UNIVERSITE LIBANAISE
(Faculté de Génie)

UNIVERSITE SAINT-JOSEPH
(Faculté d'Ingénierie)

Sous l'égide de l'Agence Universitaire de la Francophonie
AUF

Diplôme d'Etudes Approfondies
Réseaux de télécommunications

Contrôle des réseaux Ad-hoc

Par

Rizkallah Karim

Encadré par : **M. Guy Pujolle**
Sidi-Mohammed Senouci
Yacine Ghamri Doudane

Soutenance le 17 décembre 2001 devant le jury composé de

MM.	Samir Tohmé	Président
	Wajdi Najem	Membre
	Imad Mougharbel	Membre
	Mahmoud Doughan	Membre
	Maroun Chamoun	Membre
	Nicolas Rouhana	Membre

Remerciements

Je voudrais exprimer ma gratitude au Pr. Guy Pujolle, de m'avoir permis d'effectuer ce stage au sein de son équipe au LIP6.

Un grand remercient à mes encadreurs, Sidi-Mohammed Senouci et Yacine Ghamri Doudane, pour leurs conseils, soutiens, disponibilités et patiences durant toute la durée de ce stage.

Je tiens aussi à remercier toute l'équipe Réseau et Performance (RP) du LIP6 pour leurs disponibilités à répondre à mes questions….

Résumé

Les réseaux informatiques basés sur la communication sans fil peuvent être classés en deux catégories : les réseaux avec infrastructure et les réseaux sans infrastructure préexistante appelés aussi réseaux ad-hoc. La deuxième catégorie essaie d'étendre la notion de mobilité à tous les composants de l'environnement (exp. téléphone mobile, PC portable, sourie…). Ces composants se déplacent librement et aucune administration centralisée n'est disponible.

Deux exemples de standards, Bluetooth et 802.11, seront traitées à titre d'exemple afin de mieux comprendre le fonctionnement des réseaux ad-hoc. Bluetooth permet la mise en place d'un réseau mobile sans infrastructure, alors que IEEE 802.11 offre la possibilité d'avoir un réseau mobile avec ou sans infrastructure. Les débits offerts par les deux standards diffèrent, en effet la version 1.0 de Bluetooth offre un débit total de 1Mbit/s alors que la version actuelle de IEEE 802.11 (802.11b) offre un débit total de 11Mbit/s tandis que la future version (IEEE 802.11a) promet des débits largement supérieurs. Malgré que dans la nouvelle version "2.0" de bluetooth le débit soit meilleur celui-ci reste toujours faible devant le débit proposé dans la nouvelle version de l'IEEE 802.11. Cependant le domaine d'utilisation diffère puisque bluetooth à été conçu dans le but d'interconnecter les différents équipements d'un même utilisateur alors que IEEE 802.11 à été conçu pour permettre à différents utilisateurs de communiquer comme dans un réseau local fixe (Ethernet).

Les standards de ces réseaux mobiles ad-hoc fournissent actuellement une sécurité de base optionnelle. En effet ces standards prévoient que les stations peuvent utiliser ou non la sécurité. D'autre part, la sécurité n'est assurée qu'entre les stations et pas de bout en bout. De plus les protocoles qui sont utilisés pour assurer la sécurité, comme WEP (Wired Equivalent Privacy) pour 802.11, présentent quelques failles. Par expemple la clé de chiffrage WEP peut être facilement déduite à cause des faiblesses de l'algorithme RC4 [6]. Dans Bluetooth, La sécurité repose sur une clé de chiffrage secrète partagée par les participants d'où la faiblesse, car si quelqu'un connaît votre clé secrète il pourra se faire passer pour vous, écouter votre conversation…

D'un autre coté, les applications sont entrain d'intégrer des fonctions multimédia (vidéo conférence, visiophonie,…) qu'on veut absolument fournir aux utilisateurs des réseaux sans fil. Or ces applications consomment énormément de ressources et ne permettent pas une utilisation efficace et équitable du support de communication surtout quand ils coexistent avec des services de données caractérisés par des transferts en rafales (Burst). La nécessité de fournir un mécanisme capable de partager équitablement les ressources, de garantir une qualité de service (QoS) et de fournir une sécurité d'utilisation est indiscutable pour le développement de ce type de réseau.

Dans les réseaux fixes, la tendance actuelle est d'utiliser un serveur de politiques PDP (Policy Decision Point) pour assurer ces deux besoins (sécurité et QoS). Ce serveur à pour rôle de prendre les bonnes décisions, les transmettre à ses clients appelés PEP (Policy Enforcement Point) en utilisant le protocole COPS (Commnon Open Policy Service) qui est un protocole de signalisation, puis les appliques sur ces PEPs.

L'utilisation d'un tel serveur de politique, permettant d'assurer une gestion par politique de la sécurité de bout en bout et de la qualité de service, nous semble être comme une idée très intéressante pour des réseaux ad-hoc. Il nous a fallut donc adapter ce modèle de politiques aux réseaux ad-hoc car contrairement aux réseaux fixes, qui eux reposent sur une infrastructure et qui contiennent des serveurs dédiés, les réseaux ad-hoc ne reposent sur aucune infrastructure et ne sont composés que de nœuds mobiles. Il paraît alors évident que le serveur de politiques PDP devra être supporté par l'un de ses nœuds mobiles

Notre proposition consiste donc à choisir un de ces terminaux afin qu'il puisse jouer le rôle d'un serveur de politiques PDP. Or vu les ressources limitées offertes par les équipements mobiles, il est claire que certains de ces équipements ne pourront pas supporter la charge d'un serveur de politique sans dégrader les performances de la machine, pénalisant ainsi l'utilisateur qui lui se connecte pour un but: communiquer et échanger des informations, et non pour gérer le réseau. Pour cela on propose un mécanisme d'élection de PDP parmi tous les terminaux du réseau.

Ce mécanisme repose sur le calcul d'un **poids** calculé localement, par chaque terminal, en fonction de quelques critères qui ont pour rôle d'éliminer de l'élection tout terminal ne présentant pas les ressources nécessaires pour supporter le serveur. Une fois le poids calculé, le terminal va essayer de se connecter au réseau. Il va donc essayer de détecter si un serveur est déjà présent. Si c'est le cas, il va se connecter à ce serveur en temps que client et lui fait parvenir son poids. Dans le cas où aucun serveur n'est présent, il se réfère à son poids pour voire s'il à les capacités requises pour supporter le serveur. Deux cas se présentent : capable de le supporter ou non capable. S'il est incapable de supporter le serveur, il va attendre qu'un serveur se déclare et se connectera en temps que client. Si le terminal est capable alors il se déclare comme serveur de politique.

