efficace des requêtes d'élection et sur l'agrégation des réponses, comme détaillée ci-après.

Dans le cas d'une diffusion aveugle (ou *blind flooding*), chaque nœud transmet un message à tous ses voisins. Ce processus est réitéré jusqu'à ce que tous les destinataires aient reçu et réexpédié une fois le message. Le nombre de paquets reçus et redondants augmente lorsque le nombre de voisins croît. Par conséquent, les performances d'une telle approche, puisqu'elles dépendent du nombre de voisins d'un nœud, diminuent lorsque la densité du réseau augmente. Plusieurs approches ont été proposées afin d'améliorer les performances d'un protocole de diffusion, une description complète de celles-ci est proposée dans [44]. Nous nous appuyons sur ces approches afin de limiter le trafic généré par la diffusion des requêtes d'élection. Plus précisément, nous supposons qu'un protocole de routage sous-jacent est utilisé. Or, les protocoles proactifs et hybrides se basent sur des protocoles de diffusion efficaces désignant un sous-ensemble de nœuds devant faire suivre les messages de contrôle. Pour cela, ils maintiennent les informations relatives à ce sous-ensemble de nœuds. En utilisant ces informations (technique de *cross-layer*), le protocole d'élection assure une diffusion efficace des requêtes d'élection. Concrètement, une interface entre le protocole de routage et de localisation de ressources permet à ce dernier d'accéder aux informations concernant les nœuds désignés pour faire suivre les messages diffusés. Elle permet aussi de fournir des informations portant sur le trafic reçu et envoyé par le nœud. Par exemple, si le protocole de routage correspond à OLSR [5], alors les informations identifient les nœuds MPRs (*Multi-Point Relay*) qui couvrent l'ensemble des nœuds accessibles en deux sauts et déterminent le trafic reçu et envoyé par les nœuds 1-voisins. Il en est de même pour le protocole TBRPF [32].

Si le protocole sous-jacent correspond à des protocoles réactifs, alors le protocole de diffusion est de type aveugle. Dans ce cas, afin de garantir une diffusion efficace, chaque nœud du réseau envoie périodiquement les informations de contrôle nécessaires à une diffusion efficace dans un message *hello*. Ces informations correspondent à :

- l'identifiant du nœud (adresse IP),
- la liste des 1-voisins,
- le trafic reçu et le trafic envoyé par le nœud.

Finalement, lorsque le coordinateur diffuse une requête d'élection sur E sauts, les nœuds désignés pour faire suivre la requête rediffusent cette dernière.

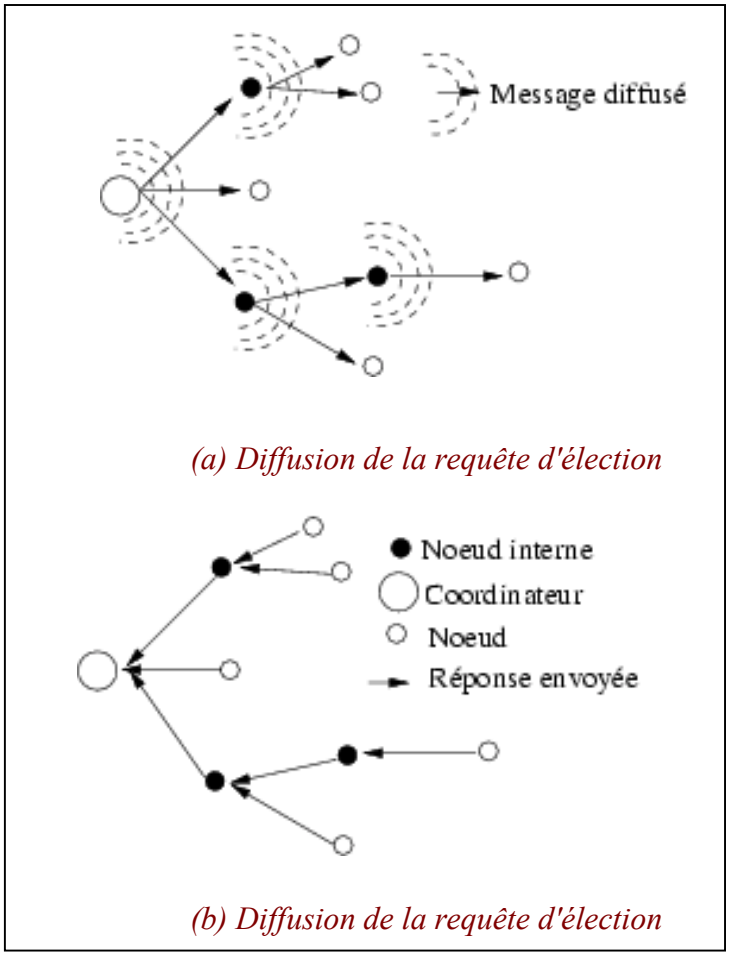


Figure 4-2 Processus d'élection

Afin de résoudre le problème lié à l'implosion du nombre de réponses reçues par le coordinateur de l'élection, nous proposons d'agréger les réponses des nœuds interrogés. L'agrégation des messages se fait au niveau des *nœuds internes* correspondant aux nœuds ayant fait suivre la requête d'élection diffusée par le coordinateur (figure 4-2 (a)). Pour obtenir une agrégation efficace des réponses, les nœuds internes qui réceptionnent les réponses envoyées par leurs 1-voisins doivent renvoyer un seul message agrégeant les réponses vers le nœud interne qui lui a précédemment fait suivre la requête d'élection (figure 4-2 (b)). En d'autre terme, un nœud interne A de niveau i (accessible en i sauts à partir du coordinateur)

doit recevoir les réponses des nœuds de niveau *i*+1 avant de renvoyer un message de réponse agrégé. Pour cela, le nœud interne *A* définit un délai (*time-out*) à partir duquel est renvoyée la réponse agrégée. L'efficacité de l'agrégation des messages dépend de la valeur du délai. Si le délai est trop petit, alors le nœud *A* renvoie plusieurs messages contenant les réponses des nœuds de niveau *i*+1. Si le délai est trop grand, alors le temps d'attente du coordinateur augmente d'autant. Nous fixons donc un délai prenant en compte la bande passante disponible entre *A* et ses nœuds 1-voisins et permettant aux nœuds 1-voisins de *A* de lui répondre avant que ce délai n'expire. La bande passante disponible est déterminée à partir des informations relatives au trafic reçues et envoyées aux 1-voisins par le protocole de routage sous-jacent ou par le protocole de localisation lui-même (voir ci-avant).
Dans un réseau sans fil, le calcul de la bande passante disponible doit prendre en compte le fait que les nœuds partagent le même canal de transmission. En effet, le trafic généré par les nœuds à portée d'émission d'un nœud *A* diminue la bande passante de *A* puisque ce dernier ne peut émettre de messages sans être à l'origine de collisions. Par conséquent, le calcul de la bande passante au niveau d'un nœud doit prendre en compte le trafic généré par les nœuds 1-voisins et les réceptions des nœuds 1-voisins, puisque la génération ou la réception de messages interdit à *A* l'accès au canal pour une émission ou une réception.

Le nœud *A* ne peut donc recevoir un message de réponse à l'élection émis par un nœud 1-voisin *B* dans le cas suivants:

- *A* est en train de recevoir un message. En d'autres termes, il existe au moins un nœud 1-voisin de *A* (noté *Ai*) autre que *B* en train d'émettre, et donc si *B* émet un message, ce dernier entre en collision avec le message actuellement reçu par *A*,
- *B* émet déjà un autre message,
- au moins un nœud, noté *Aj*, 1-voisin de *B* est en train d'émettre un message. Par conséquent, *B* ne peut émettre un message sans que ce dernier n'entre en collision avec le message envoyé par *Aj*,

Par conséquent, l'algorithme calculant le délai fixé par le nœud *A* pour pouvoir recevoir une réponse du nœud *B* prend en compte :

- le temps d'émission des nœuds *Ai*,

- le temps d'émission de l'ensemble des nœuds A_j. Notons que ce temps ne prend pas en compte le temps d'émission des 1-voisins de A et de *B* car il est pris en compte dans la ligne précédente.

Nous récapitulons dans le tableau 4-2 l'ensemble des notations utilisées.

Tableau 4-2 Récapitulatif des notations

A_i	Nœud 1-voisin de A
A_j	Nœud 1-voisin de B
T_1	Temps d'émission de l'ensemble des nœuds 1-voisins de *A*
T_2	Temps d'émission de l'ensemble des nœuds 1-voisins de *B* mais non 1-voisins de *A*

Le temps d'émission de l'ensemble des nœuds A_is vérifie :

$$T_1 = \sum_{Ai} T_{emission}(Ai),$$

avec $T_{emission}(A_i)$ correspondant au temps d'émission du nœud A_i.

Le temps d'émission de l'ensemble des nœuds A_js non 1-voisins de A vérifie :

$$T_2 = \sum_{Aj,\, Aj \neq Ai} T_{emission}\,(A_j).$$

Sachant que le trafic reçu par *B* est celui expédié par ses 1-voisins qui correspondent : (i) aux nœuds 1-voisins de *A* et de *B*, et (ii) aux nœuds 1-voisin de *B* et non de *A*, on obtient :

$$T_2 = \sum_{Aj,\, Aj \neq Ai} T_{emission}\,(A_j) = T_{reçu}(B) - \sum_{Aj = Ai} T_{emission}\,(A_j),$$

avec $T_{reçu}(B)$ désignant le temps pendant lequel *B* a reçu des messages.

Il en résulte que, si nous considérons un intervalle de temps T_{INT}, le temps pendant lequel chacun des 1-voisin de *A* peut envoyer une réponse à *A* vérifie :

$$T_{libre}(B) = T_{int} - T_{1,\,INT} - T_{2,INT},$$

avec $T_{1,\,INT}$ désignant le temps d'émission de l'ensemble des nœuds A_i durant l'intervalle de temps T_{INT}, et $T_{2,\,INT}$ correspondant au temps d'émission de l'ensemble des nœuds A_js qui ne sont pas 1-voisins de *A* durant l'intervalle de temps T_{INT}.

Par ailleurs, étant donnée bd_{max} la bande passante du protocole MAC sous-jacent, la bande passante effective entre A et B pour que B puisse émettre durant la période de temps T_{INT} vérifie :

$$bd_{effect}(B) = \frac{bd_{max} \times T_{libre}(B)}{T_{int}}.$$

On en conclut donc que le délai fixé par A pour recevoir une réponse de B vérifie :

$$delai(A\text{-}B) = \frac{tailleMessage(B)}{bd_{effect}(B)},$$

avec *tailleMessage* correspondant à la taille du message envoyé entre A et B.

Sachant que le nœud A doit recevoir toutes les réponses des nœuds 1-voisins, A doit donc attendre *le plus lent* de ces voisins. Le délai, noté $delai(A)$, qu'il fixe est donc égal à :

$$delai(A) = max_{\forall B} [\ delai(A\text{-}B)\ ]$$

Lorsque le délai $delai(A)$ est écoulé, le nœud interne A envoie au nœud interne de niveau $i\text{-}1$, lui ayant fait suivre la requête d'élection, le message contenant les réponses qu'il a agrégées. Notons que si A reçoit tous les messages de ses voisins avant que le délai ne soit écoulé, il renvoie immédiatement le message au nœud interne supérieur. Finalement, à partir des réponses reçues, le coordinateur sélectionne le nœud le plus à même, à cet instant, de jouer le rôle de répertoire et élit ce dernier comme répertoire.

4.2.2 Gestion dynamique de la configuration du système

La mobilité et les déconnexions volontaires ou non des terminaux induisent un changement de topologie pouvant être à l'origine de la non couverture de certains terminaux. Nous définissons, dans cette section, de quelle façon est effectué le redéploiement des répertoires suite aux changements de topologie. Plusieurs situations peuvent être à l'origine d'un redéploiement :

- un nœud ne souhaite plus jouer le rôle de répertoire (sa batterie a par exemple atteint un niveau critique) et souhaite attribuer cette tâche à un autre répertoire,

- un répertoire se déconnecte, et il est alors nécessaire d'en désigner un nouveau couvrant la surface,
- plusieurs répertoires couvrent la même région, c'est-à-dire que, des nœuds reçoivent les annonces de plusieurs répertoires.

Nous décrivons ci-dessous le processus de redéploiement des répertoires dans chacune de ces situations.

Lorsqu'un terminal ne souhaite plus jouer le rôle de répertoire, il commence un processus d'élection et devient coordinateur. L'algorithme de sélection utilisé diffère de celui présenté dans la section précédente puisque le répertoire ne prend pas en compte les données liées à sa propre couverture durant le processus de sélection.

La déconnexion involontaire d'un répertoire est à l'origine de la non couverture d'une zone du réseau. Un terminal appartenant à cette zone identifie qu'il n'est pas couvert puisqu'il ne reçoit pas de publicité d'un répertoire. Il initie alors une procédure d'élection comme indiquée précédemment. Durant l'intervalle de temps pendant laquelle a lieu l'élection, les nœuds appartenant à la zone de couverture du répertoire déconnecté, peuvent toujours découvrir les ressources se trouvant à l'extérieur de la zone, en interrogeant les répertoires adjacents. Ils connaissent les répertoires adjacents car ces informations sont contenues dans les publicités envoyées par chaque répertoire du réseau comme détaillé dans la section 4.3.1. Toutefois, la localisation des ressources disponibles dans la zone n'est plus assurée. Afin de résoudre ce problème, le répertoire réplique les informations relatives aux ressources disponibles dans son N-voisinage sur les répertoires adjacents.

Afin de réaliser un déploiement homogène des répertoires, chaque répertoire réévalue périodiquement sa zone de couverture. Cette réévaluation permet de limiter le nombre de terminaux couverts par plusieurs répertoires, et donc de diminuer le trafic inutilement généré. Pour cela, un répertoire définit si son rayon d'action doit être augmentée, c'est-à-dire si N doit être incrémenté, décrémenté, ou au contraire rester stable. Périodiquement, chaque répertoire R annonce sa présence en diffusant une annonce sur N+1 sauts. Les nœuds recevant cette annonce sauvegardent l'adresse du répertoire et seuls ceux se trouvant à N-1, N, et N+1 sauts lui répondent (en agrégeant leurs réponses suivant le processus défini dans la section précédente). A partir de ces réponses, le répertoire définit si sa couverture doit être

augmentée, diminuée ou rester stable. Pour cela, il calcule le ratio entre le nombre $|d_M^1(R)|$ de nœuds accessibles en M sauts et couverts seulement pas le répertoire *R* par rapport au nombre $|d_M^m(R)|$ de nœuds déjà couverts par *m* autres répertoires, en faisant varier *M* de *N-1* à *N+1* sauts. La comparaison des ratios définis pour *N-1*, *N* et *N+1* sauts, permet au coordinateur de vérifier si l'augmentation ou la diminution du rayon de couverture du répertoire diminue le nombre de nœuds couverts par plusieurs répertoires. Plus précisément, la couverture du répertoire *R* passe de *N* sauts à *N*+1 sauts si :

$$\frac{|d_{N+1}^1(R)|}{|d_{N+1}^m(R)|} > \frac{|d_N^1(R)|}{|d_N^m(R)|}.$$

Notons que l'augmentation du rayon de couverture défini par *N* est bornée par *H* afin d'éviter qu'un terminal ne gère pas la localisation des ressources sur tout le réseau et ne devienne un point unique de vulnérabilité. La valeur de *H* ayant été définie comme étant la valeur optimale du rayon d'action des répertoires (voir section 4.6).

De la même façon, la couverture *N* du répertoire *R* est décrémentée de 1 si :

$$\frac{|d_{N-1}^1(R)|}{|d_{N-1}^m(R)|} > \frac{|d_N^1(R)|}{|d_N^m(R)|}.$$

Une fois déployée sur le réseau, les répertoires gèrent dynamiquement leur redéploiement tout en réalisant la localisation des ressources dans l'ensemble du réseau (voir section ci-après).

4.3 Localisation dans le N-voisinage

Les répertoires effectuent aussi bien la localisation des ressources dans leur *N*-voisinage que leur localisation dans l'ensemble du réseau *ad hoc* et dans les réseaux inter-connectés *via* un protocole de collaboration. Dans cette section, nous nous intéressons particulièrement à la localisation de ressources dans le *N*-voisinage d'un répertoire. Dans ce contexte, plusieurs tâches incombent aux répertoires. Tout d'abord, un répertoire assure sa visibilité auprès des clients potentiels et des fournisseurs de ressources (§ 4.3.1). Il procède à l'enregistrement des ressources disponibles dans son *N*-voisinage (§ 4.3.2). Il gère un cache dans lequel il stocke les informations relatives aux ressources (§4.3.3). Ainsi, il peut répondre aux requêtes des clients et des répertoires (§4.3.4).

4.3.1 Maintenance de la visibilité

Les répertoires doivent être visibles auprès des clients et des fournisseurs de ressources afin que ces derniers puissent respectivement interroger les répertoires et s'enregistrer auprès des répertoires. Cette visibilité est assurée par le répertoire lui-même qui annonce périodiquement sa présence. Cette annonce contient:

- l'identifiant (adresses IP) du répertoire, afin que les clients puissent identifier et envoyer les requêtes au répertoire,
- la liste des répertoires adjacents. Ces répertoires sont interrogés par les clients si le répertoire est déconnecté (définitivement ou temporairement), ou n'est pas joignable.

Afin de limiter le trafic généré par l'envoi de cette annonce, ces annonces ne sont envoyées que dans le *N*-voisinage du répertoire.

4.3.2 Enregistrement des ressources

Chaque répertoire est responsable de la localisation des ressources mises à disposition dans son *N*-voisinage. Pour cela, il propose aux fournisseurs de ressources, un mécanisme d'enregistrement et de dés-enregistrement de ses ressources. Un fournisseur de ressources recevant les annonces d'un répertoire *A* doit ainsi s'enregistrer auprès de ce dernier. Pour cela, il envoie au répertoire les informations relatives à la ressource qu'il met à disposition. Ces informations sont constituées d'une partie concrète incluant l'identifiant du terminal (permettant de localiser la ressource) et la durée de mise à disposition de la ressource, et d'une description de la ressource permettant de l'identifier. Ainsi, le répertoire peut répondre aux requêtes, envoyées par les clients, qui contiennent une identification de la ressource et leur renvoyer notamment des informations concrètes relatives à la localisation de la ressource. En raison du manque de connectivité d'un réseau *ad hoc* mobile et pour garantir la disponibilité de la ressource, cet enregistrement est réalisé de façon périodique. Le terminal, mettant à disposition la ressource, définit lui-même la fréquence d'enregistrement de la ressource en fonction de *la stabilité du lien* existant entre lui et le répertoire. La stabilité du lien est définie en fonction du nombre d'annonces qu'il a reçu du répertoire par rapport au nombre d'annonces qu'il aurait du recevoir étant donné la périodicité d'envoi des annonces d'un répertoire.

Un répertoire supprime les informations relatives à la ressource lorsque le délai de validité de

l'enregistrement et la durée de mise à disposition de la ressource, sont dépassés. En effet, si le délai de validité de l'enregistrement d'une ressource est atteint, cela suppose que soit le fournisseur n'est plus connecté au réseau, ou qu'il ne se trouve plus dans le *N*-voisinage du répertoire, auquel cas, il se trouve donc à l'extérieur de sa zone de responsabilité du répertoire. Les informations portant sur une ressource peuvent aussi être supprimées dans le répertoire si, avant de se déconnecter volontairement du réseau, le terminal mettant à disposition cette ressource se dés-enregistre auprès du répertoire. Ce terminal envoie alors un message au répertoire contenant l'identification de la ressource à désenregistrer et les informations concrètes relatives à la localisation de la ressource afin que le répertoire puisse dés-enregistrer cette dernière.

4.3.3 Gestion du cache local

Un répertoire dispose de deux zones de stockage : une zone de stockage permanente, dans laquelle il conserve les informations relatives aux ressources se trouvant dans son *N*-voisinage et dont il a la responsabilité d'assurer la localisation, et, une seconde zone de stockage temporaire, correspondant à un cache dans lequel il conserve les informations portant sur des ressources accessibles dans l'ensemble du réseau. Le temps d'attente de l'utilisateur est d'autant plus réduit que l'accès aux informations relatives aux ressources est rapide par le cache plutôt que par la zone de stockage. Il permet aussi de réduire le trafic induit par la localisation des ressources distantes, c'est-à-dire se trouvant à l'extérieur du *N*-voisinage du répertoire.

La gestion des caches est effectuée suivant la politique LFU (*Least Frequently Used*) qui prend en compte la popularité des ressources. La popularité est évaluée en fonction du nombre d'accès effectué par le répertoire afin de répondre aux requêtes envoyées par les clients ou par les répertoires avec lesquels celui-ci collabore.

Lorsqu'un répertoire reçoit un message correspondant à l'enregistrement d'une ressource ou à une réponse renvoyée par un répertoire suite à son interrogation, il vérifie si les informations relatives à la ressource, notées *Ir,* peuvent être stockées dans le cache. Pour cela, étant donnée la politique de gestion de cache, il compare la popularité des éléments correspondant aux informations relatives aux ressources et conservées dans le cache avec celle de *Ir*. Le (ou les) élément(s) du cache les moins populaires et dont la popularité est inférieure à celle de *Ir,* sont

expulsé(s) du cache afin de libérer l'espace nécessaire pour ajouter *Ir*. Si la popularité de *Ir* est inférieure à celle des éléments du cache, alors *Ir* n'est pas ajoutée au cache. Plus précisément, si le message reçu correspond à l'enregistrement d'une ressource, alors *Ir* est ajouté dans la zone de stockage permanente. Au contraire, si le message est une réponse envoyée par un autre répertoire suite à son interrogation, alors les informations portant sur cette ressource sont hors de sa responsabilité et ne sont pas conservés dans la zone de stockage permanente.