Afin de rendre notre proposition plus robuste, nous avons traité le cas de la contention entre deux machines pour le rôle de PDP. Quand un nœud ne détecte pas la présence d'un serveur et présente les ressources nécessaires pour devenir serveur il démarre la procédure de serveur. Durant la phase de démarrage on propose que le nœud informe, tout autre nœud qui essaye de détecter la présence d'un serveur, qu'il est entrain de démarrer le serveur. Tout terminal qui reçoit cette information attendra la mise en route de ce serveur pour se connecter en temps que client.

Notre proposition traitera aussi les différents scénarios qui peuvent se produire dans le réseau ad-hoc :

- Le terminal PDP veut quitter le réseau.
- Le terminal PDP quitte le réseau de façon brusque suite à un dysfonctionnement.
- Le transfert du serveur vers un terminal qui présente de meilleures capacités que le terminal supportant le PDP actuellement.

Un client du protocole COPS qui est COPS-SLS, semble mieux convenir à notre approche. Cependant certaines modifications ont du être faite pour adapter ce protocole aux réseaux ad-hoc, comme pour le transfert du poids et pour le transfert de la décision de changement de serveur. L'avantage que ce client offre par rapport aux autres clients COPS est le fait que le PEP soit dans le terminal, permettant ainsi d'offrir une négociation interactive entre le terminal et le PDP ainsi que la possibilité de modifier les paramètres négociés préalablement.

Notre proposition, traite la partie qui permet d'introduire un serveur de politiques dans un réseau ad-hoc afin de mettre en place une infrastructure de contrôle. Cette solution est réellement compatible avec ce qui se fait dans le domaine du contrôle de réseaux et c'est une solution robuste vu que tous les cas possibles ont été traite, mais une validation par simulation reste une des perspectives de ce travail.

Un certain nombre de points, comme la sécurisation du PDP par rapport à l'utilisateur et la localisation du *Policy Repository,* nécessitent d'être étudié afin de pouvoir compléter cette étude. D'autre part, il faudrait aussi mettre en place des politiques propres aux réseaux ad-hoc pour le contrôle et la gestion de la sécurité et la qualité de service (QoS).

Introduction

Les progrès réalisés dans le domaine des technologies sans fils ont ouvert de nouveaux horizons dans le domaine des télécommunications. Les réseaux ad-hoc connaissent, actuellement, de plus en plus de succès du à la facilité de leur déploiement et aux économies qu'ils permettent d'effectuer. Actuellement, un certain nombre d'industriels (Ericsson, IBM, Intel...) et d'organisations (IEEE, ITU, IETF) planchent sur le sujet afin d'établir de nouveaux standards. Nous présenterons dans ce rapport deux exemples de standards, Bluetooth et IEEE 802.11, qui permettent de supporter ce genre de réseaux.

Malgré tous les avantages que peuvent avoir les réseaux ad-hoc, ils présentent quelques failles qui sont liés à la faiblesse de la sécurisation des communications (exp. WEP dans IEEE 802.11) ainsi qu'au manque de garantie de qualité de service (QoS).

La tendance actuelle pour régler ces problèmes dans les réseaux fixe, est réalisée par l'intermédiaire d'un modèle par politiques. Ce modèle utilise un serveur de politiques PDP (Policy Decision Point) qui prend des décisions de politiques, et les communique à ses clients appelés PEP (Policy Enforcement Point) qui vont les appliquer. Le protocole utilisé pour communiquer des messages entre le PDP et le PEP, est le protocole COPS (Common Open Policy Service) quoi est un flexible vu qu'il à été conçu pour supporter plusieurs type de client (COPS-RSVP, COPS-PR, COPS-SLS).

Une idée, que nous proposons dans ce rapport est d'adopter ce modèle par politique dans les réseaux ad-hoc afin de régler ces problèmes de sécurité et de QoS. Cette adaptation est nécessaire vu que ce genre de réseau ne repose sur aucune infrastructure et n'est composé que d'équipements mobiles. Notre étude va se concentrer sur le choix d'un des terminaux mobiles pour qu'il puisse jouer le rôle d'un serveur de politiques, suivant un certain nombre de critères. A ces critères sera associé un poids, qui permettra de départager les équipements en deux groupes: ceux qui peuvent supporter un serveur et ceux qui ne peuvent pas supporter un serveur. De plus, ce poids permettra de sélectionner le meilleur des terminaux pour lui assigner la tâche de serveurs.

Un type de client COPS, COPS-SLS, semble convenir aux réseaux ad-hoc. Ce client considère que le PEP peut être supporté par le terminal, ce qui fait que le terminal peut avoir une négociation interactive avec le PDP. COPS-SLS permet aussi aux PEP de modifier les paramètres négociés préalablement. Une attention particulière sera consacrée a ce client car il semble convenir à notre approche.

La suite du rapport est organisée de la façon suivante. La première partie est consacrée à l'étude des réseaux mobiles et plus spécialement les réseaux ad-hoc. Les deux Standards, IEEE 802.11 et Bluetooth, qui supportent les réseaux ad-hoc, y seront également étudiés. Dans la deuxièmes partie on étudiera le protocole COPS et ses différents types de clients dont COPS-SLS. Par la suite, la troisième partie sera consacrée à l'étude de l'adaptation du modèle par politique au réseau ad-hoc ainsi qu'aux extensions du client COPS-SLS proposé. Finalement on présentera les conclusions et les perspectives de notre travail.

PARTIE I

Les réseaux Mobiles

1 Les Réseaux Mobiles

1.1 Introduction

L'essor des technologies sans fil offre de nouvelles perspectives dans le domaine des télécommunications. De très nombreux systèmes utilisent déjà ces techniques et connaissent une très forte expansion à l'heure actuelle (notamment la radio téléphonie mobile) mais requièrent une importante infrastructure logistique et matérielle fixe.

L'environnement mobile est un système composé de sites mobiles, qui permettent aux utilisateurs d'accéder à l'information indépendamment de leur positions géographiques. Ces environnements offrent une grande flexibilité d'emploi. En effet, ils permettent la mise en place de réseaux à des endroits où le câblage serait trop onéreux à réaliser, voire même impossible.

Les réseaux mobiles ou sans fil, peuvent être répartis en deux classes : les réseaux avec infrastructure et les réseaux sans infrastructure.

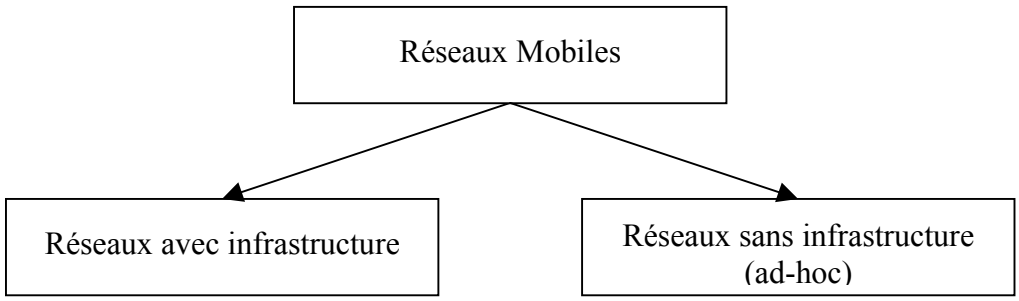

Figure 1.1 : Décomposition des réseaux mobiles

Dans le modèle des réseaux avec infrastructure (cf. fig. 1.2), chaque site mobile est composé d'une station de base (SB) qui est munie d'une interface de communication sans fils. Cette station couvre une zone appelée cellule à partir de laquelle les unités mobiles peuvent communiquer. Les différentes stations de base sont interconnectées par l'intermédiaire d'un réseau de communication filaire.

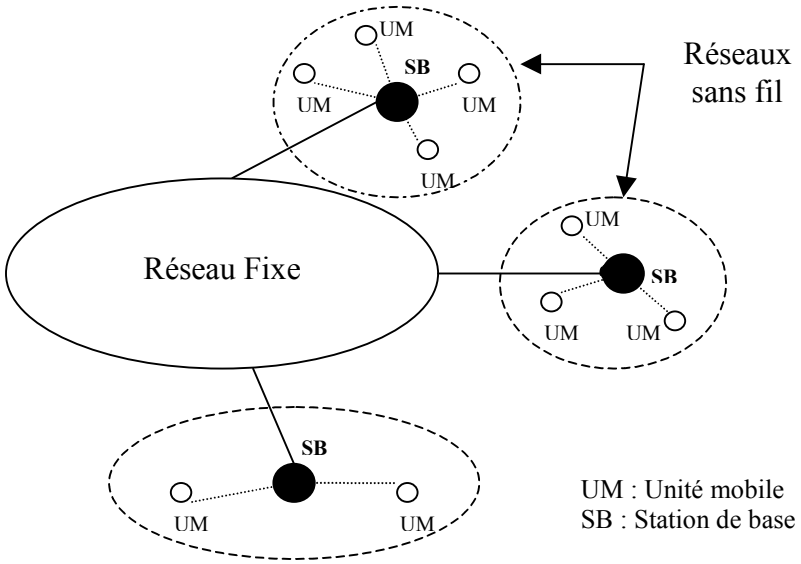

Figure 1.2 : Modèle de réseau avec infrastructure

Dans le modèle de réseau mobile **sans infrastructure préexistante** (cf. fig. 1.3), l'entité "site fixe" n'existe pas. Tous les sites du réseau sont mobiles et communiquent d'une manière directe en utilisant leurs interfaces de communication sans fils. L'absence d'une infrastructure ou du réseau filaire composé de stations de base oblige les unités mobiles à intégrer des fonctionnalités qui n'étaient rempli, jusqu'à présent, que par des unités dédiées (routeurs). En effet ces unités mobiles doivent dans un réseau sans infrastructure se comporter comme des routeurs afin de participer à la découverte et à la maintenance des chemins pour les autres hôtes du réseau.

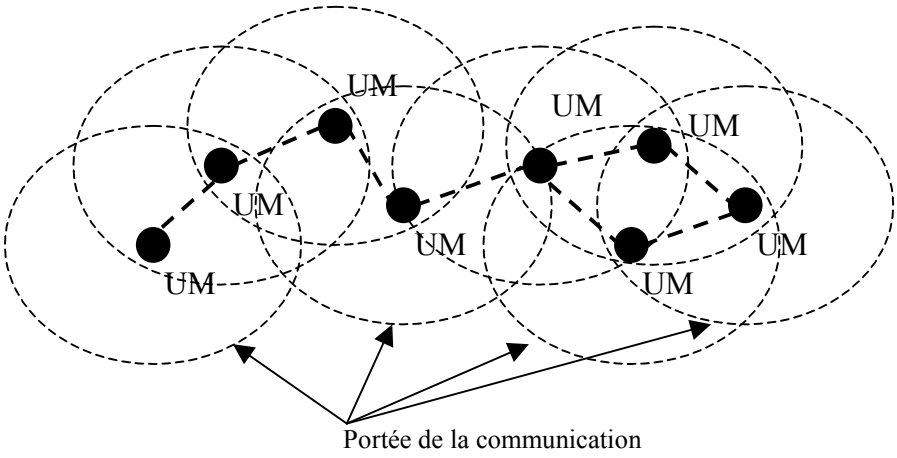

Figure 1.3 : Modèle de réseaux mobiles sans infrastructure.

1.2 Les réseaux mobiles Ad-hoc

L'essor des technologies sans fil offre de nouvelles perspectives dans le domaine des télécommunications. De très nombreux systèmes utilisent déjà ces techniques et connaissent une très forte expansion à l'heure actuelle (notamment la radio téléphonie mobile GSM) mais requièrent une importante infrastructure logistique et matérielle fixe.

Les réseaux mobiles *ad hoc* sont, des réseaux qui s'organisent automatiquement de façon à être déployable rapidement, sans infrastructure fixe, et qui doivent pouvoir s'adapter aux conditions de propagation, aux trafics et aux différents mouvements pouvant intervenir au sein des nœuds mobiles. Contrairement aux réseaux basés sur la communication cellulaire comme le GSM, aucune administration centralisée n'est disponible. Ce sont les hôtes mobiles qui forment, d'une manière *ad ho*c, l'infrastructure du réseau.