Lorsqu'un répertoire reçoit une requête de localisation de ressource de la part d'un client ou d'un répertoire, il vérifie si les informations relatives à cette ressource sont stockées dans son cache, puis si nécessaire dans la zone de stockage permanente.

4.3.4 Interrogation

Un client souhaitant accéder à une ressource envoie une requête au répertoire local se trouvant dans son *N*-voisinage. Cette requête contient une description abstraite de la ressource permettant de l'identifier. La figure 4-3 illustre le processus de localisation en fournissant une modélisation du comportement du client et du répertoire. Quand un répertoire local reçoit une requête de localisation faite par un client, il vérifie s'il conserve des informations relatives à cette ressource. Si c'est le cas, il renvoie au client une réponse, appelée message de *hit*, incluant les informations concrètes et abstraites relatives à la ressource permettant au client de pouvoir l'accéder. Sinon, il a recourt au processus de localisation de la ressource dans l'ensemble du réseau (voir section ci-après).

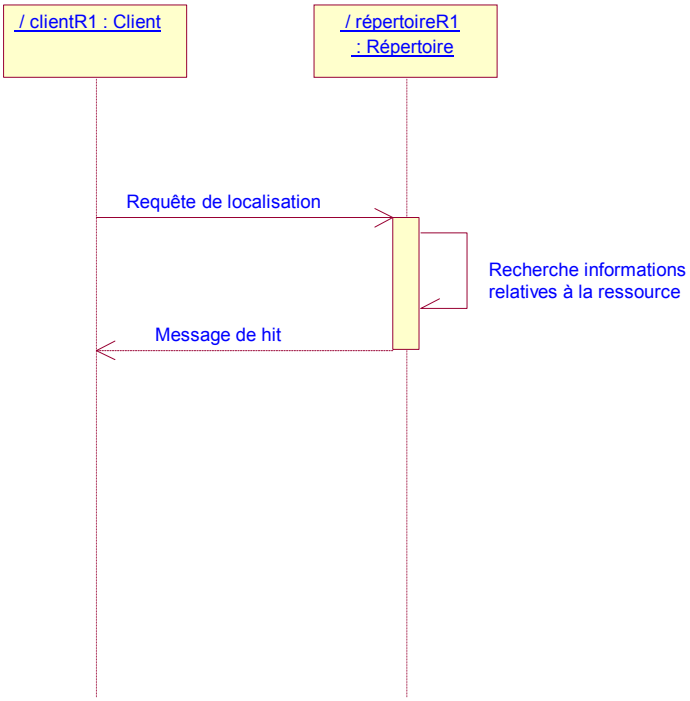


Figure 4-3 Modélisation du comportement d'un répertoire local et d'un utilisateur

4.4 Localisation dans le réseau ad hoc

Les terminaux doivent pouvoir localiser une ressource se trouvant dans l'ensemble du réseau MANET, et cela indépendamment de leur localisation ou de celles des ressources. Pour cela, les répertoires collaborent entre eux afin de localiser les ressources accessibles du réseau. Afin de gérer efficacement cette collaboration, nous proposons d'utiliser les profils des répertoires. Ces profils permettent de cibler les requêtes de localisation vers les répertoires contenant les informations relatives à la ressource demandée. Il en résulte que le délai d'attente est réduit et que le trafic généré par la collaboration des répertoires est restreint puisque les répertoires interrogés sont ciblés. Ces profils des répertoires, contiennent notamment une caractérisation compacte de l'ensemble des informations relatives aux ressources qu'ils stockent (§ 4.4.1). Ces profils sont ensuite utilisés par chaque répertoire afin de sélectionner les répertoires vers lesquels rediriger les requêtes lorsque les informations concernant la ressource recherchée ne sont pas disponibles sur l'hôte (§ 4.4.2).

4.4.1 Profil de répertoire

Le profil d'un répertoire est composé d'une caractérisation du terminal hôte, et d'un résumé compact de l'ensemble des informations relatives aux ressources conservées par l'hôte. Par la suite, nous utilisons le terme *contenu* d'un répertoire, en faisant référence à l'ensemble des informations relatives aux ressources stockées par ce répertoire. Nous utilisons une méthode basée sur le filtre de Bloom [11] (voir ci-après) afin de résumer le contenu des répertoires car ce filtre constitue un moyen efficace de décrire un ensemble d'éléments. En effet, (i) il correspond à un résumé particulièrement compact et nécessite donc une faible capacité de stockage, et (ii) en raison de sa taille restreinte, l'échange de filtre de Bloom entre terminaux nécessite peu de bande passante.

Génération d'un filtre de Bloom

L'idée principale d'un filtre de Bloom consiste à générer un vecteur de m bits (initialisés à 0), appelé filtre de Bloom, à partir d'un ensemble d'éléments stockés par un répertoire. Dans notre cas, un élément désigne la description de la ressource permettant de l'identifier. Notons que cette description n'inclut pas d'informations concrètes comme par exemple l'identifiant du terminal mettant à disposition la ressource.

Dans un premier temps, k fonctions de hachage indépendantes h_1, h_2, ..., h_k, sont choisies telles que :

$$\forall\, i \in \{0,..,k\},\ h_i : R^n \rightarrow \{0,..,m\}$$
$$w \rightarrow h_i(w)$$

Ensuite, le répertoire extrait les informations abstraites relatives à chaque ressource w. Ces informations sont hachées par ces k fonctions de hachage. Finalement, les k bits dont la position est respectivement donnée par le résultat du hachage de w par h_1, .. h_k sont initialisés à 1 dans le vecteur v correspondant au filtre de Bloom. Ce processus est répété jusqu'à ce que toutes les informations abstraites pourtant sur les ressources, soient représentées dans le filtre de Bloom. La figure 4-4 illustre l'ajout de la chaîne de caractère ``bonjour'' dans un filtre de Bloom à partir de trois fonctions de hachages h_1, h_2, h_3.

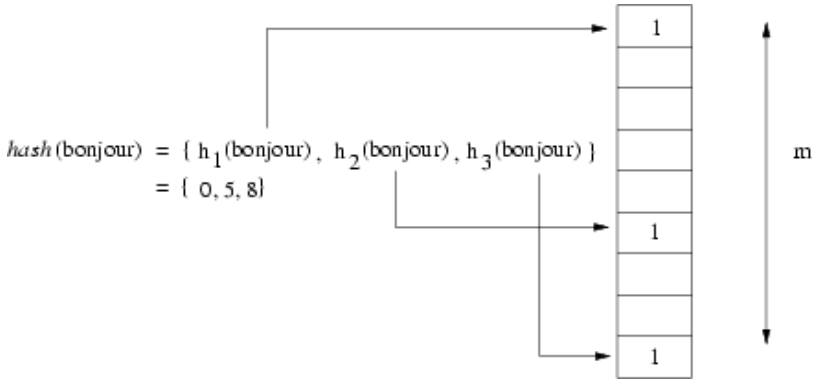


Figure 4-4 Génération d'un filtre de Bloom de taille m à partir de k fonctions de hachage, (m = 9, k = 3)

D'un point de vue pratique, afin de hacher un contenu, nous utilisons l'algorithme *MD5* [70] de 128 bits. La figure 4-5 illustre le calcul de la signature *MD5* de la chaîne de caractère ``bonjour''. Les fonctions de hachage sont créées de la façon suivante. Nous calculons une signature de la chaîne de caractère dont la taille est égale à 128 bits. Puis, nous divisons les 128 bits obtenus en k groupes de taille 128/k bits ($k = 4$ dans l'exemple). Chaque groupe i, $i \in \{1,..,k\}$, conserve un chiffre c_i appartenant à l'intervalle $[0,2^{128/k}]$. Finalement, nous calculons le modulo m de c_i ($m = 600$ dans l'exemple), et, le résultat obtenu correspond à h_i(bonjour), avec $i \in 0,..,k$ et h_i(contenu) $\in$ [0,..,m]. Les bits se trouvant aux positions h_1(bonjour), ..., h_k(bonjour) sont mis à un.

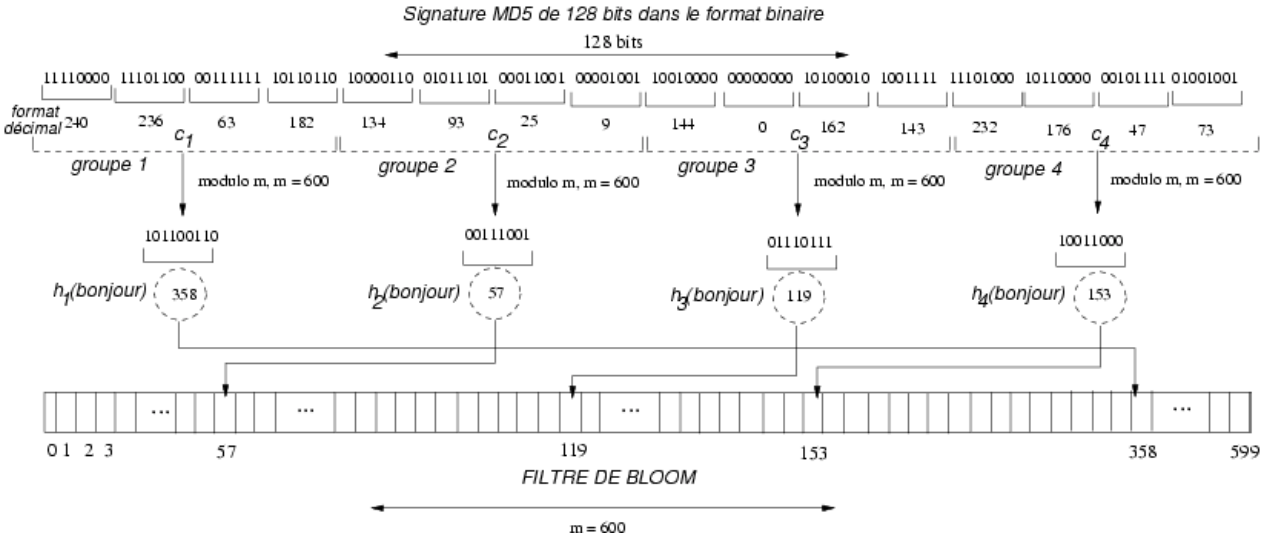


Figure 4-5 Génération d'un filtre de Bloom de taille m à partir de k fonctions de hachage utilisant MD5 128 bits, (m = 600, k = 4)

Par la suite, le filtre de Bloom est mis à jour lorsque le contenu d'un répertoire est modifié. Cette modification provient de l'enregistrement d'une nouvelle ressource (c'est-à-dire du stockage d'informations relatives à cette ressource), ou de son dés-enregistrement (c'est-à-dire de la suppression des informations conservées portant sur cette ressource). Afin que toute modification du contenu d'un répertoire ne nécessite pas la regénération du filtre de Bloom, nous proposons d'utiliser un vecteur de compteurs, noté *c*. Un compteur *c*(*i*) se trouvant à la position *i* est égal au nombre d'enregistrements de ressources ayant conduit à positionner à 1 le bit se trouvant à la position *i* dans le filtre de Bloom, noté *v*. Par exemple, si le répertoire a enregistré deux ressources à l'origine du positionnement à 1 d'un bit à la position *i* dans son filtre de Bloom, alors *c*(*i*) =2. Ainsi, quand une ressource est dés-enregistrée par le répertoire, ce dernier hache les informations relatives à la ressource. Les valeurs en résultant font référence aux positions du vecteur de compteurs ayant été précédemment incrémentées. Les compteurs, aux positions indiquées, sont alors décrémentés. Si un des compteurs décrémenté de *c* atteint la valeur 0, les bits correspondants de *v* sont positionnés à 0. Par conséquent, la modification d'un filtre de Bloom induit un faible coût de traitement. Ce filtre de Bloom local, maintenu par le répertoire, est ensuite intégré au profil de ce dernier et est échangé avec les autres répertoires. Ainsi, un répertoire est capable de localiser une ressource du réseau, comme détaillé ci-après.

4.4.2 Localisation semi-distribuée

D'après la politique de déploiement des répertoires (§ 4.2.1), un terminal *A* souhaitant accéder à une ressource envoie une requête au répertoire *B* se trouvant dans son *N*-voisinage. Cette requête contient une description permettant au répertoire de d'identifier la ressource. Le processus de localisation est illustré par la figure 4-6.

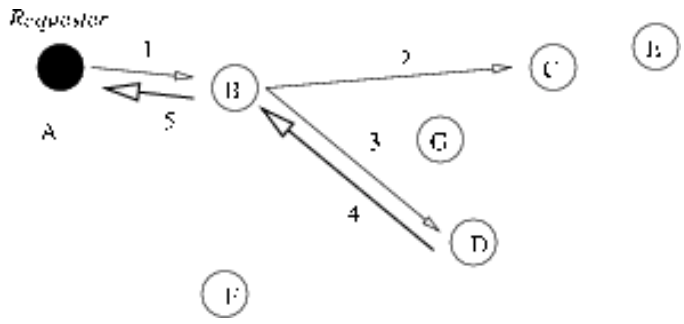


Figure 4-6 Processus de Localisation

Les figures 4-7 et 4-8 décrivent quant à elles le comportement d'un répertoire lorsque le répertoire conserve ou non les informations relatives à la ressource recherchée.

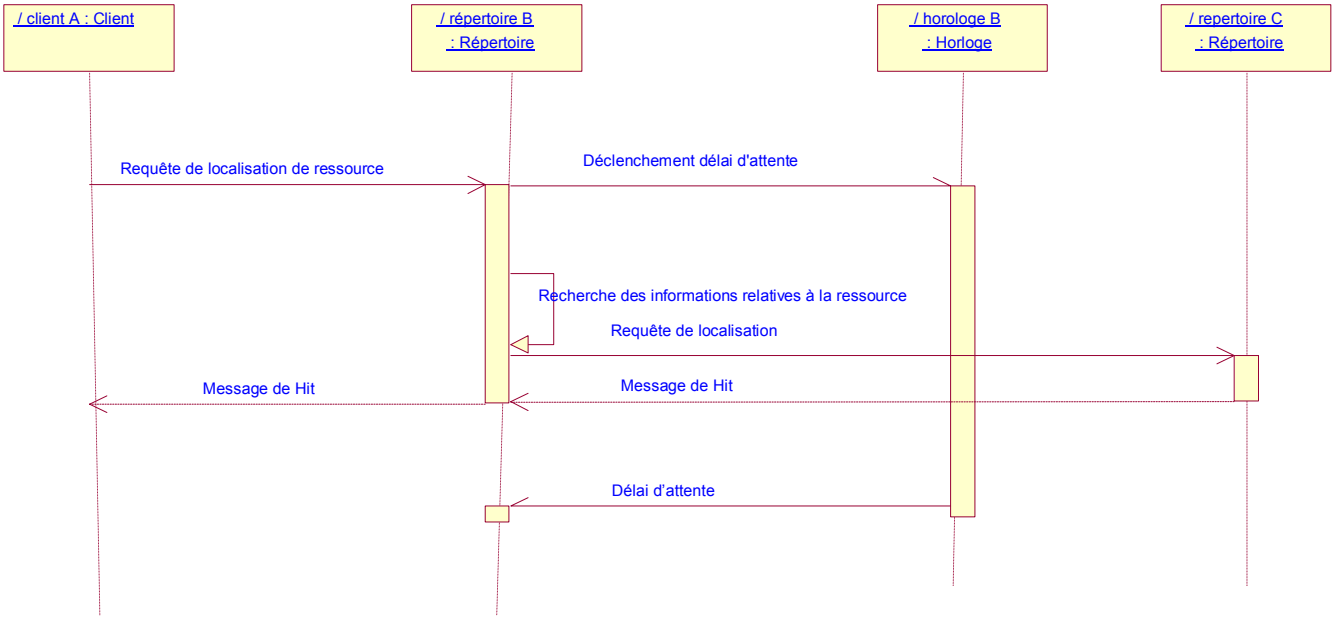


Figure 4-7 Comportement d'un répertoire local et d'un utilisateur lorsque le répertoire interrogé conserve les informations relatives à la ressource

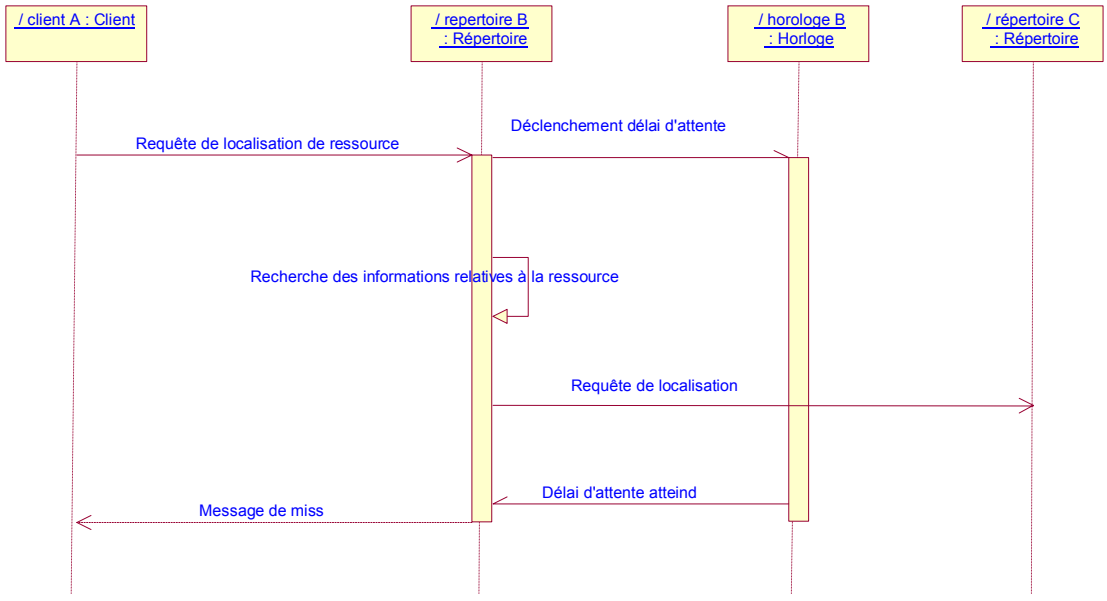


Figure 4-8 Comportement d'un répertoire local et d'un utilisateur lorsque le répertoire interrogé ne conserve pas les informations relatives à la ressource

Quand B reçoit une requête envoyée par A (figure 4-6-- message 1), il vérifie s'il stocke les informations relatives à la ressource demandée dans sa zone de stockage ou dans son cache et renvoie un message *hit* à A dans l'affirmative. Sinon, B hache les informations relatives à la ressource, fournies dans la requête. Plus précisément, B ne hache que les informations permettant d'identifier la ressource sans prendre en compte les informations concrètes telle que l'adresse IP du terminal mettant à disposition cette dernière ou la durée de mise à disposition de la ressource. Les k résultats, notés b_1, .., b_k, générés suite au hachage sont utilisés afin de vérifier si pour chaque filtre de Bloom des répertoires distants, les k bits se trouvant aux positions b_1, .., b_k sont positionnés à un. Seuls ceux vérifiant cette égalité sont supposés conserver les informations relatives à la ressource demandée. Le répertoire envoie alors une requête aux répertoires qui conservent probablement les informations souhaitées (messages 2 et 3), afin de localiser toutes les ressources disponibles dans le réseau.

Finalement, les répertoires interrogés (e.g., *C* et *D* sur la figure 4-6) envoient à *B* un message de *hit* (message 4) incluant les informations, abstraites et concrètes relatives à la ressource demandée, si ces informations sont conservées (figure 4-7). Le répertoire recevant ces réponses vérifie s'il stockera (ou non) dans son cache les informations relatives à cette ressource. Notons qu'aucun message de *miss* n'est envoyé à B afin de limiter le trafic généré. Pour cela *B* définit un délai d'attente pour recevoir les réponses des répertoires interrogés. Finalement, B envoie à A soit un message de *hit* (figure 4-7) ou de *miss* (figure 4-8), ce dernier étant envoyé si *B* ne reçoit aucun message de *hit*.

4.4.3 Gestion de la collaboration

La collaboration entre les répertoires lors de la localisation d'une ressource nécessite l'échange de leurs profils. Cet échange survient en deux occasions :

- l'apparition d'un nouveau répertoire,
- la modification du contenu d'un répertoire.

Nous décrivons, ci-dessous, le processus d'échange des profils dans chacune de ces situations.

Élection d'un nouveau répertoire.