Un réseau mobile ad-hoc, généralement appelé MANET (Mobile Ad hoc NETwork), est constitué d'un grand nombre d'unités mobiles qui se déplacent dans un territoire quelconque et dont le seul moyen de communication est l'utilisation des interfaces sans fil, sans l'aide d'une infrastructure préexistante ou administration centralisée.

La topologie du réseau ad-hoc peut changer à tout instant, elle est donc dynamique et imprévisible, ce qui nécessite l'emploi d'un routage interne des messages par les stations intermédiaires. La gestion de ce routage est assez difficile puisqu'elle consiste à établir une sorte d'architecture molle qui doit tenir en compte de la mobilité des stations et de la versatilité du médium physique. Vu la difficulté de la modélisation de cette architecture un groupe de travail, appelé MANET (Mobile ad-hoc NETwork) [17], a été crée dans ce sens à L'IETF. Ce groupe de travail œuvre à la normalisation des protocoles ad-hoc fonctionnant sous IP.

Partant des protocoles de routage de l'IP fixe, les travaux du groupe MANET en ont proposé une extension qui tienne en compte la mobilité des nœuds. Ces protocoles peuvent êtres séparés en deux catégories, les protocoles pro-actifs et les protocoles réactifs. Les protocoles pro-actifs établissent les routes à l'avance en se basant sur l'échange périodique des tables de routage, alors que les protocoles réactifs cherchent les routes à la demande.

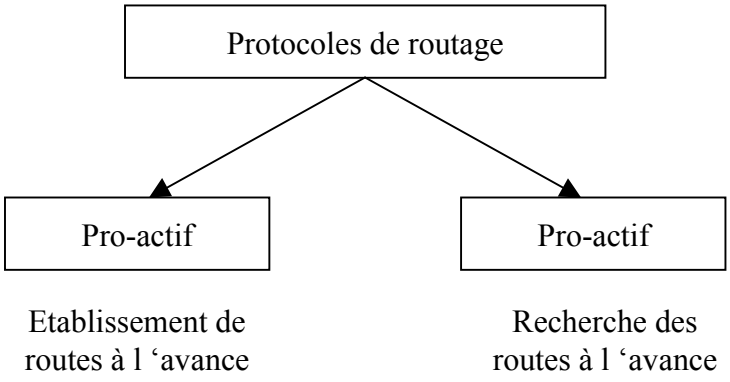

Figure 1.4 : Classification des protocoles

Plusieurs types de protocoles utilisant des techniques de routages différents, comme AODV, DSDV, DSR, OLSR…, sont proposées mais pour l'instant aucun de ses protocoles n'a été normalisé.

Le tableau ci-dessous classe quelques-uns uns des protocoles ad-hoc, d'une part en fonctions de leur appartenance à la famille des protocoles actifs ou pro-actifs, d'autre part en fonction de leurs techniques de routage (Vecteur de distance, Routage à la source, Etat du lien)

Technique de routage utilisée	Réactif	Pro-actif
Vecteur de distance	AODV [18]	DSDV [21]
Routage à la source	DSR [19]	
Etat du lien		OLSR [20]

Tableau 1.1: Protocoles de routage Ah-hoc

1.3 Conclusion

Les réseaux mobiles ad-hoc se distinguent des autres réseaux mobiles par la propriété d'absence d'infrastructures préexistante et de tout genre d 'administration centralisée. Seul les hôtes mobiles seront responsables d'établir et de maintenir la connectivité du réseau d'une manière continue.

Ces réseaux ont la particularité, contrairement aux réseaux avec infrastructure, de s'organiser automatiquement de façon à être déployable rapidement, sans infrastructure fixe, et qui doivent s'adapter aux trafics et aux différents mouvements pouvant intervenir au sein des nœuds mobile.

2 Le standard Bluetooth

2.1 Introduction

2.1.1 Historique

Tout commença en 1994 quand Ericsson Mobile Communication lança une étude pour investiguer sur la faisabilité d'une interface radio à faible consommation et à bas prix, capable d'éliminer les câbles entre téléphones et cartes de PC, écouteurs sans fils,… En février 1998, un groupe spécial d'intérêt « SIG » comprenant Ericsson, Nokia, IBM, Toshiba et Intel se format, et en mai 1998 le consortium **Bluetooth** fut annoncer au public. Ce consortium a pour but d'établir un standard pour l'interface air et un logiciel pour le contrôle de cette interface.[1]

2.1.2 Bluetooth

Bluetooth est le nom donné à une nouvelle technologie qui utilise des liens radio de courte distance. Cette technologie a pour objectif de remplacer les câbles qui, actuellement, relient les différents équipements mobiles et/ou fixes entre eux (cf. fig. 2.1) . Elle permet d'offrir, à tout équipement un accès radio au réseau « LAN », « PSTN », cellulaire et Internet. [2]

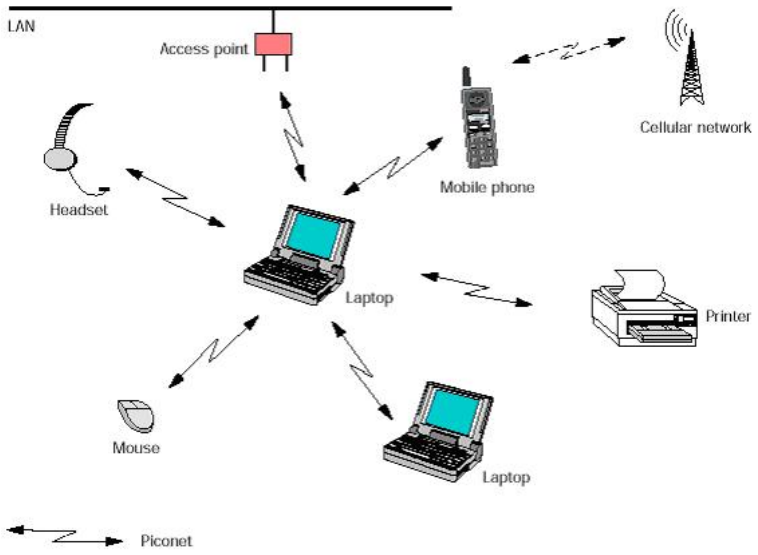

Figure 2.1 : Connectivité via Bluetooth [2]

Bluetooth permet aux différents équipements électroniques de se connecter et de communiquer sans fils. Deux ou plusieurs équipements communiquant ensemble forment un réseau ad hoc de courte distance appelé « piconet » (cf. fig. 2.2 (a) et (b)). Ces piconets s'établissent dynamiquement et automatiquement sans aucune intervention de l'utilisateur. Pour chaque piconet il y a un maître qui organise le trafic. N'importe quel équipement Bluetooth peut devenir maître vu que tous sont identiques. Si plusieurs piconets se recouvrent, alors elles forment une scatternet (cf. fig. 2.2 (c)).

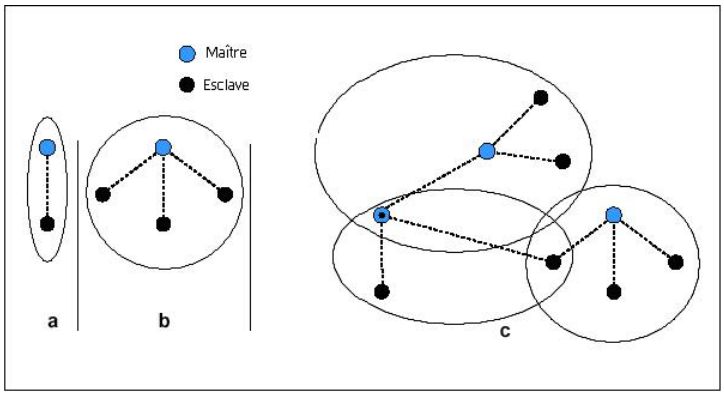

Figure 2.2: Piconet avec un seul esclave (a), à plusieurs esclaves (b), scatternet (c)

2.1.3 La Piconet

La piconet est une cellule composée d'au moins deux unités communicantes entres-elles (cf. fig. 2.2 (a) et (b)). Il peut y avoir, au maximum, sept esclaves actifs en même temps par cellule. Pour chacune des piconets, il y a un seul maître qui gère le trafic et distribue les adresses. Les rôles « maître, esclave » dans la piconet peuvent être inverse à tout moment et par n'importe quelle unité : maître ou esclaves.
La piconet est synchronisée sur l'horloge et l'identité du maître. Les esclaves synchronisent leurs horloges natives (qui n'est jamais ajusté ni éteinte) par le biais d'une compensation (offset). Cette compensation est corrigée par n'importe quel paquet transmis puisqu'il suffit que du code d'accès du paquet pour synchroniser l'esclave. Chaque esclave participant à la piconet se voit attribuer, par le maître, une adresse de membre actif 'AM_Addr'. Seul le maître n'a pas d'adresse de membre actif.
L'esclave ne peut communiquer qu'avec le maître. Il peut émettre un paquet de données dans l'intervalle 'esclave-maitre' que s'il a été adressé dans l'intervalle précédent 'maitre-esclave'.

2.1.4 La Scatternet

Plusieurs piconets couvrant la même région forment une Scatternet. Chaque piconet à son propre maître et sa propre séquence et phase de saut de fréquence qui est déterminée par le maître. Chacune des piconet à un code d'accès diffèrent qui est déterminé par l'adresse de l'équipement (Device Address) du maître. Ce code permet d'éviter les confusions entre les différentes piconets qui couvrent le même endroit. Chaque unité peut participer, en utilisant la bonne identité et la bonne compensation (offset) d'horloge, à plusieurs piconets. Une unité Bluetooth peut être esclave dans plusieurs piconets, mais ne peut être maître que dans une seule piconet (cf. fig. 2.2 (c)). L'unité Bluetooth ne peut communiquer que dans une seule piconet à la fois, car Bluetooth utilise un multiplexage temporel. Pour participer à plusieurs piconets, il existe un mode suspension provisoire qui permet à l'unité de quitter provisoirement une piconet pour visiter une autre.

2.2 Architecture

Dans cette partie nous allons décrire l'architecture de Bluetooth.

2.2.1 La pile de protocole

La pile de protocole Bluetooth présentée dans la figure 2.3 est composée de 7 couches.

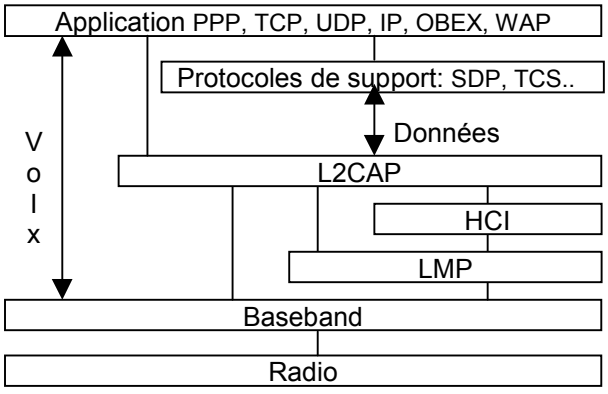

Figure 2.3 : Les couches Bluetooth

Les 3 premières couches (Radio, Baseband, LMP) sont implémentées dans le matériel « hardware/firmware ».

La couche HCI est nécessaire pour interfacer les trois premières couches à la couche L2CAP « Logical Link Control and Adaptation Protocol ». Cependant la couche HCI n'est requise que lorsque L2CAP réside dans le logiciel de l'hôte.

Nous allons dans la suite détailler chacune de ces couches.

2.2.2 La couche Radio

Bluetooth travaille sur la bande de fréquence ISM (Industrial Scientific Medical) qui varie autour de 2.4Ghz et qui n'a pas besoin de licence. Pour la majorité des pays (US et Europe) la largeur de la bande varie de 2400 à 2483.5Mhz, cependant certains pays comme la France, l'Espagne et le Japon n'utilisent qu'une partie de cette bande due aux limitations nationale.