Un répertoire *B,* nouvellement élu, doit localiser les autres répertoires et obtenir leurs profils respectifs, et cela, afin de pouvoir faire suivre de façon efficace les requêtes des clients vers ces derniers. Nous distinguons deux types de situations : soit le répertoire *B* ne connaît aucun répertoire (c'est le cas si le réseau *ad hoc* vient de se créer), soit au contraire il connaît un répertoire. Dans le premier cas, le répertoire *B* diffuse dans le réseau une annonce. Un répertoire recevant cette annonce à un temps *t* diffère l'envoi de son profil à *B* de ε (ε étant déterminé aléatoirement dans un intervalle de temps borné), et cela afin que *B* ne reçoive pas simultanément toutes les réponses. Dans le second cas, si le répertoire *B* connaît un autre répertoire *C*, alors *B* envoie une requête à *C* pour obtenir la liste des profils qu'il conserve. Dans les deux cas, le répertoire *B* crée une liste des répertoires afin de conserver leurs identifiants et leurs profils respectifs. Puis, *B* renvoie son profil local aux répertoires, dès lors que l'enregistrement des ressources se trouvant dans son *N*-voisinage est accompli.

Modification du contenu d'un répertoire.

Plusieurs causes sont à l'origine de la modification du contenu d'un répertoire. Elles correspondent, soit à l'enregistrement, soit au dés-enregistrement d'une ressource. Dans les deux cas, ce changement doit être répercuté aux autres répertoires. Plusieurs techniques peuvent être utilisées conjointement à cet effet, elles se décomposent en deux catégories. La première vise à limiter la taille des messages, la seconde à différer l'envoi de ces notifications. Afin de réduire le nombre de messages envoyés suite à la mise à jour des profils, et lorsque cela est possible, le répertoire ajoute son profil mis à jour aux messages de réponse qu'il renvoie au répertoire, suite à son interrogation. De plus, afin de réduire la taille du message, au lieu d'envoyer le filtre de Bloom en entier, il inclut seulement la liste des bits ayant été modifiés. Par ailleurs, les mises à jour des profils sont différées. En effet, les notifications sont envoyées périodiquement, c'est-à-dire lorsque les changements survenant dans le contenu du répertoire sont significatifs, et lorsque le pourcentage de *false positive* désignant la dégradation du ratio de *hit*, atteint un niveau critique.

4.5 Localisation dans le réseau hybride

Plusieurs types de réseaux (réseau *ad hoc* Wifi ou Bluetooth, réseaux sans fils ou filaires à base d'infrastructure) peuvent coexister indépendamment sans être inter-connectés. Une solution classique, pour connecter ces réseaux, est d'utiliser des passerelles. Ces dernières correspondent à des terminaux munis de plusieurs interfaces de communication, chacune fournissant une connectivité vers un réseau de type différent. Ces réseaux inter-connectés forment ainsi un réseau hybride. Bien que les terminaux puissent communiquer avec les terminaux se trouvant dans d'autres réseaux appartenant au réseau hybride, la localisation de ressources se limite à celles accessibles dans le réseau. Or, fournir une localisation de ressources plus efficace nécessite de fournir des mécanismes additionnels afin de localiser les ressources dans le réseau hybride.

Cette approche nécessite qu'un terminal offrant un service de passerelle joue le rôle de répertoire. Il conserve les profils, incluant le résumé des contenus des répertoires se trouvant sur les réseaux qu'il inter-connecte. Pour chaque réseau *R* qu'il inter-connecte, il diffuse à la fois son profil et celui des répertoires des réseaux adjacents (c'est à dire, des réseaux, à l'exclusion de *R*, qu'il inter-connecte). Ainsi, les ressources se trouvant dans l'ensemble du réseau hybride sont localisables. Ce répertoire passerelle inclut dans les profils de ces répertoires, une caractérisation du réseau dans lequel les répertoires se trouvent et son adresse afin que les terminaux puissent, une fois la ressource localisée, accéder à ce dernier. Notons que pour accéder au réseau externe sur lequel se trouve la ressource, un terminal peut utiliser d'autres passerelles.

Par ailleurs, lorsqu'un terminal répond à une élection il intègre au profil matériel, qu'il envoie au coordinateur de l'élection, le fait qu'il joue le rôle de passerelle et la description du (ou des) réseau(x) qu'il inter-connecte. Plus précisément, si plusieurs terminaux passerelles, se trouvant dans la zone d'élection, et fournissent un accès au même réseau, alors un seul d'entre eux est sélectionné pour jouer le rôle de répertoire. Si dans un réseau, plusieurs répertoires passerelle assurent la localisation de ressources dans les mêmes réseaux inter-connectés, seul celui ayant la plus faible adresse IP répercute aux autres répertoires d'un des réseaux qu'il inter-connecte, les notifications de changement survenus au niveau des contenus des répertoires.

Cette approche nécessite que le protocole de localisation que nous proposons soit déployé sur les réseaux inter-connectés, ce qui inclut des réseaux à base d'infrastructure pouvant être filaire ou sans fil. Dans ce type de réseaux, le déploiement est réalisé de façon statique par l'administrateur du domaine et non pas dynamiquement par le protocole de localisation. Un

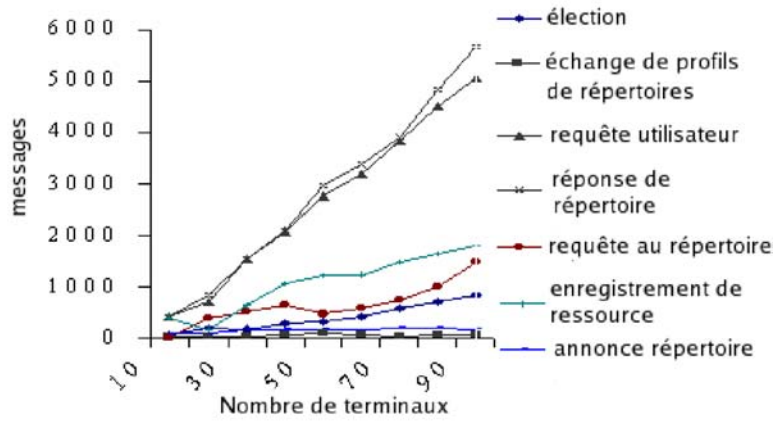
protocole de configuration du réseau tel que DHCP [109] annonce aux terminaux la localisation des répertoires. Le protocole de localisation se comporte donc différemment s'il se trouve dans un réseau administré puisque l'élection n'est pas initiée. Toutefois, il est important de noter que le protocole que nous proposons n'a pas été conçu dans le but d'opérer dans un réseau à grande échelle tel que l'Internet qui nécessite des mécanismes de localisation particuliers et adaptés au contexte. Une localisation de ressources dans de tels réseaux à grande échelle doit être effectuée par un autre protocole de localisation adapté. Nous n'abordons pas ici le problème de l'inter-opérabilité entre les protocoles de localisation. Notons que la solution proposée par Raverdy et al. [69] permet de résoudre ce problème d'interopérabilité.

4.6 Évaluation

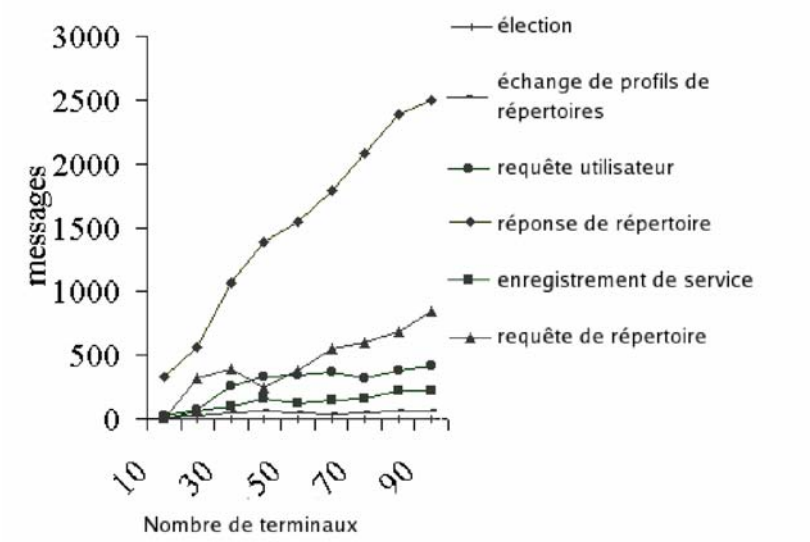
Afin d'évaluer les performances du protocole que nous avons proposé, nous avons réalisé une simulation avec le simulateur NS-2[16]. Nous décrivons ci-dessous les paramètres que nous avons utilisés lors de cette simulation.

Les terminaux sont déployés aléatoirement et se déplacent selon des trajectoires aléatoires. Plus précisément, une destination est choisie de façon aléatoire et les terminaux se dirigent vers cette dernière à une vitesse constante. Nous utilisons lors de cette simulation le protocole de routage OLSR et le protocole IEEE 802.11 en tant que protocole sous-jacent. Nous bénéficions notamment des mécanismes implémentés au niveau du protocole OLSR pour l'élection des relais multi-points. Cela permet de réduire le trafic généré et les calculs dupliqués. En effet, l'élection de nœuds relais multi-points induit un trafic supplémentaire puisque les nœuds envoient périodiquement un message pour définir qu'un nœud a été élu par un autre nœud, pour devenir relais multi-points. Afin de fournir une mesure exhaustive du trafic généré par notre protocole de localisation, nous prenons en compte à la fois (i) le trafic généré par l'expéditeur et (ii) le trafic généré suite à la retransmission des messages. De plus, les données sont collectées depuis le début de la simulation afin de fournir le coût de déploiement des répertoires lors de l'initialisation.

[16] http://www.isi.edu/nsnam/ns/



(a) *Trafic généré par l'expéditeur initial*

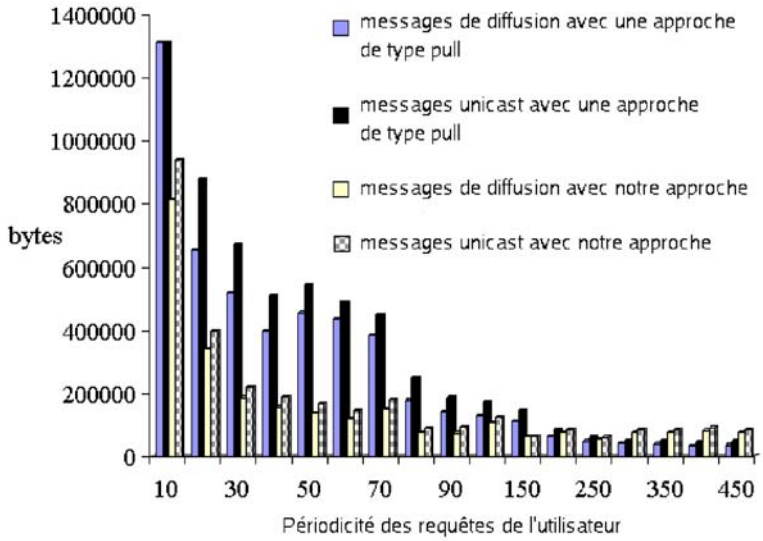


(b) Trafic induit par l'envoi des messages par les nœuds intermédiaires]

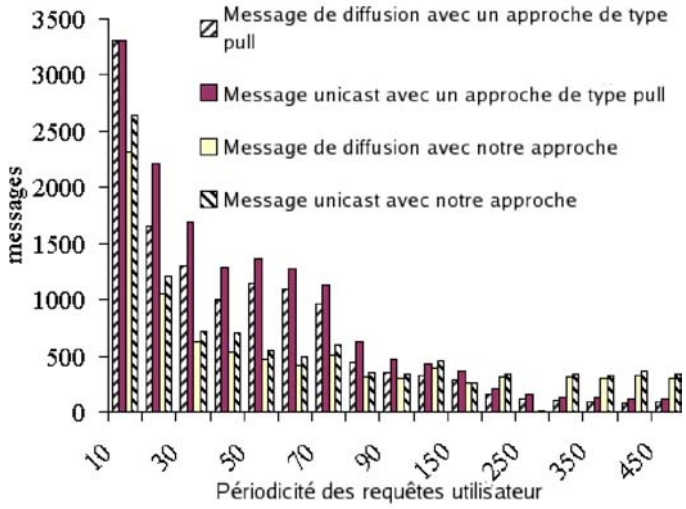
Figure 4-9 Trafic généré en fonction de la densité des nœuds

Nous considérons dans un premier temps un réseau *ad hoc* de *NT* terminaux (*NT* variant selon les simulations) déployés sur une surface de S = 600 m ×600m. De plus, 10% des terminaux mettent à disposition chacun E = 2 ressources. Chaque client initie périodiquement un processus de localisation pour une ressource parmi les $E \times NT \times 10\%$ ressources disponibles dans le réseau. Le réseau *ad hoc* est structuré autour des répertoires ayant été élus dans une zone dont la portée est définie par H = 1 saut. La figure 4-9 fournit le trafic généré par notre protocole de découverte de ressources en fonction de la densité des terminaux, tout en considérant chaque type de message de façon séparée. Il est normal de constater que, le nombre de requêtes de localisation expédiées par les utilisateurs et de réponses renvoyées par les répertoires, augmente lorsque la densité des terminaux croit. Il est intéressant de noter que comparativement la densité des terminaux a un faible impact sur le déploiement des répertoires (messages d'élection, annonces de répertoires et d'enregistrement des ressources).

Enfin, le trafic généré par l'échange de profils entre les répertoires est quasiment indépendant de la densité des terminaux. La différence entre le trafic généré par les requêtes de localisation envoyées par les clients et les répertoires, provient du fait que les réponses des répertoires incluent à la fois les réponses aux requêtes des utilisateurs et aux répertoires. Le coût attribué à la découverte des ressources dans l'ensemble du réseau provient des requêtes des répertoires (i.e., les requêtes envoyées aux autres répertoires) et de l'échange des filtres de Bloom. Le coût de la découverte locale est dû aux les requêtes envoyées par les clients aux répertoires et au réponses reçues en retour.



(a) Trafic généré par l'expéditeur initial



(b) Trafic du au ré-acheminement des messages]

Figure 4-10 Comparaison entre une approche décentralisée et une approche de type pull

La figure 4-10 compare notre protocole de localisation de ressources avec un protocole décentralisé de type *Pull*. Plus précisément, avec une approche de type *pull*, un terminal

souhaitant accéder à une ressource diffuse une requête de localisation dans l'ensemble du réseau et reçoit en échange une réponse (envoyée *via* un message point-à-point) de la part du terminal mettant à disposition la ressource. Les paramètres de simulation sont les mêmes que ceux utilisés précédemment. Notre protocole de découverte de localisation de ressources induit un coût fixe associé au déploiement et à la maintenance de la collaboration entre les répertoires. En contrepartie, il se comporte mieux qu'un protocole décentralisé de type *pull* lorsque le taux de requêtes des utilisateurs atteint un certain seuil.

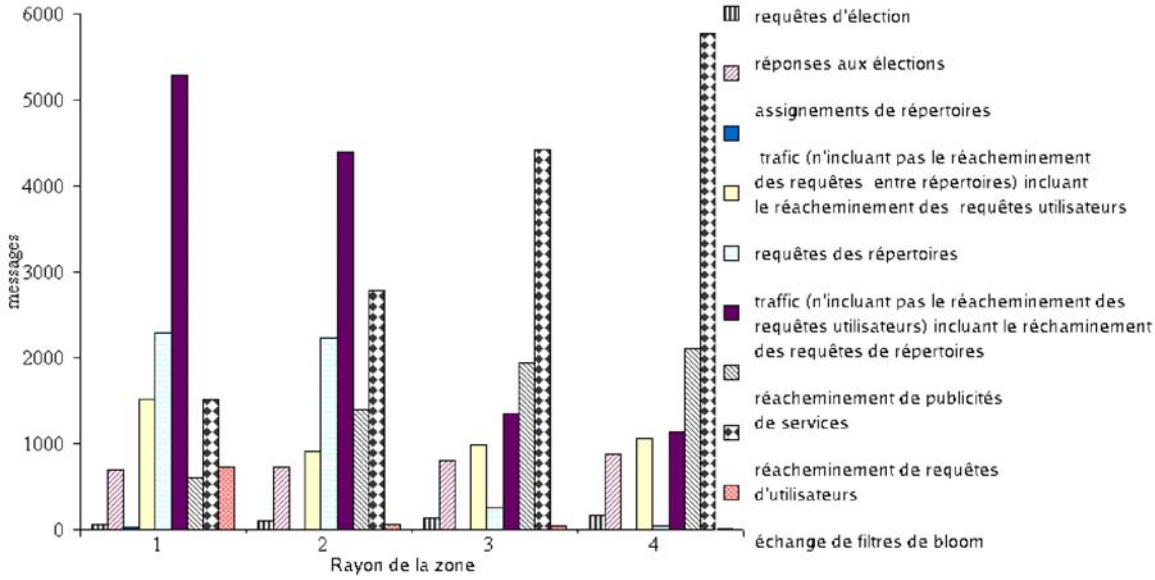


Figure 4-11 Trafic généré en fonction de la couverture des répertoires

Nous considérons maintenant un réseau composé de NT=90 nœuds sur une surface de S = 1000m × 1000m. La figure 4-11 évalue le trafic généré par notre protocole de découverte de ressources en fonction du nombre de sauts H. Le coût associé à l'élection d'un répertoire inclut le trafic généré par les messages de diffusion envoyés par le coordinateur de l'élection et les réponses renvoyées par les terminaux H-voisins. Dans cette simulation, nous supposons que tous les terminaux acceptent de jouer le rôle de répertoire. Le trafic est relativement stable puisqu'un compromis s'effectue entre le nombre de coordinateurs d'élections (qui est inversement proportionnel à H) et le nombre de nœuds qui répondent au coordinateur de l'élection (qui est proportionnel à la densité des nœuds et à H). Le trafic induit par l'échange de résumés est directement proportionnel au nombre de répertoires déployés dans le réseau. La figure 4-11 donne aussi le coût associé au ré-acheminement des requêtes de localisation de ressources et aux enregistrements des ressources (correspondant à des messages *unicast*), qui est proportionnel à N puisque le nombre de sauts influe directement sur la distance moyenne entre les clients et les répertoires. En prenant en compte le coût du déploiement des

répertoires (messages d'élection, échange des filtres de Bloom et assignation de la tâche de répertoire), il apparaît que la valeur optimale dans ce contexte est *H* égal à 2.

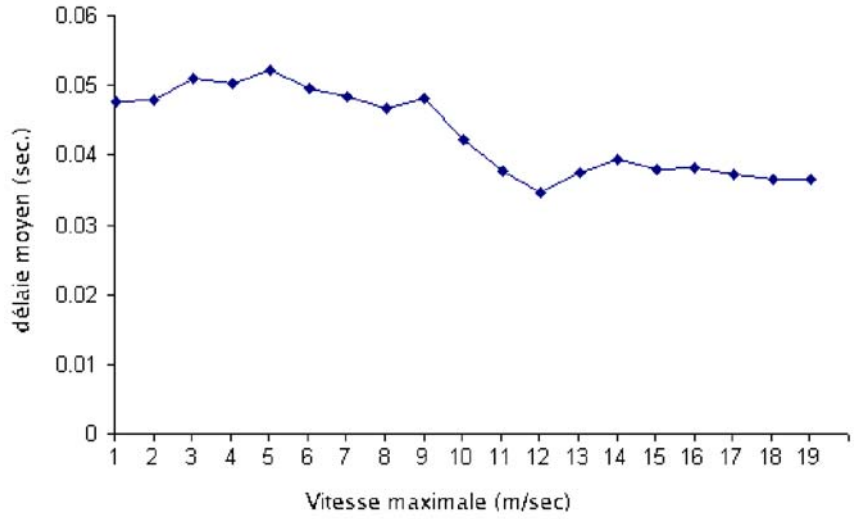


Figure 4-12 Comparaison des délais d'attente moyens

La figure 4-12 présente une évaluation du délai moyen d'attente, par terminal, en fonction de la vélocité des terminaux. Nous considérons une surface de réseau de 1000m × 1000m composée de $NT = 50$ terminaux et $N = 2$. Cette évaluation démontre que le délai d'attente d'une localisation de ressources est stable par rapport à la vélocité des nœuds. Ce résultat est dû au fait qu'une vélocité accrue des répertoires et des terminaux, mettant à disposition les ressources, conduit à une dissémination accrue des informations relatives aux ressources. Ainsi, quand un client envoie une requête à un répertoire, la probabilité que ce dernier conserve les informations relatives à la ressource demandée est plus forte.

Nous évaluons maintenant la probabilité que les informations relatives à une ressource ne soient pas stockées par un répertoire, alors que l'étude du filtre de Bloom de ce dernier affirme le contraire. Ceci est appelé un *false positive*. La probabilité d'un *false positive* est donnée par :

$$(1 - (1 - \frac{1}{m})^{kn})^k \approx (1 - e^{k\,n/m})^k,$$

avec n définissant le nombre de ressources ayant été enregistrées auprès du répertoire, m la taille du filtre de Bloom et k le nombre de fonctions de hachage. La valeur de k et m devrait être choisie de telle façon que la probabilité d'un *false positive* soit minimum, c'est-à-dire que $k \approx m\ ln\ 2 / n$. Toutefois, en pratique, la valeur de k est inférieure à sa valeur optimale, afin de

réduire la surcharge liée au hachage des informations relatives aux ressources. Comme exemple, considérons un répertoire ayant enregistré 100 ressources. Supposons que la taille moyenne des informations relatives à chaque ressource soit de 3kbits. Selon ces hypothèses, la figure 4-13 présente la probabilité qu'un *false positive* survienne en fonction de la taille du filtre de Bloom pour k=1,...,8. Cette figure illustre le fait que si k=3, alors fixer la taille du filtre de Bloom à 700 bits est suffisant pour obtenir une probabilité de *false positive* proche de zéro.