La technique de transmission utilisée par Bluetooth est l'étalement de spectre (Spread Spectrum) à saut de fréquence (Frequency Hopping). Afin d'éviter les interférences qui peuvent être causé par les réseaux Bluetooth voisins (piconets) on utilise un saut de fréquence rapide et des paquets de données plus petits. Le saut de fréquence est fixé a 2402 + k Mhz (k=0,1,2,...,78), il y a donc 79 porteuses (cf. tab. 2.1) pour les pays utilisant toute la bande sinon le nombre est limité à 23 porteuses[1]. La cadence dans les deux cas est de 1600 sauts par seconde ce qui donne des intervalles de 625 µs. Seul un paquet peut être transmis par intervalle. Chaque paquet est transmis avec une fréquence différente.(cf. fig. 2.4)

Pays	**Bande**	**Canal Fréquence Radio**
USA, Europe (la plupart)	2400 - 2483.5 Mhz	f= 2402+k Mhz, k=0,1,2,...,78
France	2446.5 – 2483.5 Mhz	f= 2402+k Mhz, k=0,1,2,...,22

Tableau 2.1: Bande de fréquence [3]

L'espacement entre deux porteuses est de 1 Mhz. et la modulation utilisée est G-FSK (Gaussian-shaped Frequency Shift Keying) ce qui permet d'avoir un taux de 1Mb/s.

2.2.3 La couche BaseBand

Le BaseBand est la couche qui contrôle la couche Radio. Elle s'occupe du maintien de la liaison par l'intermédiaire d'un temporisateur Ce dernier est remis à zéro chaque fois que le maître ou l'esclave reçoit un paquet qui lui est destiné. Sinon à l'expiration du temporisateur la connexion est libérée.. Cette couche est aussi responsable du cryptage pour les liens sécurisés cf. section A ci-dessous, pour plus de détails. La BaseBand définit aussi les différents états (Attente, Connexion) et sous états ("page, page scan, inquiry,...) que peut avoir un terminal Bluetooth pour plus de détails se référer à l'annexe A.2.

1 Il est à noté que les produits qui opèrent sur la bande de fréquence réduite ne seront pas compatibles avec les autres produits qui fonctionnent sur toute la bande car des algorithmes spéciaux pour le saut de fréquence ont du être élaborés.

Bluetooth utilise une combinaison du saut de fréquence et du TDD (Time Division Duplex) pour l'échange des paquets. Le maître et l'esclave communiquent en s'envoyant un ou plusieurs paquets à tour de rôle. C'est toujours le maître qui initie la communication. Le maître utilise les intervalles de temps paires tandis que les intervalles impairs sont réservés aux esclaves (cf. fig. 2.4). Les paquets transmis par le maître peuvent s'étendre sur un, trois voir cinq intervalles de temps.

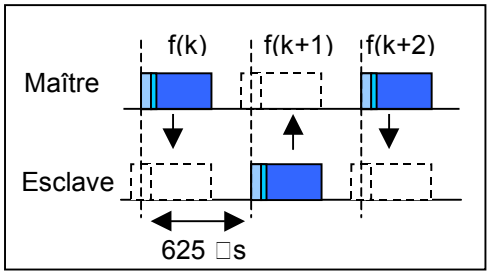

Figure 2.4: TDD avec saut de fréquence [3]

Il existe deux types communications entre le maître et l'esclave: Synchrone (SCO) et Asynchrone (ACL).

- La communication SCO (Synchronous Connection Oriented) est une communication point à point entre le maître et un esclave où le débit est 64 Kbit/s.

- La communication ACL (Asynchronous Connection Less) est une communication point à multipoints entre le maître et tous les esclaves participant à la piconet. où le débit de cette communication peut atteindre 433,9 Kbit/s en bidirectionnel (full-duplex) ou 723.2 Kbit/s et 57.6 Kbit/s en communication déséquilibrée.

Pour ces deux types de communications SCO et ACL, les données sont transmises par l'intermédiaire de paquets. Pour plus de détails sur les types de paquets se référer à l'annexe A.2

A) Sécurité

Bluetooth fournit une sécurité au niveau de la couche Baseband afin de pouvoir assurer une protection et une confidentialité. L'application peut crypter les informations pour plus de sécurité mais la grande partie concernant la sécurité se fait au niveau du lien. [3]

L'authentification et le cryptage sont implémentés de la même façon dans chaque unité BLUETOOTH. Pour le maintien de la sécurité au niveau de la couche liaison, il y a quatre entités présentées dans le tableau 2.2 ci dessous.

Entité	Taille
Adresse publique "BD_Addr"	48 bits
Clé privée pour l'authentification	128 bits
Clé privée pour le cryptage	8-128 bits
RAND	128 bits

Tableau 2.2: Les quatre entités utilisées pour l'authentification et le cryptage [3].

- L'adresse BD_Addr[2] est une adresse publique, elle est unique pour chaque équipement BLUETOOTH.

- Les deux clés sont déduites durant la phase d'initialisation et ne sont jamais divulguées. La clé de cryptage est normalement déduite de la clé d'authentification durant le processus d'authentification.

- Le RAND est un nombre aléatoire dérivé d'un processus aléatoire ou pseudo-aléatoire de l'unité BLUETOOTH et change à chaque nouvelle transaction.

2.2.4 La couche LMP

Les messages LMP (Link Manager Protocol) servent á l'établissement, au contrôle du lien. Ces messages sont transférés dans la charge utile des paquets et sont distingués grâce à une valeur réservée dans le champ L_CH de l'entête de la charge utile. Ces messages sont filtrés par la couche LM (Link Manager) coté récepteur et ne sont pas propagés aux couches supérieures(fig.2.5).

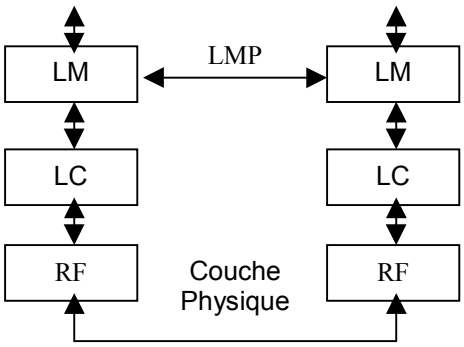

Figure 2.5: Link Manager.

LMP permet d'attacher/détacher les esclaves, inverser les rôles entre le maître et l'esclave, établir des liens ACL et SCO. Il s'occupe aussi des modes *hold* et *park,* présentés dans le paragraphe précédent, qui permettent d'économiser la puissance quand l'unité n'a pas de données à envoyer.

2.2.5 La couche HCI

La couche HCI[3] (Host Controller Interface) fournit une interface de commande au contrôleur Baseband et au gestionnaire de lien (LMP).

Pour certains équipements, comme les PC fixes et portable, le module BLUETOOTH à été ajouté par l'intermédiaire d'une carte PCI ou par un adaptateur USB. Seule les couches radio, baseband et LMP sont implémentées dans ce module. Les données qui doivent être envoyée à la couche LMP ou baseband transitent par la couche de contrôle de transport de l'hôte (Host Control Tansport Layer)[4]. Dans le cas ou on ajoute un module Bluetooth comme le montre la figure 2.6 on a besoin d'un 'driver' (HCI driver) coté hôte et d'une Interface HCI 'firmware' sur le module (BLUETOOTH Hardware) pour que les données puissent être acceptées.

[2] Elle est dérivée du standard IEEE 802
[3] Cette couche n'est requise que lorsque la couche L2CAP réside dans un programme chez l'hôte.
[4] Exemple : bus physique (USB, carte PC, etc…).

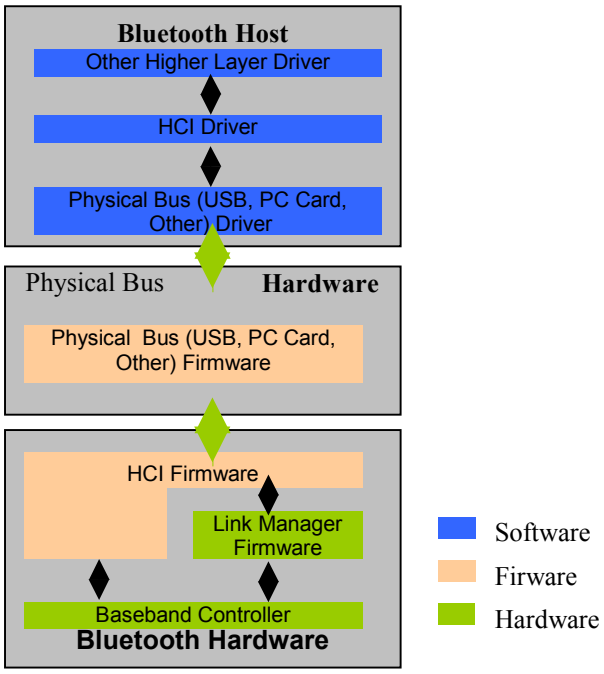

Figure 2.6: Schéma des couches basses nécessaires pour un module Bluetooth ajoutée.

La couche control de transport de l'hôte permet aux deux couches HCI d'échanger des données de façon transparente. Il existe trois types de couche transport : couche de transport HCI USB, couche de transport HCI RS232, couche de transport HCI UART.

2.2.6 La couche L2CAP

La couche L2CAP (Logical Link Control and Adaptation Protocol) se situe au-dessus de la Baseband et réside dans la couche liaison de données Data Link (cf. fig. 2.7). L2CAP est définie seulement pour les liens ACL qui supportent un trafic Best Effort. Certains paquets ne transportent pas des informations L2CAP car ils n'ont pas de CRC.
L2CAP supporte le multiplexage de protocole, la segmentation, le ré-assemblage des informations (SAR), la qualité de service (QoS) ainsi que la gestion de groupe.

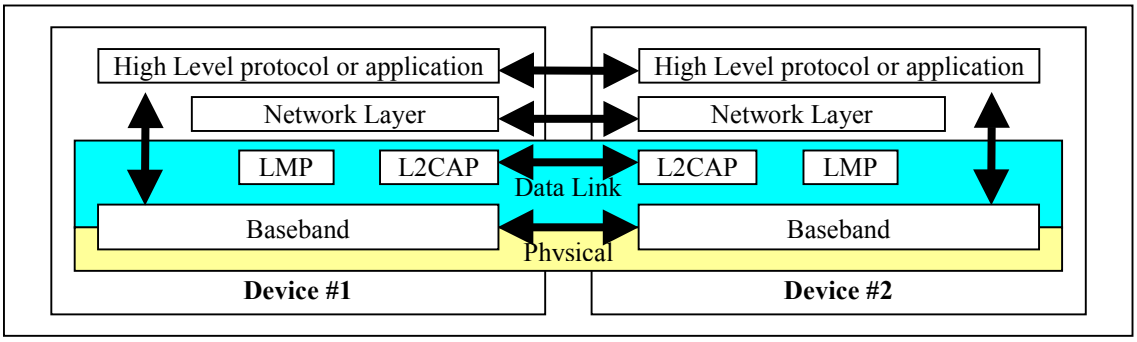

Figure 2.7: L 2CAP avec les couches de protocoles.

L2CAP est située au-dessus de la couche Baseband et s'interface avec les protocoles de communications comme SDP (Service Discovery Protocol), RFCOMM et TCS (Telephony Control Specification). La voix est généralement envoyée sur les liens SCO, cependant des données audios codés (Packetized) comme IPTelephony peuvent être envoyées via L2CAP (cf. fig. 2.8).

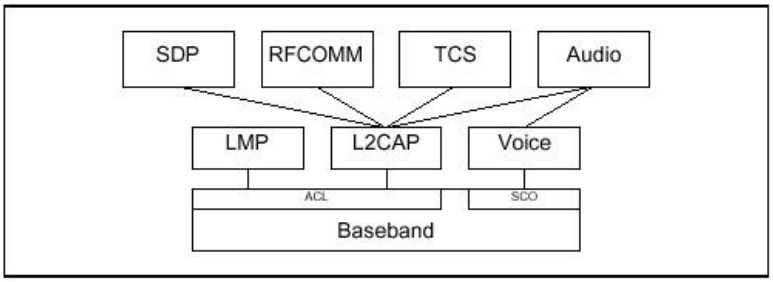

Figure 2.8: L 2CAP dans l'architecture protocolaire de Bluetooth [3].

L2CAP est à base de datagramme, mais suit un mode de communication basée sur les canaux. Le canal représente le flot de données entre deux entités L2CAP. Chaque canal est supposé être full duplex avec une QoS dans chaque direction. Il existe deux types de canaux : orienté connexion (Connection Oriented) et non orienté connexion (ConnextionLess) Il est à noter que L2CAP établie un canal de signalisation séparé pour la requête, la configuration, la déconnexion et le test de la connexion. L2CAP compte sur la Baseband pour assurer la sécurité et l'ordonnancement des données.