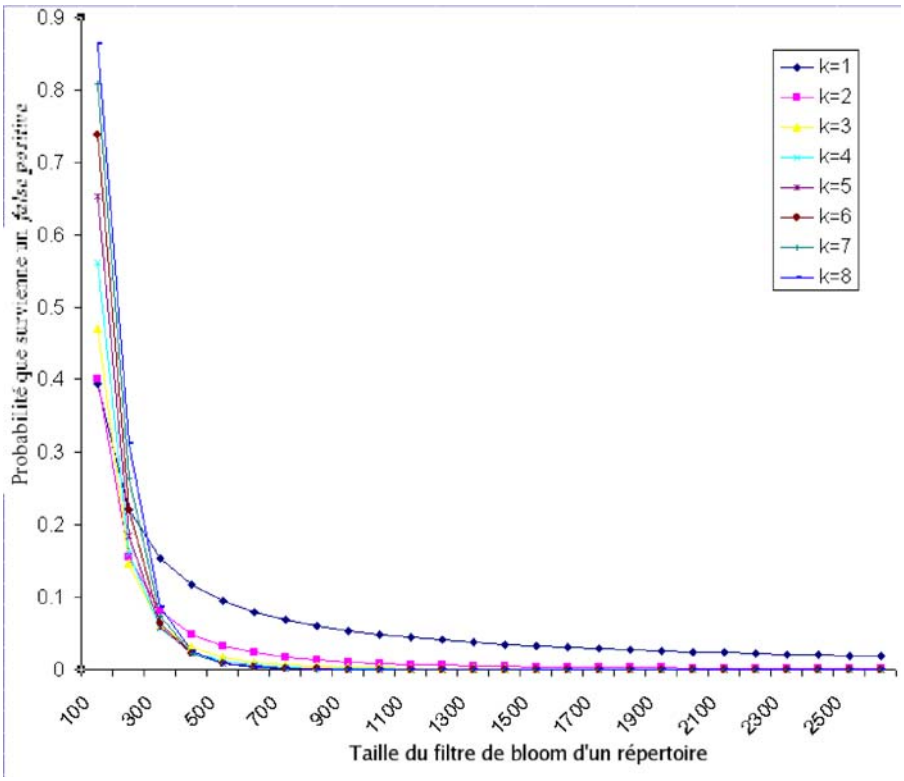


Figure 4-13 Probabilité d'un *false positive*

Nous déduisons de cette série d'évaluation que lorsque la surface du réseau *ad hoc* est importante et que la densité des nœuds augmente, le trafic généré par notre approche est maîtrisé. Cela s'explique par le fait que lorsqu'un répertoire noté *A* ne stocke pas les informations relatives à une ressource recherchée par un client, il ne diffuse pas la requête de localisation mais l'adresse à un nombre restreint de répertoires sélectionnés. Cette sélection des répertoires est effectuée par *A* à partir des résumés (ou filtres de Bloom) du contenu des autres répertoires, que *A* conserve. La taille des résumés pouvant être fixée de façon à ce que la probabilité d'interroger un répertoire ne stockant pas la ressource recherchée soit quasiment nulle. Enfin, nous avons comparé notre protocole à un protocole de type *pull*, il en résulte que

notre protocole génère moins de trafic lorsque la fréquence de recherche de ressources augmente[17].

4.7 Conclusion

Nous avons présenté dans ce chapitre un protocole de localisation de ressources semi-distribué. Le principal objectif ayant influencé la conception de ce protocole, est de garantir une localisation des ressources dans un réseau *ad hoc* à large échelle. Dans tel réseau, une localisation effectuée de façon centralisée ou entièrement distribuée n'est pas envisageable. En effet, une localisation centralisée conduit notamment à la formation d'un goulet d'étranglement et à l'épuisement des ressources énergétiques des terminaux se trouvant dans le voisinage de l'entité centralisatrice. Lorsque le nombre de terminaux et la surface du réseau sont importants, une approche entièrement décentralisée se traduit par l'augmentation drastique du trafic induit par l'envoi de requêtes de localisation et par les réponses associées. Nous proposons à cet effet un protocole de localisation de ressources semi-distribué. Notre approche se base sur un ensemble de répertoires distribués dynamiquement et de façon homogène dans le réseau *ad hoc* et gérant la localisation des ressources pour les utilisateurs. Cela permet de limiter le nombre de terminaux coopérant ensemble afin de localiser les ressources. Indirectement, cela réduit le trafic généré en limitant la quantité de requêtes envoyées et de réponses renvoyées. De plus, cette localisation de ressources ne se base pas sur un seul répertoire pouvant être déconnecté et représentant un point de cassure. Cette localisation de ressources est effectuée suite à la coopération des répertoires entre eux. Afin de fournir une coopération efficace, les répertoires échangent entre eux un profil contenant les caractéristiques physiques de l'hôte et de leur environnement ainsi qu'un résumé compact de leur contenu. Enfin, ils utilisent ces profils pour sélectionner et rediriger la requête provenant du terminal client vers les répertoires conservant les informations portant sur la ressource demandée.

Cette approche permet de réduire le trafic résultant de la localisation. En effet,

[17] Notons que dans un réseau à large échelle, du fait de la forte densité des terminaux et de la surface importante du réseau, la fréquence de recherche de ressources est importante.

- la portée des requêtes de localisation est limitée car les clients interrogent le répertoire le plus proche,
- une requête de localisation d'un client n'est pas diffusée. Elle est au contraire adressée à un seul destinataire, à savoir le répertoire voisin,
- les répertoires échangent entre eux des profils compacts,
- la collaboration (et l'échange de profils compacts) entre les répertoires permet de localiser de façon efficace, sans recourir à des techniques de diffusion, les répertoires conservant les informations relatives aux ressources demandées et de rediriger les requêtes vers ces derniers,
- l'utilisation de caches au niveau des répertoires et des clients réduit le nombre de requêtes envoyées.

Par ailleurs, le déploiement de ces répertoires se base sur une élection dynamique des terminaux appelés à jouer le rôle de répertoires. Il répond aux exigences d'un environnement mobile *ad hoc* car :

- il assure une répartition équitable de la charge de répertoire en réaffectant périodiquement cette dernière,
- le processus d'élection d'un répertoire prend en compte les ressources (et notamment les ressources énergétiques) disponibles des terminaux,
- il se base sur une technique de diffusion efficace des messages d'élection, et sur l'agrégation des réponses réduisant ainsi le nombre de paquets échangés,
- la portée d'une élection est limitée,
- il s'adapte dynamiquement aux changements survenant sur le réseau.

De plus, afin de résoudre les problèmes liés au manque de connectivité, deux techniques sont utilisées. La première se base sur l'utilisation de protocoles de routage fournissant la possibilité à des terminaux distants de localiser et d'accéder des ressources *via* un ou plusieurs terminaux intermédiaires faisant suivre les messages. La seconde vise à rattacher un réseau *ad hoc* à d'autres réseaux (filaires ou non, basés sur des infrastructures ou non) à l'aide d'une passerelle pouvant correspondre, par exemple, à un terminal ayant plusieurs interfaces de communications (ou cartes réseaux). Ainsi, nous proposons une solution qui gère aussi bien la localisation de ressources au sein du réseau *ad hoc* que celle dans le réseau hybride. De plus, elle est facilement déployable à la fois dans un réseau *ad hoc* et dans un réseau basé sur des infrastructures puisque cette solution se base sur un ensemble de répertoires dont le

déploiement est soit réalisé par le protocole lui-même, dans le cas d'un réseau *ad hoc*, soit par l'administrateur dans le cas d'un réseau basé sur une infrastructure. Il en résulte que cette approche permet une localisation des ressources adaptée dans un environnement hybride constitué de réseaux *ad hoc* ayant une forte densité et une faible connectivité.

Finalement, nous avons validé cette solution en réalisant une simulation du protocole nous permettant d'évaluer ses performances dans un réseau à large échelle. Nous introduisons en outre une mise en œuvre du protocole de localisation pour des ressources correspondant à des services Web dans le chapitre suivant. Cela permet l'exploitation de notre protocole par la réalisation de l'informatique diffuse.

Chapitre 5 Mise en œuvre d'une localisation semi distribuée à la découverte de services

5.1 Introduction

Dans le chapitre précédent, nous avons introduit un protocole de localisation de ressources pour réseaux MANET à large échelle. Nous avons notamment indiqué l'intérêt de ces réseaux pour les applications d'intervention en cas de sinistre, ou encore du domaine militaire, du fait du rapide déploiement du réseau et de l'espace devant être couvert. Toutefois, l'exploitation des réseaux MANET à large échelle ne doit pas être seulement considérée pour ces domaines d'applications spécifiques. Ces réseaux offrent un particulier des propriétés intéressantes pour le développement d'applications relevant du domaine de l'*informatique diffuse*, également qualifiée d'*intelligence ambiante* dans le cas où la population d'usagers visée est le grand public et non pas seulement le monde professionnel (comme cela est considéré dans le domaine du « *pervasive computing* »).

Brièvement, la notion d'*intelligence ambiante* a émergé à la fin des années 90 dans le but de mettre les nouvelles technologies de l'information et de la communication au service du grand public, par opposition à la production de technologies d'un usage de plus en plus complexe, comme cela a été la tendance dans les années 80-90. La vision d'intelligence ambiante vise ainsi à exploiter les nouvelles technologies de l'information et de la communication pour améliorer la qualité de vie des usagers. Concrètement, cette vision se traduit par le couplage d'interfaces intelligentes et de systèmes informatiques ubiquitaires. Les *interfaces intelligentes* exploitent les dernières technologies en matière d'interaction multimodale (reconnaissance de la parole, synthèse vocale, perception de l'environnement, …), permettant une communication aussi naturelle que possible de l'usager avec l'environnement numérique, par exemple au moyen de la parole. Les *systèmes informatiques ubiquitaires* s'appuient sur

les dernières avancées technologiques en matière de réseaux et dispositifs sans fil numériques. Les systèmes informatiques ubiquitaires deviennent ainsi accessibles à l'usager dans la plupart des situations, puisque celui-ci peut toujours avoir un dispositif numérique sans fil en sa possession, comme par exemple un téléphone cellulaire, et bénéficier des équipements de l'environnement où il se trouve, *via* leur mise en réseau. Les systèmes informatiques réalisant la vision d'intelligence ambiante offrent potentiellement des services numériques de nature variée à l'utilisateur, puisqu'ils peuvent être exploités pour des tâches d'ordre professionnel, domestique ou encore ludique.

Le développement de systèmes supportant cette vision d'intelligence ambiante a fait l'objet de nombreuses études et d'investissements significatifs, tant dans le monde académique qu'industriel. Nous voyons même émerger des offres commerciales de tels systèmes, comme illustré par le slogan publicitaire « *sense and simplicity* » de PHILIPS. Nous voyons en outre émerger des systèmes de communication virtuelle qui contribuent directement à la réalisation de cette vision, même s'ils ne sont pas affichés comme tel. Nous pouvons notamment citer les systèmes de communication au-dessus de/*via* l'Internet ou encore les systèmes de visiophonie. Toutefois, de nombreux défis scientifiques et technologiques restent posés par la réalisation effective de la notion d'intelligence ambiante, comme illustré par les différents projets en cours dans ce domaine (voir par exemple les actions de recherche et développement dans le domaine de l'intelligence ambiante, supportées par le consortium AIR&D[18]).

Considérant plus spécifiquement le développement de systèmes informatiques ubiquitaires, il s'agit de permettre à l'utilisateur d'accéder aux contenus et services, publics ou privés, en tout lieu, à tout instant. Une telle exigence requiert en autres une connectivité réseau et une plate-forme logicielle ubiquitaires. Les réseaux *ad hoc* nous semblent être un élément essentiel dans le déploiement d'une connectivité réseau ubiquitaire : ils pallient les absences d'infrastructure réseau en certains lieux, permettent aux usagers de communiquer à un coût financier moindre et favorisent les communications spontanées ou encore l'établissement de communautés virtuelles. Les réseaux *ad hoc* à large échelle équipés de notre protocole de localisation de ressources peuvent notamment être avantageusement exploités pour la réalisation de ville communicante, à l'instar des maisons communicantes qui sont les premières incarnations des systèmes de l'intelligence ambiante[19]. Toutefois, pour devenir de vrais systèmes ubiquitaires,

[18] http://www.air-d.org
[19] Par exemple, voir http://www.hitech-projects.com/euprojects/amigo/.

de tels réseaux doivent être complétés avec une plate-forme logicielle qui permet que les services et contenus puissent être accédés mais également déployés sur les différents terminaux du réseau *ad hoc*, et en particulier les terminaux sans fil de capacité relativement limitée. L'exigence clé pour cette plate-forme est la garantie de l'interopérabilité entre les terminaux, qui peut être obtenue de deux manières : (i) la définition d'un standard de plate-forme logicielle dont l'acceptation est telle que l'on puisse supposer son déploiement à large échelle et ainsi la disponibilité de services dans la plupart des situations [98], ou (ii) la définition d'un intergiciel réalisant l'interopérabilité entre des services logiciels hétérogènes [102]. La seconde approche soulève différents défis qui dépassent le cadre de notre travail de thèse, et font notamment l'objet de différents travaux [110]. En revanche, l'architecture de services Web apparaît comme une plate-forme logicielle de choix pour devenir un standard de l'informatique diffuse, notamment du fait du caractère diffus du Web. Toutefois, cette architecture doit encore évoluer pour répondre aux différentes exigences des systèmes informatiques ubiquitaires, considérant notamment le déploiement de services Web dans les réseaux *ad hoc* à large échelle. Dans le cadre de notre thèse, nous nous sommes spécifiquement intéressés à ce dernier problème, ce qui nous a conduit à proposer une mise en œuvre de notre protocole de localisation de ressources introduit dans le chapitre précédent, dans l'architecture de services Web. Notre travail, décrit dans ce chapitre, regroupe ainsi le développement d'une plate-forme logicielle pour le déploiement de services Web sur des terminaux sans fil, mobiles, de relativement faible capacité, et un protocole de localisation de services Web mobiles pour réseau *ad hoc* à large échelle.

Ce chapitre se décompose comme suit. Tout d'abord, nous introduisons brièvement l'architecture de services Web et en particulier les technologies sur lesquelles s'appuient les services Web dans la section 5.2. Puis, dans la section 5.3, nous présentons notre intergiciel, appelé *Ariadne*, qui permet le déploiement, la localisation et l'accès de services Web mobiles dans un réseau *ad hoc* à large échelle. Nous présentons ensuite, dans la section 5.4, le prototype d'Ariadne, que nous avons développé et qui est disponible sous licence LGPL[20]. Cette présentation est complétée dans la section 5.5 par une évaluation d'Ariadne par expérimentations. Les résultats obtenus montrent que notre intergiciel offre des performances comparables aux intergiciels traditionnels en terme de temps de réponse et que les temps de réponse de la localisation de services sont comparables à ceux des accès aux services, validant

[20] http://www-rocq.inria.fr/arles/download/ariadne/index.html

notre approche tant du point de vue de l'exploitation des services Web que de celle des réseaux *ad hoc* pour l'informatique diffuse. Enfin, nous concluons ce chapitre par une synthèse de notre contribution dans la section 5.6.

5.2 Architecture de services Web

Initialement, les services Web ont été introduits pour le déploiement d'applications de commerce électronique sur l'Internet. A l'heure actuelle, ils sont également utilisés dans d'autres contextes d'application comme, par exemple, pour partager des ressources dans des environnements à large échelle tels que des GRID [30], ou encore accéder à des applications Web à partir de téléphones sans fils dans des réseaux télécoms [49]. La popularité des services Web et leur utilisation dans ces contextes variés s'expliquent par le fait que les services Web peuvent opérer dans des environnements distribués particulièrement hétérogènes aussi bien du point de vue des plates-formes logicielles déployées que des modèles de programmation utilisés. Cela les rend aussi adaptés à une utilisation dans les réseaux *ad hoc* caractérisés par l'hétérogénéité des plates-formes logicielles des terminaux et leur structure décentralisée.

Succinctement, un service Web correspond à une application connectée à un réseau, et capable d'interagir avec d'autres applications en utilisant des protocoles d'échange standardisés. Un service Web est caractérisé par une interface publique décrite suivant un langage de description standard. Spécifiquement, les technologies des services Web fournissent un niveau significatif d'interopérabilité pour les communications entre services, de par l'utilisation du format de données standard XML et la définition d'un certain nombre de langages et protocoles standards, tous basés sur XML. L'architecture de services Web promeut notamment une représentation des applications Web unifiée, et indépendante des plates-formes. Pour cela, elle définit le langage WSDL (*Web Service Description Language*) [85] basé sur XML pour décrire les interface publiques des applications Web, que nous définissons brièvement dans la section suivante. Un autre élément clé de l'architecture des services Web, présenté dans la section 5.2.2. est le protocole SOAP (*Simple Object Access Protocol*) [84] pour l'échange de données XML lors des interactions avec les services. Enfin,

indiquons que les services Web peuvent être déployés sur la plupart des plates-formes logicielles, comme par exemple J2ME[21] ou .NET[22].

5.2.1 Langage de description de services Web

Le langage WSDL (*Web Service Description Language*) [85] est utilisé pour décrire les interfaces fonctionnelles des services Web. Le langage WSDL est défini de façon à séparer la description abstraite de l'interface d'un service de sa description concrète, et cela afin de pouvoir réutiliser facilement la description abstraite d'un service Web. Plus précisément, la partie abstraite d'une spécification WSDL définit une description fonctionnelle (et abstraite) du service Web contenant notamment les opérations offertes par le service. La description concrète fournit les informations relatives au format des messages et au déploiement du service Web. La figure 5-1 détaille de façon plus précise la structure d'une spécification WSDL et le tableau 5-1 identifie le rôle de chaque élément la constituant.

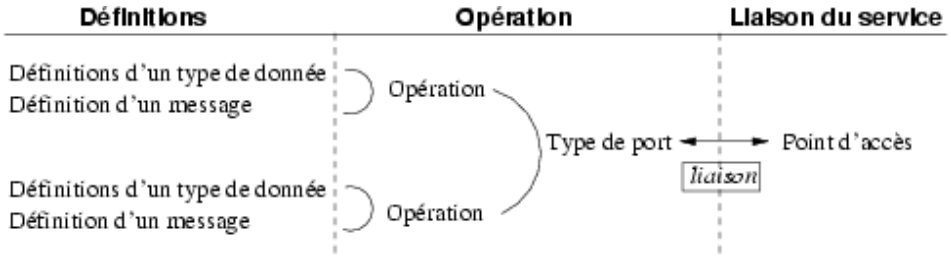


Figure 5-1 Structure d'un document WSDL

La description abstraite d'un service est ainsi composée d'un ensemble d'opérations, regroupées pour former une interface définie par l'élément caractérisé par un *Type de Port* (ou *portType*). Chaque opération est caractérisée par ses paramètres d'entrée et sortie, définis par l'élément message. La partie concrète de la spécification WSDL définit pour sa part le format d'échange des paramètres des opérations et le point d'accès à partir duquel l'interface d'un service peut être appelée.

[21] http://java.sun.com/j2me/.

[22] http://www.microsoft.com/net.

Types	*<types>*	*container* pour les définitions de types de données
Message	*<message>*	définition abstraite des données échangées
Opération	*<operation>*	définition abstraite des fonctionnalités
Type de Port	*<portType>*	ensemble abstrait d'opérations supportées par un ou plusieurs points d'accès
Liaison	*<binding>*	définition concrète du format des messages pour un type de port particulier
Point d'accès	*<port>*	URI d'un point d'accès lié à un type de port

Tableau 5-1 Entités d'un document WSDL

Le point d'accès d'un service est défini par son URI (*Universal Resource Identifier*) qui désigne l'adresse du point d'accès, le chemin d'accès au service et son nom. Les parties concrètes et abstraites sont liées ensemble par une liaison définie par l'élément *binding*.

5.2.2 Protocole SOAP

Le protocole SOAP (*Simple Object Access Protocol*) [84] est un protocole qui gère l'échange de messages avec les services Web, et en particulier les appels à des procédures distantes (RPCs pour *Remote Procedure Calls*) induits par l'invocation des opérations des services Web. Un message SOAP correspond à un document XML. Par convention, un message SOAP est constitué d'une en-tête et d'un corps du message (voir Annexe 1). L'en-tête du message est optionnelle, et contient les informations nécessaires pour traiter le corps du message. Le corps du message contient les informations à destination du service Web. Le protocole SOAP est indépendant du protocole de transport, au sens où n'importe quel protocole de transport peut être utilisé. Toutefois, le protocole SOAP utilise typiquement le protocole HTTP.

Les standards SOAP et WSDL sont les composants sur lesquels se base l'intergiciel Ariadne afin de gérer la localisation dynamique et l'accès à des instances de services Web dans un réseau *ad hoc* mais aussi dans un réseau hybride et/ou sur l'Internet.

5.3 Ariadne : un intergiciel pour réseau ad hoc large échelle

L'intergiciel Ariadne gère d'une part la localisation des instances des services Web se trouvant dans un réseau *ad hoc* ou hybride, et d'autre part l'accès aux instances des services Web, c'est-à-dire l'invocation des opérations fournies par les instances des services Web localisées. Plus précisément, l'intergiciel Ariadne s'appuie sur le logiciel WSAMI-CSOAP[23] pour effectuer l'accès au aux services Web déployés sur des terminaux mobiles. Le logiciel WSAMI-CSOAP implémente un *container* SOAP qui héberge les services Web pour gérer les appels à leurs opérations suivant le modèle RPC (RPC pour *Remote Procedure Call*) synchrone ou asynchrone (*oneway*) [98]. Nous présentons le prototype WSAMI-CSOAP de container SOAP développé afin de pouvoir être déployé sur des PDAs, dans la section 5.4.1.

Dans cette section, nous nous intéressons plus spécifiquement au service de localisation dynamique des instances des services Web dans le réseau *ad hoc*, qui constitue le cœur de notre contribution et qui s'appuie sur le protocole de localisation de ressources que nous avons présenté dans le chapitre 4. Nous décrivons précisément de quelle façon le service de localisation des services Web gère les informations relatives aux instances des services (§5.3.1), et comment le protocole de localisation permet de garantir la découverte dynamique des instances de services Web du réseau *ad hoc* (§5.3.2).

5.3.1 Service de localisation de services Web

Le service de localisation d'Ariadne est déployé sur chacun des terminaux du réseau *ad hoc* et effectue la localisation des instances des services Web pour les applications clientes s'exécutant sur le terminal. Pour cela, une application cliente fournit au service de localisation la partie abstraite de la spécification WSDL du service Web qu'elle souhaite localiser.

L'application cliente dispose d'un certain degré de liberté pour exprimer ses exigences concernant le service Web auquel elle souhaite accéder. En effet, elle peut inclure dans sa requête, soit la description fonctionnelle et abstraite en entier, soit un sous-ensemble de cette description incluant le nom du service et/ou seulement la liste des opérations fournies. En retour, le service de localisation fournit à l'application les descriptions des instances de services Web correspondantes se trouvant dans le réseau *ad hoc* et dans le réseau hybride.

[23] http://www-rocq.inria.fr/arles/download/ozone/index.html

Cette description des instances des services Web donne pour chaque instance : sa description abstraite, la description de son point d'accès (i.e., adresse IP, chemin d'accès et nom du service). Ces informations permettent ensuite à l'application cliente de pouvoir invoquer les opérations fournies par l'instance du service Web qu'elle aura finalement sélectionné.

Notons que la description des instances des services Web du réseau *ad hoc* est fournie par le service de localisation. Si l'application dispose d'une connexion vers l'Internet et souhaite accéder à une instance de service Web s'y trouvant, nous supposons qu'elle dispose des informations nécessaires pour y accéder. Lors de la conception de l'application cliente, le développeur définit ainsi quelles sont les instances de services Web disponibles sur l'Internet en se basant sur les informations contenues dans un répertoire central et global, de type UDDI. Ces informations sont donc enregistrées par l'application cliente et utilisées par celle-ci lorsqu'elle souhaite accéder à une instance de service Web se trouvant sur l'Internet. L'application cliente fournit en particulier ces informations à l'intergiciel qui gère l'accès au service Web. En revanche, lorsqu'un client souhaite accéder à un service Web du réseau *ad hoc*, son service de localisation vérifie dans un premier temps si l'instance du service Web concerné est disponible localement. Si ce n'est pas le cas, le service de localisation utilise le protocole de localisation de services Web, comme présenté dans la section 5.3.2 suivante. Puis, le service de localisation fait suivre à l'application cliente les informations relatives aux instances de service Web retrouvées par le protocole.

Enfin, dans le but de pouvoir assurer la visibilité des services Web disponibles sur le terminal, le service de localisation gère leur enregistrement et leur *désenregistrement*. Lors de son enregistrement, un service Web fournit au service de localisation, ses descriptions abstraites et concrètes.

5.3.2 Localisation des instances des services Web

Nous présentons dans cette section le protocole de localisation implémenté par le service de localisation déployé sur chaque terminal. Ce protocole s'appuie sur des notions que nous avons introduite dans le chapitre 4. Nous rappelons ici rapidement le fonctionnement de ce dernier, le lecteur pouvant se référer au chapitre 4 pour une description complète.

Le protocole de localisation de services Web se base sur un ensemble de répertoires déployés dans le réseau *ad hoc*. Un répertoire correspond à un service de localisation déployé sur un terminal, auquel a été affectée une tâche supplémentaire. Cette tâche consiste à gérer la localisation des instances des services Web se trouvant dans l'ensemble du réseau *ad hoc* pour le compte des autres terminaux. Notons que ces répertoires sont déployés dynamiquement dans le réseau *ad hoc*. Toutefois, notre protocole tout aussi bien être intégré dans un réseau à base d'infrastructure, auquel cas les répertoires seront déployés par l'administrateur du réseau. Les répertoires déployés gèrent la localisation des services Web en collaborant avec les autres répertoires du réseau, qu'ils interrogent lorsqu'ils ne stockent pas d'informations relatives à la localisation de services requis.

Précisément, un répertoire accomplit trois fonctions principales. Tout d'abord, il enregistre les informations relatives aux instances des services Web mises à disposition par les terminaux. De plus, il gère la collaboration avec les autres répertoires. Pour cela, il échange avec les autres répertoires son profil qui inclut un résumé de l'ensemble des services qu'il a enregistré. En réalisant ces deux tâches, le répertoire est capable de localiser les instances des services Web se trouvant dans l'ensemble du réseau. Enfin, le répertoire répond aux requêtes de localisation envoyées par les services de localisation des terminaux souhaitant accéder à des services Web. Nous décrivons ci-dessous de quelle façon le répertoire réalise chacune de ces tâches.

Enregistrement des services Web

Le service de localisation d'un fournisseur de service Web enregistre les services que ce dernier met à disposition auprès du répertoire le plus proche. Pour cela, le fournisseur de services envoie au répertoire un message d'enregistrement qui contient :

- la description concrète du service déployé dans le réseau, qui inclut les informations relatives au(x) point(s) d'accès du service,
- la description abstraite fonctionnelle du service.

Le répertoire enregistre ces informations afin de pouvoir répondre aux requêtes envoyées par les services de localisation des répertoires et des clients. Puis, il ajoute ses informations dans son résumé.

Collaboration entre répertoires

Lorsqu'un répertoire interroge un autre répertoire, la requête qu'il lui envoie contient les mêmes informations que celles envoyées par un client. Afin de limiter le nombre de requêtes de localisation de services Web envoyées entre les répertoires, ces derniers échangent un résumé de leur contenu, ce contenu correspondant à l'ensemble des informations relatives aux services Web qu'ils ont enregistré. La génération d'un résumé du contenu d'un répertoire se base sur le *hachage* des spécifications abstraites des interfaces fonctionnelles des services Web enregistrés. Notons qu'une requête de localisation peut contenir la spécification abstraite complète de l'interface fonctionnelle du service ou bien un sous-ensemble cette dernière. Hacher la spécification abstraite complète de l'interface fonctionnelle du service Web n'est donc pas suffisant. Une proposition alternative consiste à hacher tous les sous-ensembles des interfaces. Dans ce cas, le calcul en résultant est coûteux pour le terminal. Par conséquent, nous limitons le nombre de sous-ensembles de la spécification abstraite des interfaces fonctionnelles d'un service Web. En nous appuyant sur le standard WSDL, nous définissons un sous-arbre de profondeur égale à deux où :

- La racine de l'arbre correspond au service et sa valeur correspond au nom du service.
- La racine de l'arbre a plusieurs nœuds enfants correspondant aux opérations fournies par le service, la valeur de chaque nœud désignant le nom de l'opération.
- Chacun nœud enfant possède plusieurs nœuds fils correspondant aux paramètres d'entrées et sorties de l'opération. La valeur de chacun de ces fils correspond respectivement au nom du paramètre d'entrée ou de sortie de l'opération.

La racine de l'arbre et tous les chemins de longueur égale à 1 (i.e., limité à une opération) et 2 (incluant les paramètres d'entrée et de sortie de l'opération) de l'arbre ci-dessus, définit alors tous les sous-ensembles de la description abstraite du service. La valeur de chaque sous-ensemble est obtenue en concaténant la valeur du nœud d'un chemin spécifique, et est hachée par les k fonctions de hachage afin d'être incluse dans le résumé correspondant à un filtre de Bloom (voir section 4.4.1).

A partir des résumés des contenus des répertoires distants qu'il conserve, un répertoire est capable de rediriger les requêtes de localisation vers les répertoires conservant les informations portant sur le service Web demandé. Afin de déterminer si un répertoire conserve la description d'un service Web w à l'aide du filtre de bloom de ce dernier, un

répertoire interrogé examine chaque sous-ensemble de la description du service, comme défini précédemment. Pour chaque sous-ensemble noté $sub_i(w)$ extrait, tous les bits se trouvant à la position $h_1(sub_i(w))$, $h_2(sub_i(w))$, ..., $h_k(sub_i(w))$, avec k correspondant aux nombre de fonctions de hachage utilisées, sont étudiés. S'ils sont tous positionnés à 1, le répertoire est supposé contenir les informations relatives au service Web *w*. Le répertoire interrogé fait suivre la requête de localisation aux répertoires supposés savoir répondre. Lorsque le répertoire obtient effectivement une réponse en échange, il renvoie cette dernière au service de localisation du terminal ayant initié la localisation.

Localisation des services Web

Lorsqu'une application cliente souhaite accéder à un service Web, elle interagit avec le service de localisation disponible sur son terminal. Si le service de localisation conserve les informations relatives au service demandé, alors il renvoie la réponse à l'application cliente. Sinon, il utilise le protocole de localisation de service Web. Il contacte alors un répertoire en lui envoyant une requête de localisation.

Lorsqu'il reçoit une requête de localisation, un répertoire vérifie s'il dispose des informations relatives aux services Web demandé. Pour cela, il extrait de la requête, la description abstraite (pouvant être partielle) du service Web. Puis, il hache cette description. A partir du résultat obtenu, il vérifie si les bits se trouvant dans son filtre de Bloom sont positionnés à un. Si c'est le cas, cela signifie qu'il conserve les informations relatives à l'instance du service Web recherché. Il recherche alors parmi l'ensemble des informations relatives aux instances des services Web stockées, celles portant sur le service Web demandé. Si le répertoire ne conserve pas d'informations relatives au service demandé il coopère avec les autres répertoires du réseau suivant le protocole décrit précédemment. En échange, le répertoire retourne, soit un message de *hit* incorporant les informations relatives au(x) service(s) Web recherché, soit un message de *Miss*. Plus précisément, un message de type *hit* contient la description concrète du service, ce qui inclut celle des points d'accès du service Web. Le message comprend également la description abstraite et fonctionnelle du service. Lorsque le service de localisation du client reçoit un message de *hit*, il retourne les informations contenues dans le message, à l'application.

5.4 Prototype d'Ariadne

Nous présentons dans cette section le prototype de l'intergiciel Ariadne. Cet intergiciel permet de déployer des services Web sur des terminaux ayant des capacités limitées. De plus, il gère l'accès dynamique aux services Web se trouvant aussi bien dans le réseau *ad hoc* que sur l'Internet. Dans le premier cas, Ariadne supporte une localisation dynamique des instances des services Web dans le réseau *ad hoc* et les réseaux interconnectés à ce dernier. Concrètement, l'intergiciel Ariadne est constitué d'un service de déploiement de services Web, *container* SOAP gérant l'accès aux instances des services Web (voir §5.4.1), et d'un service de localisation dynamique des instances des services Web (voir §5.4.2).

La figure 5-2 présente les principaux composants du prototype d'Ariadne. Les composants coloriés en gris dégradé (respectivement en gris uni) correspondent aux implémentations utilisées par des terminaux caractérisés par leurs faibles ressources tels que des PDAs (respectivement des terminaux plus riches en ressource tels que des ordinateurs portables). Les composants en pointillés (respectivement à gros traits) présentent les implémentations disponibles que nous avons utilisées (respectivement développées).

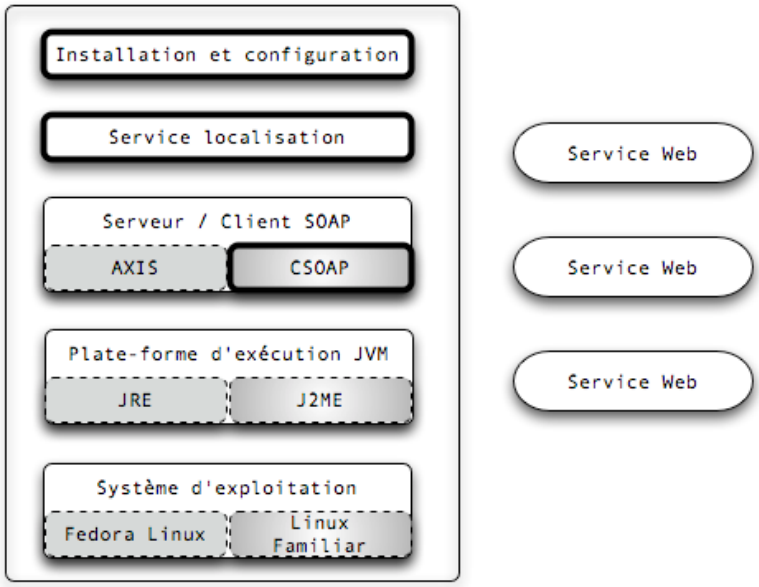


Figure 5-2 : Architecture du prototype Ariadne

Le système d'exploitation que nous avons utilisé est Linux. Pour les terminaux ayant des ressources limitées, la distribution Linux Familiar[24] incluant le noyau et l'environnement

[24] http://www.handhelds.org.

Linux, est utilisée. Nous avons développé l'intergiciel Ariadne dans le langage Java. Ce choix est motivé par l'hétérogénéité des terminaux sans fil, au niveau de leur plate-forme logicielle et matérielle. En effet, un code Java exécuté sur la machine virtuelle Java (ou JVM pour *Java Virtual Machine*) assure la portabilité du code et un certain niveau d'indépendance entre l'application et la plate-forme matérielle et logicielle sous-jacente. Il en résulte que nous utilisons la plate-forme d'exécution java JRE (Java Runtime Environnement) fournissant la JVM et les composants nécessaire à l'exécution des applications développées dans le langage Java. Les terminaux ayant des ressources limitées utilisent pour leur part la plate-forme Java 2 Micro Edition (J2ME[25]). Celle-ci fournit un environnement d'exécution flexible pour des terminaux tels que des téléphones sans fil, des PDAs et plus généralement des systèmes embarqués. Plus précisément, nous utilisons la configuration *Connected Device Configuration* (CDC) incluant une JVM et un ensemble minimal de librairies standards pouvant être utilisées pour les terminaux ayant 32-bits de CPU et au minimum 2Mb de mémoire disponible/allouée à la plate-forme Java et aux applications associées.

L'intergiciel Ariadne intègre notre service de localisation d'instances de services Web. A partir des informations fournies par ce service, une application cliente peut accéder au service Web ainsi localisé. Pour cela, les applications clientes intègre un client SOAP. Les fournisseurs de services intègrent quant à eux un serveur SOAP. Plusieurs implémentations de *containers* SOAP sont disponibles pour différents types de plates-formes. Pour les terminaux ayant des ressources suffisantes, nous utilisons l'implémentation Java *open source* AXIS[26] (*Apache eXtensible Interaction System*). Pour les terminaux ayant des ressources limitées, nous utilisons le *container* CSOAP[27] [98], que nous présentons dans la section suivante.

5.4.1 Container CSOAP

Le *container* CSOAP, fournisseur de services Web, gère le déploiement des services Web et le traitement des messages induits par les appels aux opérations offertes par ces services (voir figure 5.3). L'implémentation du *container* CSOAP se base sur le parseur XML Xerces[28] afin

[25] http://java.sun.com/j2me.
[26] http://ws.apache.org/axis/.

[27] http://www-rocq.inria.fr/arles/download/ozone/index.html
[28] http://xml.apache.org/xerces2-j/.

d'interpréter les messages SOAP, et sur Sun JAX-RPC[29] (*Java API for XML-based RPC*) pour gérer le déploiement et les appels aux opérations des services Web.

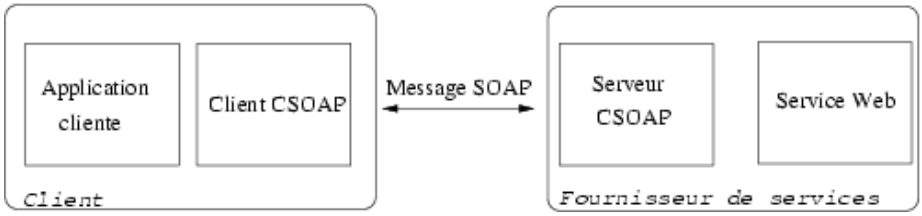


Figure 5-3 Structure d'un container SOAP

Le serveur CSOAP gère le déploiement des services Web et l'invocation des opérations fournies par les services Web déployés sur le terminal hôte. Il écoute de façon continue sur un port TCP donné afin de traiter les messages SOAP provenant des clients, pour les faire suivre au service concerné. Une application cliente interagit avec le client CSOAP au travers d'une interface que ce dernier met à disposition. L'application utilise cette interface afin de pouvoir invoquer les fonctionnalités d'un service Web.

Comme le montre la figure 5-4, un *container* CSOAP est constitué de plusieurs composants, à savoir un fournisseur, un pivot, un gestionnaire global, de déploiement, de l'encodage, et de transport. Nous détaillons ci-dessous le rôle de chacun de ces composants.

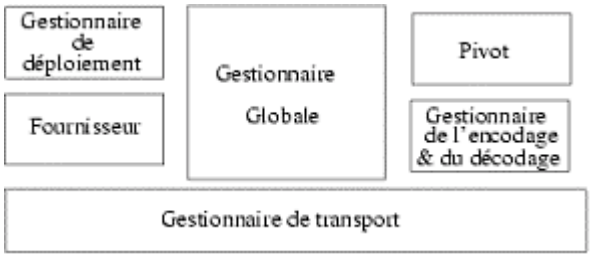


Figure 5-4 Container CSOAP

Le *gestionnaire de déploiement* gère la configuration d'un service Web. Plus précisément, une fois qu'un service Web est implémenté, il est nécessaire de définir au niveau du serveur SOAP un ensemble d'informations permettant à ce dernier de traiter de façon automatique les accès à ce service Web. Pour cela, un fichier de configuration est disponible. Il est exprimé en XML et contient toutes les informations relatives aux services, à savoir son nom, les classe qu'il

[29] http://java.sun.com/xml/jaxrpc/.

implémente, ses méthodes, les paramètres d'entrée et de sortie de ses méthodes, les types dans lesquels sont décodés et encodés les paramètres d'entrées et de sortie de ses méthodes, et la chaîne de traitement devant être suivie pour l'invoquer. Cette chaîne de traitement indique notamment si des traitements supplémentaires (e.g., compression du message SOAP) doivent être effectués.

Les autres composants présentés dans la figure 5-4 gèrent quant à eux les invocations des services Web. Le *gestionnaire de transport* gère la réception et l'acheminement des messages suivant le protocole de transport utilisé. Actuellement, seul le protocole HTTP est géré mais, d'autres protocoles tels que SMTP ou FTP peuvent normalement être utilisés. Le gestionnaire de transport garantit l'indépendance de CSOAP avec le protocole de transport sous-jacent. Le *gestionnaire d'encodage et de décodage* gère l'encodage et le décodage des messages SOAP.

Le *pivot* (intercepteur) procède au prétraitement des messages SOAP. Plus précisément, certains traitements supplémentaires peuvent devoir être effectués sur un message SOAP avant son envoi ou avant d'invoquer une méthode d'un service Web. Par exemple, pour des raisons de sécurité, l'en-tête du message SOAP peut contenir des informations relatives à des droits d'accès devant être traités avant d'invoquer l'opération. De même, un message SOAP peut par exemple être compressé avant d'être envoyé. Le pivot redirige alors le message vers le composant capable de réaliser ce traitement.

Le *fournisseur* charge l'instance du service Web lorsque toute la chaîne de traitement du message SOAP a été effectuée. Enfin, de *Gestionnaire globale* gère la coordination entre les différents composants afin de pouvoir traiter l'arrivée d'un message SOAP et de renvoyer une réponse en échange.

Le client CSOAP fournit pour sa part une interface aux applications clientes. Cette interface contient les méthodes nécessaires pour que l'application puisse : (i) définir les paramètres de l'invocation d'une opération de service et (ii) l'invoquer. L'accès à une instance d'un service Web nécessite au préalable une localisation dynamique des instances dans le réseau *ad hoc*. Nous présentons dans la section suivante l’implémentation du protocole de localisation de services Web.

5.4.2 Service de localisation

Le service de localisation d'Ariadne est constitué de deux composants : le gestionnaire de localisation de services, et le répertoire (voir figure 5-5).

Le *composant répertoire* est déployé dynamiquement et sélectivement sur certains terminaux du réseau. Le répertoire effectue trois tâches majeures. Tout d'abord, il procède à l'enregistrement des services Web mis à disposition sur le terminal ou sur d'autres terminaux. A cet effet, il gère les informations relatives à l'enregistrement des services Web. De plus, il accomplit la localisation des services Web pour les applications clientes déployées sur le terminal hôte ou pour les services de localisation des autres terminaux. Cette localisation inclut d'une part la gestion des interactions les services de localisation, et d'autre part la collaboration avec les répertoires déployés dans le réseau.

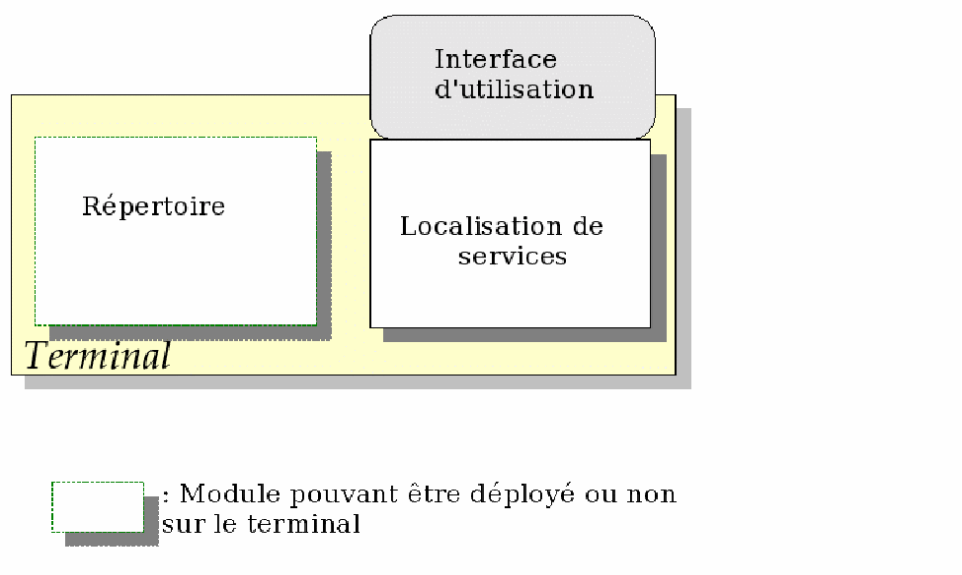


Figure 5-5 Architecture de localisation de services Web

Le *gestionnaire de localisation* est pour sa part déployé sur tous les terminaux du réseau. Il fournit une interface permettant d'utiliser le service de localisation. Plus précisément, cette interface permet à un service Web disponible sur l'hôte de s'enregistrer auprès du répertoire et à une application cliente de pouvoir interroger le répertoire, ce répertoire pouvant être déployé sur l'hôte lui-même ou sur un autre terminal. Ainsi, le gestionnaire de localisation traite les demandes d'enregistrement, de *désenregistrement* et d'interrogation, faites à travers l'interface, en faisant suivre les messages correspondants vers le répertoire déployé sur un autre terminal ou sur l'hôte. De plus, il effectue le déploiement des répertoires dans le réseau en procédant à leur élection et en maintenant les informations relatives aux répertoires déployés. Nous

définissons de façon plus précise la structure du service de localisation dans la section suivante.

5.4.3 Structure du service de localisation

Comme l'illustre la figure 5-6, le gestionnaire de localisation de services est constitué d'un gestionnaire de transport, de communication, d'élection, de localisation des répertoires, de traitement des interrogation et des enregistrement, d'un *déployeur* et d'une interface. Nous détaillons ci-après les fonctionnalités offertes par chacun de ces éléments.

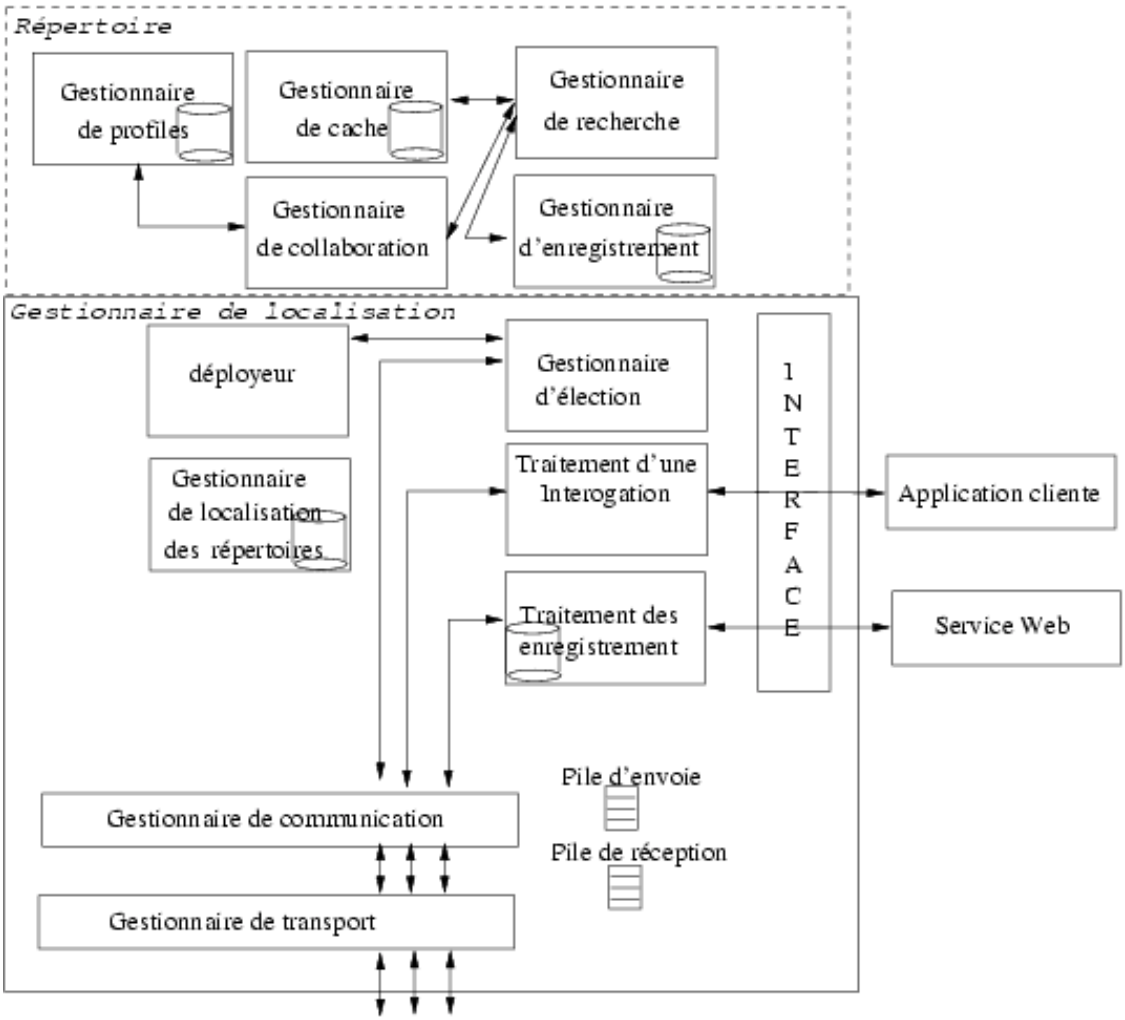


Figure 5-6 Structure du service de localisation

Le *gestionnaire de transport* correspond à un processus *démon* (*deamon*) qui gère la réception des messages en les stockant dans une pile, appelée pile de réception, et l'envoi des messages se trouvant dans une pile, appelée pile d'envoie.

Le *gestionnaire de communication* est un élément central du service de localisation. Il gère l'encodage des messages avant de les stocker dans une pile d'envoi, et le décodage des messages se trouvant dans la pile de réception. De plus, il assure l'extraction des données des messages et les redirige vers les composants (e.g., gestionnaire d'élection, gestionnaire de localisation des répertoires) devant les gérer.

Le *gestionnaire d'élection* est responsable de l'élection des répertoires sur le réseau *ad hoc*. Plus précisément, si aucun terminal ne dispose d'un répertoire procédant à la localisation des services Web dans son *N*-voisinage, alors le gestionnaire d'élection initie un processus d'élection. Pour cela, il interagit avec le gestionnaire de communication afin que ce dernier diffuse une requête d'élection et collecte les réponses renvoyées par les terminaux interrogés. Puis, il traite les réponses reçues et sélectionne le terminal devant déployer un répertoire. Il lui envoie une annonce *via* le gestionnaire de communication. Le gestionnaire de transport et de communication du terminal recevant cette annonce transmet cette information au *déployeur*. Le *déployeur* est responsable du déploiement du répertoire sur l'hôte.

Le *gestionnaire de localisation* met à disposition une interface qui peut être utilisée par les applications clientes (pour localiser un service) et les services Web (pour être publiés) de l'hôte. Le *gestionnaire de traitement des enregistrements* conserve les informations relatives aux services Web offerts par l'hôte et les fait également suivre au répertoire le plus proche. Notons que le répertoire le plus proche est désigné par le *gestionnaire de localisation des répertoires* qui conserve les informations relatives aux répertoires présents.

Le *gestionnaire d'interrogation* vérifie si l'instance du service Web demandée est disponible sur l'hôte. Si nécessaire, il fait suivre cette requête vers le répertoire le plus proche *via* le gestionnaire de communication.

Le composant répertoire dispose de plusieurs gestionnaires, utilisés afin de stocker les informations relatives à la localisation des services Web. Ces derniers correspondent à un gestionnaire de profil, de cache et des informations relatives aux services Web. Le *gestionnaire d'enregistrement* des services Web gère les informations portant sur les instances des services de localisation des terminaux mis à disposition par l'hôte par d'autres terminaux. Cela inclut l'ajout (respectivement suppression) de ces informations suite à l'enregistrement

(respectivement *désenregistrement*) d'un service Web. Le *gestionnaire de cache* effectue la gestion du cache conservant les informations relatives aux instances des services Web suivant une politique de placement et de remplacement. Le *gestionnaire de profil* génère un résumé du contenu du répertoire de l'hôte, et interagit avec le gestionnaire du profil locale du module de localisation afin d'obtenir une caractérisation de l'hôte. De plus il gère les profils des autres répertoires présents dans le réseau.

Finalement le *gestionnaire de recherche* interagit avec tous ces composant afin de localiser les instances des services Web disponibles suivant le fonctionnement du protocole que nous avons introduit dans le chapitre 4.

5.4.4 Déploiement et utilisation du service de localisation

Le service de localisation peut être déployé de deux façons différentes. Soit le service de localisation est déployé en tant que service Web, soit il est déployé en tant qu'application. Dans le premier cas, une application cliente ou un service Web utilisent le container SOAP pour interagir avec le service de localisation. Notons que le déploement du service de localisation en tant que service Web permet, de part les propriétés de ces derniers, de garantir l'interopérabilité enter les applications déployées sur l'hôte et le service de localisation. Dans le second cas, ils utilisent l'API Java (*Application Program Interface*) du service de localisation.

L'API ou l'interface Web du service de localisation offrent un ensemble de primitives pouvant être utilisées par une application cliente ou un service Web. Nous décrivons ces primitives ci-après. Nous illustrons cette description en présentant à titre d'exemple un programme interagissant avec le service de localisation à travers son API.

La première primitive de l'interface Web du service de localisation permet d'enregistrer un service auprès du répertoire local :

```
public void register(String AbstractDefinition, String EndPoint)
{}
```

Elle possède deux paramètres d'entrée : la description abstraite WSDL, et celle du point d'accès du service. Nous présentons ci-dessous un exemple d'enregistrement d'un service

Web en utilisant l'API du service. Cet enregistrement est composé de trois étapes. La première consiste à accéder à la description abstraite du service Web :

```
String wsdlFilePath="/wsdl/MySrvAbstract.wsdl";
 java.io.FileInputStream wsdlAbstractFile =
   new java.io.FileInputStream(wsdlFilePath);
 while (wsdlAbstractFile.available() >0)
 {
  byte[] data=new byte[wsdlAbstractFile.available()];
  wsdlFile.read(data);
  AbstractDescription=AbstractDescription+
              (new String(data));
  }
```

Algorithme 5-1

La deuxième étape consiste à fournir les informations relatives au(x) point(s) d'accès du service, ce qui inclut l'adresse de l'hôte, le port réseau, le chemin d'accès au service et le nom du service :

```
 String hostAddress="128.10.10.10";
 String port="8080";
 String servicesContext="services";
 String serviceName="myService";
 String EndPoint="http://"+hostAddress+:"+port+
           "/"+servicesContext+"/"+serviceName;
 //EndPoint=http://128.10.10.10:8080/services/myService
```

Algorithme 5-2

Enfin, la dernière étape consiste à enregistrer ce service :

```
sd.register(AbstractDescription,EndPoint,ProtocolFormat);
```

La deuxième primitive de l'interface Web du service de localisation correspond à la primitive de *désenregistrement* :

```
public void unregister(String AbstractDefinition, String EndPoint)
{
}
```

Cette primitive possède deux paramètres d'entrée, à savoir la description abstraite WSDL et la description du point d'accès du service. Elle permet de désenregistrer le service auprès du répertoire local. Ce désenregistrement nécessite plusieurs étapes. La première étape consiste à extraite la description abstraite de WSDL (Algorithme 5-1) puis à extraire les informations portant sur le point d'accès (Algorithme 5-2) et enfin à *desenregistrer* le service :

```
sd.unregister (wsdlDescription, EndPointURL);
```

Enfin, un client désirant localiser un service peut fournir au protocole la description entière du service Web qu'il souhaite accéder ou une description partielle de ce dernier. A cet effet, le client utilise la primitive *getServices* permet de localiser l'instance d'un service dans le réseau. Cette primitive prend en paramètre la description abstraite de la spécification WSDL du service que le client souhaite accéder. Elle renvoie en échange un tableau dont chaque élément correspond à la classe *GetServiceData*.

```
public GetServiceData[] getServices(String wsdl)
{
}
```

Cette classe est constituée du point d'accès du service, de la description du point d'accès, de la description abstraite de WSDL:

```
Package SDService;
Public class GetServiceData                implements java.io.Serializable {
  Private java.lang.String serviceEP;
  Private java.lang.String wsdlDocument;
  Public GetServiceData() {
  }
```

Nous récapitulons ci-dessous quelles sont les étapes à suivre pour accéder à un service. La première consiste à obtenir la description abstraite WSDL du service (voir Algorithme 5-1). La deuxième étape consiste à demander à Ariadne de localiser la ou les instances des services Web correspondantes.

```
SDService.GetServiceData[] services = sd.getServices(partialDescription);
```

La troisième étape consiste à extraire les informations reçues :

```
if (services != null && services.length > 0)
 {
  String serviceEndPoint=services[0].getServiceEP();
 }
```

Enfin, la dernière étape consiste à accéder au service souhaité :

```
java.net.URL serviceURL= new java.net.URL(serviceEndPoint);
ServiceInterface service=new Service.ServiceBindingStub(serviceURL,null);
```

5.4 Évaluation

Nous présentons dans cette section une série d'expérimentations effectuées afin d'évaluer les performances de l'intergiciel Ariadne et aussi de les comparer avec celles des intergiciels existants. Plus précisément, nous avons concentré notre évaluation sur les performances de CSOAP (§ 5.4.6) et sur celles du service de localisation (§5.4.7). Cela s'explique par le fait que les performances associées aux opérations fournies par le service Web découlent directement de celles de la plate-forme d'exécution Java.

5.4.1 Évaluation de l'accès aux services Web

Afin d'évaluer les performances du protocole CSOAP, nous avons développé un service Web. Ce service offre comme fonctionnalité une opération qui prend en entrée un tableau d'entiers et fournit en retour une chaîne de caractères. Du fait de la simplicité de l'opération exécutée, le temps de réponse suite à l'invocation de l'opération correspond au temps nécessaire pour transporter le message SOAP, l'encoder et le décoder, en extraire les paramètre d'entrée et de sortie et l'exécuter. Lors de ces expérimentations nous avons utilisé des PDAs et des ordinateurs portables. Nous présentons ci-dessous leurs caractéristiques. Les PDAS correspondent à des Compaq IPAQ 3600 disposant de 206 MHz de CPU et de 32 MB de

RAM et ayant comme système d'exploitation Linux et la distribution Familiar 0.6 et comme environnement d'exécution J2ME CDC 1.0.1 JVM. Les ordinateurs portables sont des Compaq PC ayant 500 MHz de CPU et 192 MB de RAM. Ils disposent du système d'exploitation Linux et de la distribution Redhat 7.3 et comme environnement d'exécution de la J2ME CDC JVM.

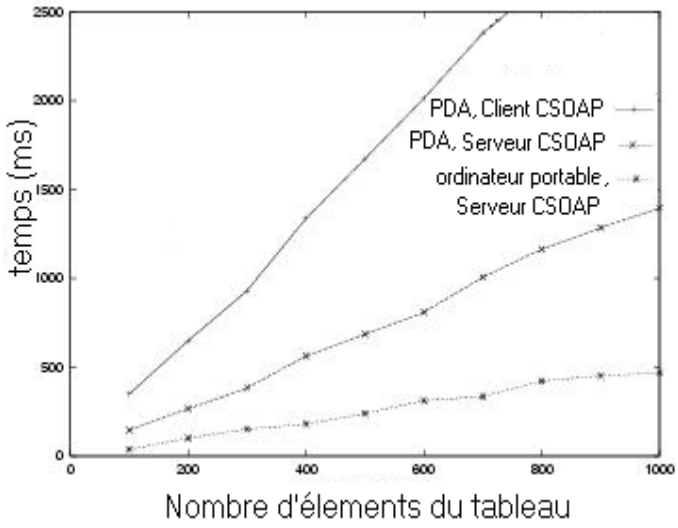


Figure 5-7 Temps de réponse CSOAP pour un tableau de grande taille

Les figures 5-7 et 5-8 présentent le temps d'attente perçu par le client entre le début de l'invocation et l'obtention du résultat, ainsi que le délai entre le début de l'invocation par le client et l'exécution de l'opération au niveau du serveur, en fonction de la taille du tableau fourni en entrée, variant de 10 à 100 (figure 5-8) voir 1000 éléments (figure 5-7).

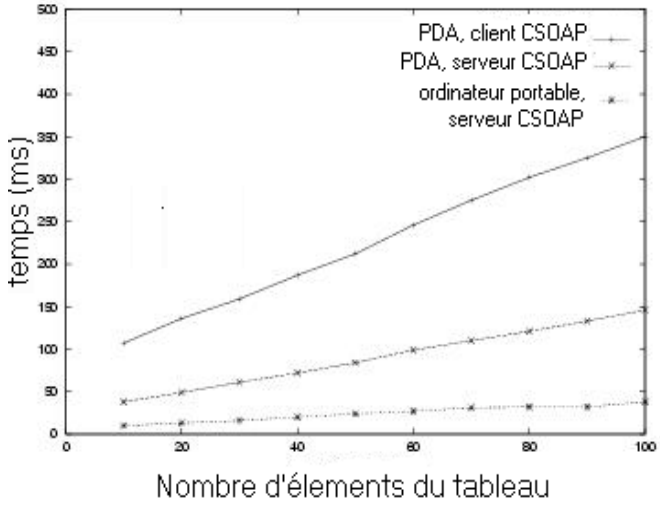


Figure 5-8 Temps de réponse CSOAP pour un tableau de taille réduite

Ces deux figures comparent aussi ces temps d'attente lorsque le serveur CSOAP est déployé sur un PDA ou sur un ordinateur portable et cela afin de montrer l'impact qu'ont les capacités des terminaux sur le traitement des messages lors de l'invocation d'une opération au niveau du serveur. Ainsi, le traitement d'un message comprenant 100 paramètres dans le tableau nécessite environ trois fois plus de temps de traitement sur un PDA que sur un ordinateur portable (figure 5-8). Lors de nos expérimentations, nous avons remarqué que 80 % du temps utilisé par un serveur pour traiter une invocation (incluant le temps entre la réception du message et l'exécution de l'opération) était utilisé pour filtrer le message XML. Notons que la figure 5-7 illustre particulièrement bien l'impact du traitement XML sur le temps de traitement pour le client lorsque la taille du tableau augmente.

La figure 5-9 compare quant à elle les performances de CSOAP avec celles d'AXIS en évaluant le délai entre le début de l'invocation d'une opération et l'obtention de la réponse au niveau du client. Le serveur et le client, pour CSOAP et AXIS, sont déployés sur un ordinateur équipé d'un processeur Intel PIII de 800 MH et de 384 MB de RAM, et ayant comme système d'exploitation Linux-Redhat 7.3 ainsi que la JVM J2SE 1.3.1. Le temps d'attente induit par CSOAP est environ 10 % inférieur à celui d'AXIS. Toutefois, nous envisageons de réduire cette différence en développent un analyseur syntaxique d'XML plus efficace.

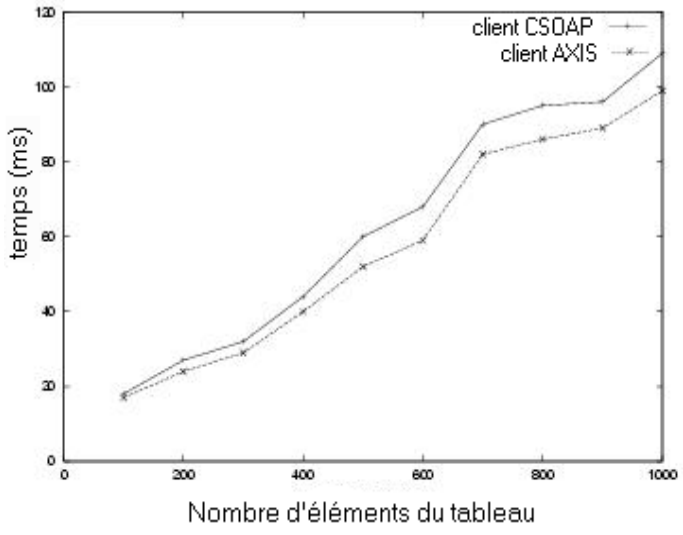


Figure 5-9 CSOAP versus AXIS

Enfin, la figure 5-10 compare les performances de CSOAP avec celles de Java RMI. Java RMI fait partie des intergiciels utilisés traditionnellement pour accéder aux fonctionnalités offertes par des terminaux distants dans un environnement mobile. Pour cette expérimentation, nous utilisons pour le client Java RMI les paquetages optionnels de Sun et pour le serveur l'implémentation pour les plateformes Intel du répertoire RMI[30]. Le client et le serveur CSOAP et Java RMI sont exécutés sur des ordinateurs portables interagissant via un réseau *ad hoc* WLAN d'un débit de 11MBps.

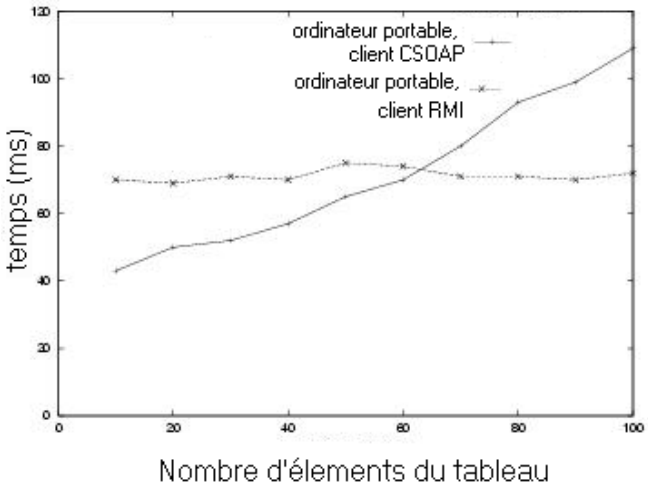


Figure 5-10 CSOAP versus Java RMI

La figure fournit le temps de réponse induit par un appel à une procédure, et perçu par le client. Les résultats obtenus montrent que CSOAP offre de meilleures performances pour un tableau dont la taille est inférieure à 60 éléments, alors que Java RMI est plus efficace pour un tableau de plus de 60 éléments. Toutefois, si nous considérons l'intelligence ambiante en tant que contexte d'application, il semble plausible que la plupart des opérations distantes invoquées demandent comme paramètre moins de 60 éléments. En effet, si nous considérons un service Web particulièrement complexe d'achat en ligne tel que celui de ebay[31], la description WSDL du service Web correspondant contient 30 opérations. Si nous considérons le service Web de recherche mis à disposition par le moteur de recherche Google[32], l'interface correspondante est constituée de trois opérations (*GetCachePagem, doSpellingSuggestion,* et *doGoogleSearch*). Nous pouvons donc conclure qu'au niveau de l'invocation des opérations distants, les performances d'Ariadne comparée à celles d'intergiciels Java RMI sont meilleures et donc que le fait d'offrir une solution basée sur les services Web est préférable du

[30] http://java.sun.com/rmi/
[31] http://www.ebay.com/
[32] htt

point de vue des performances mais aussi pour supporter des services ubiquitaires qui nécessitent des interactions à peu de paramètres.

5.4.2 Évaluation du service de localisation

Nous évaluons dans cette section les performances en terme de délai de réponse de notre service de localisation de services Web.

Ces évaluations ont été réalisées sur des terminaux correspondant à des ordinateurs (ou PC), des ordinateurs portables et des PDAs. Nous définissons ci–dessous les caractéristiques de ces terminaux. Les PDAs correspondent à des Compaq iPAQ 3600 disposant d'un processeur Strong-ARM 206Mhz et de 32 MB de RAM, ayant comme système d'exploitation Linux Familiar 0.6 et la JVM J2ME CDC 1.01 personal profile. Les ordinateurs portables sont des Compaq PC ayant un processeur PIII de 500Mhz et 192 Mb de RAM. Ils utilisent le système d'exploitation Linux Redhat 7.3 et la JVM J2SE 1.4.2. Enfin les PC disposent de processeurs P4 de 2.6Ghz et ont 512Mb de RAM et comme système d'exploitation Redhat 9.0 et la JVM J2SE 1.4.2. Les cartes sans fils utilisées lors de ces expérimentations sont des cartes Lucent IEEE 802.11b offrant 11Mbps de débit.

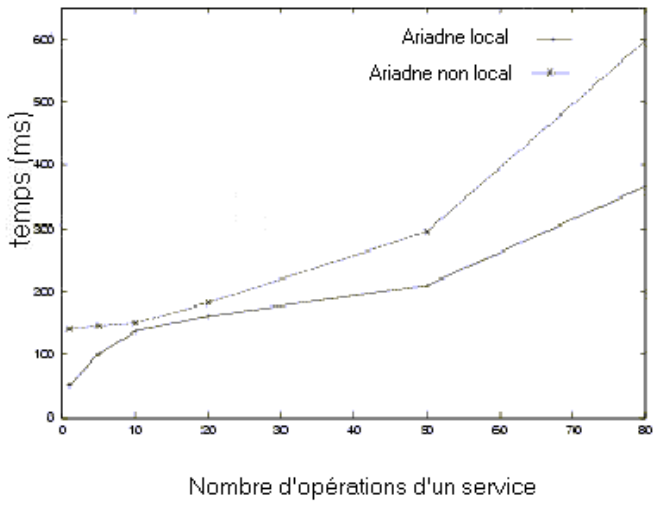


Figure 5-11 Temps de réponse lors d'une localisation locale ou non locale

Tout d'abord, la figure 5-11 donne le délai entre l'envoi par une application cliente d'une requête de localisation et la réception (par cette application) des informations relatives aux instances de services Web disponibles sur le réseau. De plus, cette figure compare le délai lorsque les services Web sont déployés sur l'hôte même et sur d'autres terminaux. Un *déploiement locale*, c'est-à-dire sur l'hôte, signifie que les services Web, l'application cliente,

et le répertoire sont déployés sur un seul ordinateur portable, et que la recherche des instances de services se restreint aux instances disponibles sur l'hôte. Au contraire un *déploiement non local* indique que le répertoire et les services Web sont installés sur un ordinateur portable sans fil et que l'application cliente utilise le réseau sans fils *ad hoc* pour communiquer avec le répertoire. Cette évaluation est effectuée en fonction du nombre d'opérations contenues dans la description du service Web, et cela afin de montrer notamment l'impact qu'a sur les performances, l'analyse des descriptions XML des services Web effectuée par le répertoire lors de la localisation d'une instance. Plus précisément, la figure montre que le temps induit par l'analyse XML des descriptions de services par le répertoire augmente en moyenne de 2,6 ms par opération traitée, alors que le coût de traitement du reste de la description du service est fixe et est égale à d'environ 50 ms. Lorsque le déploiement de Ariadne est effectué localement et non localement, la différence, en terme de délai de réception d'une réponse par l'application cliente, provient de l'acheminement des messages (requêtes de localisation et réponses) entre l'application cliente et le répertoire distant, et du traitement des messages (dés-encapsulation et encapsulation). Plus précisément, ce délai est en moyenne de 112ms et son augmentation est en moyenne deux fois supérieure à l'augmentation du temps nécessaire pour effectuer le traitement XML des descriptions de services au niveau répertoire.

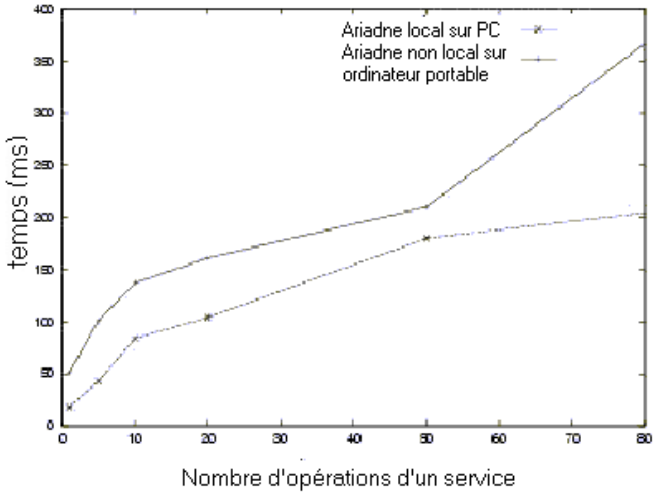


Figure 5-12 Comparaison du temps de réponse pour une localisation entre PC et ordinateur portable

La figure 5-12, compare le délai de réponse, lorsque la localisation des ressources est effectuée localement par un ordinateur portable et un PC en fonction du nombre d'opérations contenues dans la description du service Web, et cela afin de comparer les performances du protocole suivant les capacités de terminaux. Lorsque le nombre d'opérations est inférieur à 50, le délai de réponse de l'ordinateur portable par rapport du PC, est environ une fois et

demie plus élevé, puis ensuite entre 50 et 80 opérations, ce délai s'accentue jusqu'à être deux fois plus important pour 80 opérations.

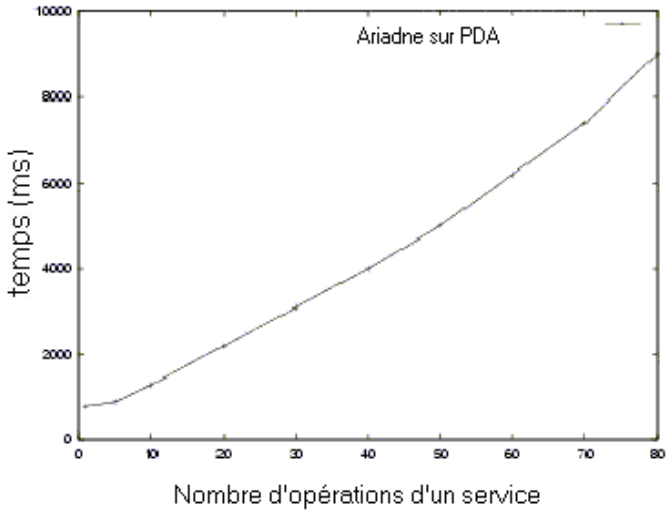


Figure 5-13 Temps de réponse lors de la localisation avec un PDA

Les figures 5-13 et 5-14 présentent le délai de réponse induit par la localisation non-locale des instances de services à partir de PDAs en fonction du nombre de services disponibles (figure 5-14) et du nombre d'opérations fournies par chaque service Web (figure 5-13). Plus précisément, dans le cas de la figure 5-14 nous avons supposé que chaque service Web mettait à disposition deux opérations. On observe que, lorsque le nombre de services croit, le délai d'attente fournit par la figure 5-13 augmente faiblement comparé à l'augmentation du délai d'attente induit par la localisation d'un service Web dont le nombre d'opérations fournies augmente (figure 5-13). Cela provient du fait que le répertoire abandonne l'analyse des descriptions abstraites des services dès lors qu'il s'aperçoit que la description abstraite d'un service ne correspond pas à celle du service recherché.

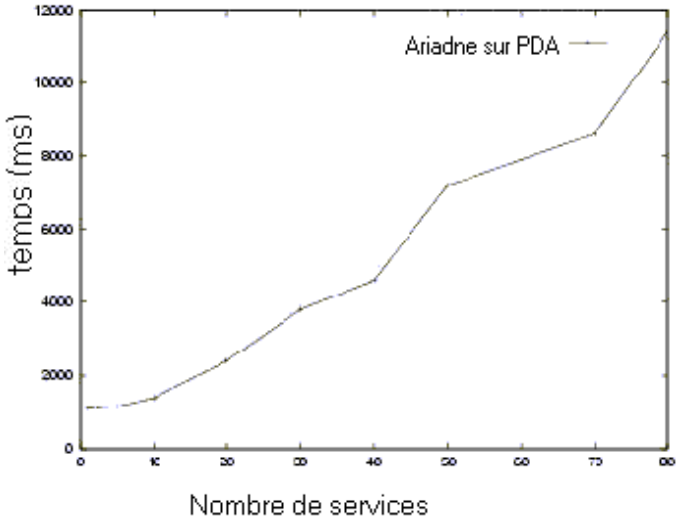


Figure 5-14 Temps de réponse en fonction du nombre d'instances de services disponibles

La figure 5-15 compare le temps d'attente induit par la localisation des ressources entre Ariadne et d'UPnP[33], en fonction du nombre d'opérations offertes par un service Web. Pour cette expérimentation, nous utilisons l'implémentation java UPnP Cibergarage[34]. La localisation des instances des services est effectuée localement sur un PC. La figure montre que les performances de Ariadne sont meilleures que celles d'UPnP lorsque le nombre d'opérations offertes par le service et incluses dans sa description est inférieur à 45. Au contraire UPnP offre de meilleures performances lorsque le nombre d'opérations définies dans la description du service est supérieur à 45 opérations. Notons que des services Web déployés dans un environnement mobile et dans le contexte de l'intelligence ambiante offrent typiquement moins de 20 opérations. Par conséquent, dans le contexte de l'informatique diffuse les performances de notre protocole sont supérieures à celle de UPnP.

[33] http://www.upnp.org
[34] http://www.cybergarage.org

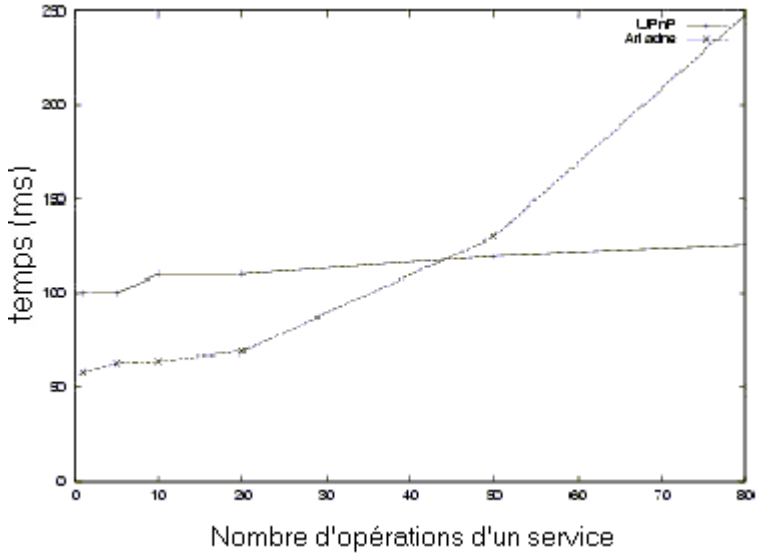


Figure 5-15 Temps de réponse lors de la localisation pour Ariadne versus UPnP

5.5 Conclusion

Supporter le développement de l'intelligence ambiante ou du *pervasive computing* fait partie des nouveaux défis technologiques posés à l'heure actuelle. Les dernières technologies en matière de communication sans fil et de dispositifs numériques, permettent de répondre en partie à ce défi. Les réseaux *ad hoc* notamment jouent un rôle fondamental dans le contexte de l'intelligence ambiante en fournissant un connectivité aux terminaux, et ce même si aucune infrastructure de communication n'est déployée. Toutefois, offrir aux utilisateurs un accès à des contenus ou à des services à tout moment et n'importe où, à partir d'un terminal, nécessite de définir de nouvelles plateformes ubiquitaires. Ces plates-formes doivent rendre accessible les services et contenus aux utilisateurs tout en supportant des interactions transparentes[35] entre l'utilisateur et les services offerts.

A cet effet, nous avons proposé l'intergiciel Ariadne qui effectue la localisation dynamique des instances de services Web disponibles, tout en gérant les interactions entre le terminal de l'utilisateur et ces services. De plus, cet intergiciel se base sur les technologies des services Web et donc sur des standards universellement acceptés et utilisés. Ainsi, Ariadne peut opérer dans des environnements distribués particulièrement hétérogènes, au niveau des plates-formes

[35] du point de vue de l'utilisateur.

logicielles, tels que des réseaux *ad hoc* ou hybrides. tout en pouvant être déployé sur des terminaux ayant des capacités limitées. Par ailleurs, nous avons présenté le prototype de l'intergiciel Ariadne. Ce prototype permet de déployer des services Web sur n'importe quel type de terminal, ce qui inclut des terminaux ayant des capacités restreintes tels que des PDAs. Nous avons évalué les performances du prototype, en terme de temps de réponse, et comparé ces performances à celles des intergiciels existants. Les résultats obtenus montrent que les performances du prototype, au niveau de la localisation et de l'accès de services Web, sont équivalentes à celles des intergiciels traditionnels, basés ou non sur les technologies des services Web. Cela valide donc notre approche du point de vue de l'exploitation des services Web pour l'informatique diffuse dans les réseaux *ad hoc*.

Chapitre 6 Conclusion

Les dernières décades ont été caractérisées par l'émergence de systèmes de communication permettant aux utilisateurs d'accéder à tout moment et n'importe où aux ressources qu'ils désirent. Cela a notamment été rendu possible par la miniaturisation des dispositifs électroniques et le faible coût des terminaux (e.g., PDAs, ordinateurs ou téléphones portables) qui sont à l'origine de la popularité[36] croissante des terminaux sans fil auprès des utilisateurs. De plus, le développement des technologies sans fil et l'expansion des réseaux sans fil *ad hoc (e.g.,* WLAN, BLUETOOTH) ou basés sur des infrastructures (GSM, GPRS, UMTS, WLAN) ont permis de supporter la mobilité des utilisateurs et des terminaux.

Dans ce contexte, les réseaux *ad hoc* jouent un rôle fondamental puisqu'ils permettent aux utilisateurs nomades d'obtenir une connectivité spontanée au réseau lorsque des infrastructures de communication ne sont pas déployées. De plus, ils peuvent être utilisés à un faible coût financier et être interconnectés à d'autres types de réseaux (filaire, sans fil, à base d'infrastructure) et offrir ainsi une connectivité globale aux utilisateurs. Pour ces raisons, les réseaux *ad hoc* sont exploitables par différentes applications comme l'informatique diffuse à destination du grand public, et les applications dédiées, militaires ou de secours. Toutefois, l'exploitation des ressources disponibles dans un réseau *ad hoc* nécessite une localisation dynamique de ces ressources. Divers protocoles permettant de localiser des ressources aussi variées que des services multimédias ou des ressources matérielles, ont été proposés pour les réseaux filaires. Ces protocoles se basent sur une approche centralisée. Ces solutions ont ensuite été adaptées afin de pouvoir être utilisées dans des réseaux sans fil basés sur des infrastructures. Toutefois, ces solutions centralisées sont inexploitables dans les réseaux *ad hoc* en raison du manque de connectivité. A cet effet, nous avons proposé dans cette thèse des solutions permettant d'assurer la localisation et l'accès aux ressources dans un réseau *ad hoc*, en prenant en compte leurs caractéristiques (densité, surface, trafic). Notons que les caractéristiques d'un réseau *ad hoc* influencent les choix de conception des protocoles de communication. Cela est notamment illustré par les protocoles de routage : si le nombre de

[36] Selon *Frost and Sullivan*, l'industrie du LAN sans fil, qui a dépassé les 300 millions de dollars en 1998, représente 1,6 milliards en 2005.

terminaux et de trafic transitant est faible, l'utilisation d'un protocole réactif est plus indiquée ; en revanche, si le réseau est dense et le trafic important, un protocole proactif ou hybride est préconisé. Pour ces raisons, nous avons abordé distinctement la localisation de ressources dans les réseaux *ad hoc* à petite et large échelle.

Pour les réseaux *ad hoc* à petite échelle, nous avons proposé une solution entièrement distribuée dans laquelle les terminaux coopèrent afin de localiser les ressources dans le réseau. Cette coopération est effectuée en plusieurs étapes de façon à limiter le trafic généré sur le réseau et par là même la consommation énergétique résultant des échanges de messages. Plus précisément, un terminal souhaitant accéder à une ressource diffuse dans son voisinage une requête de localisation. Puis, il interroge si nécessaire les terminaux se trouvant à l'extérieur de ce voisinage de façon ciblée. Pour sélectionner efficacement les terminaux interrogés, nous nous basons sur des profils qui intègrent une caractérisation, sous une forme compacte, des capacités physique du terminal (e.g., énergie disponible) et des centres d'intérêts des utilisateurs (e.g., descriptions des ressources fréquemment accédées). La portée limitée de la diffusion d'une requête et la sélection des terminaux distants interrogés permet de réduire le trafic généré. De plus, afin de limiter le trafic généré sur le réseau *ad hoc*, le protocole de localisation de ressources tire profit des infrastructures réseau disponibles en redirigeant les requêtes de localisation vers ces infrastructures. Par ailleurs, lorsque les ressources correspondent à des contenus volumineux, le protocole utilise des caches lui permettant de stocker ces contenus. Cela permet de réduire à la fois le trafic généré et le temps d'attente induit par l'échange de contenus. Nous définissons en outre des techniques de préchargement prenant en compte les ressources énergétiques du terminal hébergeant le cache, et les habitudes de l'utilisateur. Les performances de notre protocole de localisation ont été évaluées en terme d'énergie dépensée par les terminaux, qui est la ressource la plus critique dans un réseau *ad hoc*. Les résultats de cette évaluation montrent que notre protocole est performant dans le cas de réseaux à faible densité, dédiés à des applications dont les participants collaborent fortement.

Pour les réseaux à large échelle, à forte densité et large surface, nous avons proposé une approche semi-distribuée basée sur des répertoires effectuant la localisation des ressources pour l'ensemble des terminaux. Cette approche est adaptée aux réseaux à grande échelle car le nombre de terminaux interrogés est restreint, ce qui limite le nombre de requêtes de localisation et de réponses renvoyées. Les répertoires sont déployés de façon homogène et

dynamique sur le réseau afin de répartir équitablement la charge entre les répertoires. Ce déploiement est périodiquement réévalué en fonction des changements de topologie survenant dans le réseau de façon à ce que les terminaux puissent contacter à tout moment un répertoire gérant pour eux la localisation de ressources. Enfin, notre protocole de localisation effectue aussi bien la localisation des ressources dans les réseaux *ad hoc* que dans les réseaux auxquels il est connecté. Ainsi, cela permet d'intégrer des environnements différents comme les réseaux filaires et les réseaux sans fil. Nous avons réalisé une évaluation des performances de notre protocole de localisation de ressources en termes de trafic généré et de temps de réponse, montrant son applicabilité dans les réseaux à large échelle. Nous avons également introduit une mise en œuvre de ce protocole de localisation pour un intergiciel dédié à l'informatique ubiquitaire. Cet intergiciel garantit l'interopérabilité entre les terminaux, malgré leur l'hétérogénéité au niveau matériel et logiciel, en se basant sur l'architecture de services Web. Précisément, nous avons développé l'intergiciel Ariadne qui permet le déploiement de services Web sur des terminaux sans fil, mobiles, de relativement faible capacité, et leur localisation et accès dans des réseaux *ad hoc* à large échelle. A partir du prototype d'Ariadne, nous avons mené une série d'expérimentations afin d'évaluer les performances de la localisation et de l'accès aux ressources. Les résultats obtenus, comparés à ceux des intergiciels gérant l'accès et/ou la localisation des services, sont équivalents. Cela valide notre approche du point de vue de l'exploitation des services Web pour l'informatique diffuse dans les réseaux *ad hoc*.

Nos solutions complètent les travaux du domaine des réseaux MANETs qui sont, pour les plus aboutis, focalisés sur les protocoles de routage. En effet, les protocoles de localisation de ressources que nous avons introduits permettent l'exploitation effective des réseaux *ad hoc* au niveau applicatif, en fournissant les mécanismes nécessaires à la localisation et à l'accès aux contenus et services présents dans le réseau. Toutefois, le développement d'applications pour réseaux *ad hoc* soulève encore de nombreux défis, qui requièrent de définir des méthodes, outils et plate-forme d'exécution adaptés. Un défi clé est notamment de faciliter le développement d'applications efficaces en terme de trafic généré et plus généralement de ressources consommées. Pour répondre à ce problème, l'approche que nous avons utilisée dans cette thèse est d'utiliser des techniques de *cross-layer* afin d'obtenir et de partager des informations entre la couche de routage et nos protocoles de localisation. Une approche plus générique consiste à proposer un intergiciel offrant différents services de base pour les applications destinées aux réseaux *ad hoc*. Le but est à la fois d'offrir une palette de

mécanismes standard (diffusion, réplication, distribution, gestion de groupes, surveillance du réseau ou des terminaux), mais aussi de partager les informations[37] obtenues par ces mécanismes. Il en découle d'une part que les applications se basent sur des solutions optimales pour assurer certaines tâches, et d'autre part qu'elles n'exécutent pas les mêmes mécanismes de façon concurrente.

[37] Informations de déploiement, localisation de terminaux, topologie du réseau, état de l'hôte ou du réseau.

Annexe 1

Message SOAP pour un requête de localisation

```
(1) <?xml version="1.0" encoding="UTF-8" ?>
(2) <SOAP:Envelope
(3)       xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
(4)       SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
(5)       xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
(6)       xmlns:xsd="http://www.w3.org/1999/XMLSchema">
(7)    <SOAP:Header>
(8)      ...
(9)    </SOAP:Header>

(10)   <SOAP:Body>
(11)     ...
(14)  </SOAP:Body>
(15) </SOAP:Envelope>
```

Bibliographie

[1] Napster protocol specification, http://opennap.sourceforge.net/napster.txt, 2000.

[2] S. Agrawal et S. Singh. An experimental study of TCP's energy consumption over a wireless link. In *Proc. of EPMCC*, 2001.

[3] A. Ahuja, S. Agarwal, J.P Singh, et R. Shorey. Performance of TCP over different routing protocols in mobile *ad hoc* networks. In *Proc. of IEEE VTC*, 2000.

[4] R. Atkinson, R. Brendle, P. Conley, et al. UDDI, version 2.03, replication specification. *Technical report*, http://uddi.org, July 2000.

[5] H. Badis et K. Al Agha. Quality of service for ad hoc optimized link state routing protocol. *IETF DRAFT,* http://www.ietf.org, April 2005.

[6] S. Banergee, A. Misra, J. Yeo, et al. Energy-efficient broadcast and multicast trees for reliable wireless communication. In *Proc. of IEEE WCNC*, 2003.

[7] S. Bansal, R. Gupta, R. Shorey, et al. Energy efficiency ad throughput for TCP traffic in multi-hop wireless networks. In *Proc. of IEEE INFOCOM*, 2002.

[8] S. Basagni, I. Chlamtac, B. A. Woodward, et al. A distance routing effect algorithm for mobility (DREAM). In *Proc. of ACM/IEEE Mobicom*, 1998.

[9] R. E. Bellman. On routing problem. *Quarterly of Applied Mathematics*, 16(1), 1958.

[10] R. E. Bellman. The shortest path through a maze. In *Proc. of the International Symposium on the theory of switching*, 1959.

[11] B. Bloom. Space/time trade-offs in hash coding with alowable errors. *Communication of the ACM*, 13(7), 1970.

[12] Bluetooth special interest group. Specification of the bluetooth system 1.0b, volume 1: Core, https://www.bluetooth.org, November 2003.

[13] B.E. Brewington et G. Cybenko. How dynamic is the web?. *Computer Networks*, 33(6), 2000.

[14] M. Cagalj, J.P. Hubaux, et C. Enz. Minimum-energy broadcast in all-wireless networks: NP-completeness and distribution issues. In *Proc. of ACM SIGMOBILE*, 2002.

[15] C. Campo, A. Marín, C. García, et al. Service discovery in pervasive multi-agent systems. In *Proc. of AAMAS Workshop*, 2002.

[16] J.C Cano et P. Manzoni. A performance comparison of energy consumption for mobile ad hoc routing protocols. In *Proc. of IEEE MASCOTS*, 2000.

[17] G. Cao. On improving the performance of cache invalidation in mobile environments. *Mobile Networks and Application*, 7(4), 2002.

[18] G. Cao, L. Yin, et C. Das. Cooperative cache-based data access in ad hoc networks. *IEEE Computer*, 37(2), february 2004.

[19] J. Cartigny, D. Simplot, et I. Stojmenovic. Localized minimum-energy broadcast in all wireless networks. In *Proc. of IEEE INFOCOM*, 2003.

[20] B. Chen, K. Balakrishnan, H. Morris, et al. Span: An energy-efficient algorithm for topology maintenance in ad hoc wireless networks. In *Proc. of ACM/IEEE MOBICOM*, 2001.

[21] L. Cheng et I. Marsic. Service discovery and invocation for mobile ad hoc networked appliances. In *Proc. of the IEEE IWNA*, 2000.

[22] S. Cheshire et M. Krochmal. Multicast dns. Internet draft. http://files.multicastdns.org, february 2004.

[23] T. Clausen et P. Jacquet. Optimized link state routing protocol (OLSR). *IETF RFC 3626*, http://ietf.org, October 2003.

[24] Y. Aumann, H. Wu, et A. Abouzeid. Predicting event sequences: data mining for prefetching web pages. In *Proc. of ACM KDD*, 1998.

[25] S.R. Das, C.E. Perkins, E.M. Royer, et al. Performance comparison of two on-demand routing protocols for ad hoc networks. *IEEE Personal Communications Magazine, special issue on Ad hoc Networking*, 2001.

[26] S. Doshi, S. Bhandare, et T.X. Brown. An on-demand minimum energy routing protocol for a wireless ad hoc network. In *Proc. of ACM SIGMOBILE*, 2002.

[27] R. Droms. Dynamic host configuration protocol. *IETF RFC 2131*, http://ietf.org, March 1997.

[28] L. Feeney et M. Nilsson. Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In *Proc. of IEEE INFOCOM*, 2001.

[29] L. R. Ford et D. R. Fulkerson. Flows in networks. *Princeton University Press*, 1962.

[30] T. Friese, M. Smith, et B. Freisleben. Hot service deployment in an ad hoc grid environment. In *Proc. of ACM ICSOC*, 2004.

[31] Y.Y. Gloand, T. Cai, P. Leach, et al. SSDP: Simple service discovery protocol. *IETF Draft*, http://ietf.org, October 2000.

[32] R. Guimaraes, J. Morillo, L. Cerda, et al. Quality of service for mobile ad hoc: an overview. *Internal Report,* July 2004.

[33] N. Gupta et S.R. Das. Energy-aware on-demand routing for mobile ad hoc networks. In *Proc. of IEEE IWDC*, 2002.

[34] E. Guttman, C. Perkins, J. Veizades, et al. Service location protocol, version 2. *IETF RFC 2608*, June 1999.

[35] J Haas. A new routing protocol for the reconfigurable wireless networks. In *Proc. of IEEE ICUPC*, 1997.

[36] Z. J. Haas et M.R. Pearlman. The performance of query control schemes for the zone routing protocol. In *Proc. of ACM SIGCOMM,* 2001.

[37] T. Hara. Replica allocation methods in ad hoc networks with data update. *Mobile Networks and Applications*, 8(4), 2003.

[38] W. Heinzelman, J. Kulik, et H. Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. In *Proc. of the ACM/IEEE MobiCom*, 1999.

[39] W. R. Heinzelman, A. Chandrakasan, et H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *Proc. of IEEE HICSS*, 2000.

[40] S. Helal, N. Desai, V. Verma, et C. Lee. Konark : A service discovery and delivery protocol for ad-hoc networks. In *Proc. of IEEE WCNC*, 2003.

[41] R. Hermann, D. Husemann, M. moser, et al. Transient ad-hoc networking of pervasive devices. *Computer networks*, 35(4), 2001.

[42] G. Holland et N. Vaidya. Analysis of TCP performance over mobile ad hoc networks. In *Proc. of IEEE/ACM MOBICOM*, 1999.

[43] The Institute of Electrical and Electronic Enginners.Wireless LAN medium access control (MAC) and physical layer (PHY) specifications: High-speed physical layer in the 5 ghz band. *Technical report, IEEE standard 802.11*, 1999.

[44] F. Ingelrest, D. Simplot-Ryl, et I. Stojmenovic. A dominating sets and target radius based localized activity scheduling and minimum energy broadcast protocol for ad hoc and sensor networks. In *Proc. of IFIP MED-HOC-NET*, 2004.

[45] Intel Corp. Mobile power guide rev 1.00, http://intel.com, 1998.

[46] Intel Corp.Mobile power guide rev 2.00,http://intel.com, 2000.

[47] M. Jiang, J. Li, et Y.C Tay. Cluster based routing protocol (CBRP), *IETF draft*, August 1999.

[48] David B. Johnson, D.A. Maltz, et Y.C. Hu. The dynamic source routing protocol for mobile ad hoc networks (dsr), *IETF draft*, http://ietf.org, April 2003.

[49] J.O. Kephart. Research challenges of autonomic computing. In *Proc. of ACM ICSE*, 2005.

[50] Y.B Ko et N.H. Vaidya. Location-aided routing (LAR) in mobile ad hoc networks. *Wireless Networks*, 6(4), 2000.

[51] S.J. Lee, W. Su, et M. Gerla. On-demand multicast routing protocol in multihop wireless mobile networks. *Mobile Networks and Application, special issue on Multipoint Communication in Wireless Mobile Networks*, 7(6), 2002.

[52] F. Li et I. Nikolaidis. On minimum-energy broadcasting in all-wireless networks. In *Proc. of IEEE LCN*, 2001.

[53] S. Lindsey et S. Raghavendra. PEGASIS: power efficient gathering in sensor information systems. In *Proc. of IEEE Aerospace Conference*, 2002.

[54] G. Malkin. RIP version 2, *IETF STD 56*, http://ietf.org, November 1998.

[55] M. K. Marina et S. R. Das. Performance of route caching strategies in dynamic source routing. In *Proc.of WNMC*, 2001.

[56] J. Moy. Ospf version 2. *IETF RFC 2328,* http://ietf.org, April 1998.

[57] K.J Negus, J. Waters, J. Tourrilhes, et al. Homerf and swap: Wireless networking for the connected home. *ACM Mobile Computing and Communications Review*, 2(4), 1998.

[58] J. Ng. *An abstract routing protocol and medium access protocol for mobile ad hoc networks*. PhD thesis, Université Polytechnic, New York, USA, 1999.

[59] M. Nidd. Service discovery in DEAPSPACE. *IEEE Personal Communication*, 8(4), 2001.

[60] R. Ogier, F. Templin, et M. Lewis. Topology dissemination based on reverse-path forwarding (TBRPF). *IETF RFC 3684*, February 2003.

[61] N. Panchal et N. B. Abu-Ghazaleh. Active route cache optimization for ad hoc routing protocols. In *Proc. of IEEE Infocom*, 2002.

[62] V. D. Park et M. S. Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. In *Proc. of IEEE INFOCOM*, 1997.

[63] C. Perkins, E. Belding-Royer, et S. Das. Ad hoc on-demand distance vector (AODV) routing. *IETF RFC 3561,* July 2003.

[64] C.E. Perkins et P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *Proc. of ACM SIGCOMM*, 1994.

[65] J.B. Postel. Transmission control protocol. IETF *RFC 793,* http://ietf.org, September 1981.

[66] G.J. Pottie et W.J. Kaiser. Wireless integrated network sensors. *Communications of the ACM*, 43(5), 2000.

[67] R.C. Prim. Shortest connection networks and some generalizations. *Bell System Technical Journal*, 36(6), 1957.

[68] V. Ramasubramanian, Z.J. Haas, et E. G. Sirer. SHARP: a hybrid adaptive routing protocol for mobile ad hoc networks. In *Proc. of the ACM MobiHoc*, 2003.

[69] P.G. Raverdy et V. Issarny. Context-aware service discovery in heterogeneous networks. In *Proc. of IEEE WoWMoM*, 2005.

[70] R. Rivest. The MD5 message-digest algorithm. *IETF RFC 1321,* http://ietf.org, April 1992.

[71] V. Roduplu et T. Meng. Minimum energy mobile wireless networks. In *Proc. of IEEE JSAC*, 1999.

[72] F. Sailhan et V. Issarny. Cooperative caching in ad hoc networks. In *Proc. of IEEE MDM*, 2003.

[73] R.C. Shah et J. M. Rabaey. Energy-aware routing for low energy ad hoc sensor networks. In *Proc. of IEEE WCNC*, 2002.

[74] H. Singh et S. Singh. Energy consumption of TCP reno, newreno, and SACK in multi-hop networks. In *Proc. of ACM SIGMETRICS*, 2002.

[75] S. Singh et C. Raghavendra. Pamas: Power aware multi-access protocol with signalling for ad hoc networks. *ACM Computer Communications Review*, 28(3), 1999.

[76] S. Singh, M. Woo, et C. Raghavendra. PAMAS: Power-aware routing in mobile ad hoc networks. In *Proc. of IEEE/ACM MOBICOM*, 1998.

[77] K.M. Sivalingam, J.C. Chen, P. Agrawal, et al. Design and analysis of low-power access protocols for wireless and mobile ATM networks. *Wireless Networks*, 6(1), 2000.

[78] C-K Toh. A novel distributed routing protocol to support ad-hoc mobile computing. In *Proc. of IEEE Xplore*, 1996.

[79] G. Toussaint. The relative neighborhood graph of finite planar set. *Pattern Recognition*, 12(4), 1980.

[80] V. Tsaoussidis, H. Badr, X. Ge, et al. Energy/throughput tradeoffs of TCP error control strategies. In *Proc. of IEEE ISCC 2000*, 2000.

[81] V. Tsaoussidis, H. Badr, et R. Verma. Wave and wait: An energy-saving transport protocol for mobile IP-devices. In *Proc. of IEEE ICNP*, 1999.

[82] K.C. Almeroth V. Thanedar, et E.M. Belding-Royer. A lightweight content replication scheme for mobile ad hoc pull-based information dissemination environments. In *Proc. of IFIP Networking*, 2004.

[83] R.D.J. van Nee, G.A. Awater, Masahiro Morikura, et al. New high-rate wireless lan standards. *IEEE Communications Magazine*, 37(12), 1999.

[84] W3C. SOAP, version 1.2, part 1: Messaging framework. *Technical report*, http://w3.org, June 2003.

[85] W3C. Web service description language (WSDLl) version 2.0. *Technical report*, http://w3.org, May 2005.

[86] P.J. Wan, G. Gaminescu, et X.Y. Li. Minimum energy broadcast routing in static wireless networks. In *Proc. of IEEE INFOCOM*, pages 1162-1171, 2001.

[87] A. Wang, W. Heinzelman, A. Sinha, et al. Energy-scalable protocols for battery-operated microsensor networks. *Journal of VLSI Signal Processing*, 29, 2001.

[88] W.G. Wang, T. Hara, M. Tsukamoto, et al. AODV compatible routing with extensive use of cache information in ad-hoc networks. In *Proc. of the ACM SAC*, 2002.

[89] J.E. Wieselthier, Gam D. Nguyen, et al. Ephremides. Energy-efficient broadcast and multicast trees in wireless networks. *Mobile Network Applications*, 7(6), 2002.

[90] P. Wyckoff et S. McLaughry. T spaces. *IBM Systems Journal*, 37(3), 1998.

[91] W. Ye, J. Heidemann, et D. Estrin. An energy-efficient mac protocol for wireless sensor networks. In *Proc. of IEEE INFOCOM*, 2002.

[92] N. Zhou, H. Wu, et A. Abouzeid. Reactive routing overhead in networks with unreliable nodes. In *Proc. of ACM SIGMOBILE*, 2003.

[93] M. Zorzi et R.R. Rao. Energy constrained error control for wireless channels. *IEEE personal Communication Magazine 4*, 4(6), 1997.

[94] M. Zorzi et R.R. Rao. Is TCP energy efficient?. In *Proc.of IEEE MOMUC*, 1999.

[95] M. Zorzi et R.R. Rao. Throughput and energy performance of TCP on a wideband CDMA air interface. *Wireless Communications and mobile Computing*, 2(1), 2002.

[96] J. Liu, F. Sailhan, D. Sacchetti, et V. Issarny. Group Management for Mobile Ad hoc Networks: Design, Implementation and Experiment. In *Proc. of IEEE MDM,* 2005.

[97] F. Sailhan et V. Issarny. Scalable Service Discovery for MANET. In *Proc. of ACM PERCOM,* 2005.

[98] V. Issarny, D. Sacchetti, F. Tartanoglu, F. Sailhan, R. Chibout, N. Levy, et A. Talamona. Developing Ambient Intelligence Systems: A Solution based on Web Services**.** In *Journal of Automated Software Engineering.* Vol 12. 2005.

[99] F. Sailhan et V. Issarny. Energy-aware Web Caching over Hybrid Networks. In *Handbook of Mobile Computing*. CRC Press, 2004.

[100] V. Issarny, F. Tartanoglu, J. Liu, et F. Sailhan. Software Architecture for Mobile Distributed Computing. In *Proc. of the 4th IEEE/IFIP WICSA,* 2004.

[101] F. Sailhan et V. Issarny. Energy-aware Web Caching for Mobile Terminals. In *Proc. of the IEEE ICDCS Workshop on Web Caching Systems,* 2002.

[102] Y. Bromberg et V. Issarny. Service Discovery Protocols Interoperability in the Mobile Environment. In *Proc. of the SEM,* 2004.

[103] A. Wasfi. Collecting User Access Patterns for Building User Profiles Collaborative Filtering. In *Proc. of ACM IUI, 1999.*

[104] N. Neigus. File Tranfer Protocol, Serveur FTP. *IETF RFC 0542,* http://ietf.org, August 1973.

[105] F. Perich, S. Avancha, D. Chakraborty, et al.. Profile Driven Data Management for Pervasive Environments. In *Proc. of ACM DEXA*, 2002.

[106] Y. Aumann, et al. Predicting event sequences: Data mining for prefetching web-pages. In *Proc. of ACM KDD*, 1998.

[107] Jinag, Z. et L. Kleinrock. Web prefetching in a mobile environment. *IEEE Personal Communications*, 5(8), 1998.

[108] S. Vanderwiel et D.J. Lilja. A Survey of Data Pretching Techniques. *Technical Report*, University of Minnesota, 1996.

[109] R. Droms. Dynamic host configuration protocol. *IETF RFC 2131,* http://ietf.org, March 1997.

[110] N. Georgantas, S. Ben-Mokhtar, Y.D. Bromberg, et al. The Amigo Service Architecture for the open networked home Environment. Soumis à IEEE/IFIP WICSA, 2005.