2.2.7 Les Protocoles de support

Il existe, comme le montre la figure 2.8, trois types de protocole de support : SDP, RFCOMM et TCS.

A) Le protocole SDP

Le Protocole de découverte de service SDP (Service Discovery Protocol) fournit un moyen, aux applications, de découvrir quels sont les services disponibles. On a deux types de SDP : Serveur et client. Le SDP serveur est utilisé par les unités Bluetooth qui veulent que leurs services soient découverts tandis que le SDP client permet à l'unité de découvrir les services des autres unités. Une unité peut avoir un client pour chaque application, mais elle ne peut avoir plus d'un SDP serveur. Le serveur établit une liste de tous les services que l'unité veut offrir. Le client envoie une requête au serveur qui peut être pour une classe particulière de services ou pour toutes les classes. En fonction de la requête, le serveur répond.

B) Le protocole RFCOMM

RFCOMM est un simple protocole de transport, qui émule le port série RS232 sur des liens sans fils. Ce protocole peut supporter, simultanément, jusqu'à 60 connexions entre deux unités Bluetooth.

C) Le protocole TCS

Le protocole de contrôle et de la téléphonie TCS (Telephony Control Protocol) est basé sur la recommandation Q931 de l'ITU (International Telecom Union) [22]. Il définit le contrôle de la signalisation pour l'établissement d'appels vocaux et de données entre les équipements Bluetooth.

2.2.8 Les Protocoles Adoptés

BLUETOOTH supporte des protocoles existant comme : PPP, TCP, UDP, IP, OBEX, WAP

A) Le protocole OBEX
OBEX (Object Exchange Protocol diminutif de IrBOEX) est un protocole de session développé par l'Association de Données Infra-rouge IrDA (Infrared Data Association) pour l'échange de données objets de façon simple et spontanée. OBEX est implémenté au-dessus de RFCOMM, plus tard au-dessus de TCP/IP.

B) Les protocole PPP
Dans Bluetooth, le protocole PPP se trouve au-dessus de RFCOMM. Il sert à établir des connexions point à point.

C) Les Protocoles TCP/UDP/IP
L'implémentation des ces protocoles dans Bluetooth permet aux équipements BLUETOOTH de communiquer avec n'importe quel autre équipement relié à l'Internet. L'équipement BLUETOOTH, que se soit par exemple un mobile GSM ou un point d'accès, est utilisé comme passerelle pour l'Internet. La pile de protocole TCP/IP/PPP est utilisé pour toute sorte de scénario de passerelles Internet, tandis que la pile UDP/IP/PPP est utilisé pour le transport du WAP.

D) Le protocole WAP
Le Wireless Application Protocol (WAP) est adopté par Bluetooth pour fournir un accès Internet aux équipements Bluetooth via le GSM. Bluetooth fournit la couche physique et la couche de contrôle pour la communication entre le client et le serveur WAP.

2.2.9 La couche Application

C'est la dernière couche, elle sert d'interface entre le service requis par l'utilisateur et les autres couches. Cette couche peut accéder directement à la couche L2CAP ou par certains protocole de support comme RFCOMM, SDP, TCS. Elle peut aussi utiliser d'autre protocole, supportés par Bluetooth, comme TCP/IP, WAP ou autres… L 'application se sert de SDP pour découvrir si le service dont elle a besoin est fourni par l'équipement distant.

Il existe une multitude d'applications possibles [2] voici quelques exemples :

- Trois en un : Un simple appareil téléphonique peut servir comme intercom au bureau, comme téléphone normal s'il y a un point d'accès au PSTN, et comme mobile GSM le reste du temps.
- Un mobile peut se connecter à un ordinateur portable et accéder à ces facilités comme le Email, même si ce dernier est dans sa valise.
- Synchronisation automatique entre le PDA, l'ordinateur portable et le mobile GSM permet une mise à jour et une synchronisation de l'agenda et d'autres données quand ces dernières ont été modifiées.

2.3 Récapitulatif

Bluetooth est un système assez intéressant et utile pour interconnecter les divers types d'équipements (Téléphone fixe, GSM, PC portable, PDA,…) d'un même utilisateurs. Cependant les applications que les équipements vont s'échanger, ne doivent pas exiger trop de bande passante vu que, la bande allouée, pour chaque piconet est de 1 Mbps. La nouvelle génération, version 2.0 de Bluetooth , devrait remédier a ce problème en apportant un débit de 10Mbit/s par liaison.

La sécurité fournit par le standard Bluetooth se situe seulement au niveau des couches basses. En effet c'est la couche *Baseband* qui s'occupe de la mise en place de la sécurité. D'autre part, la sécurité n'est assurée qu'entre le maître et l'esclave et pas de bout en bout. N'importe quel utilisateur qui a accès au réseau aura accès a l 'ensemble du réseau ce qui fragilise se système. D'un autre coté, l'ordonnancement du trafic et du routage dans une scatternet, avec des communications entre les piconets, est un défi et est toujours sujet d'études[1].

Un point positif pour Bluetooth est que l'IEEE à repris la proposition de bluetooth comme norme pour les réseaux personnels ou PAN (Personal Area Network) [23]. D'autre part, de nouveaux sous-groupes de travail de l'IEEE 802.15, prenant comme base la première version de bluetooth, ont été mis en place.

Pour l'instant, du a l'hésitation des industriels, la mise sur le marché de cette technologie reste assez chère par rapport à l'IEEE 802.11 qui est déjà bien implanté. L'IEEE 802.11 sera présenté dans le paragraphe suivant.

3 La norme IEEE 802.11 [5]

3.1 Introduction

C'est en 1990 que le projet de créer un réseau local sans fil appelé WLAN (Wireless Local Aera Network) fut lancé. Le standard a pour but de fournir une connectivité sans fil à des stations fixes ou mobiles qui demandent un déploiement rapide au sein d'une zone locale en utilisant différentes bandes de fréquences. En 2001, l'IEEE (Institte of Electrical and Electronics Engineers) publia le premier standard international pour les réseaux sans fil : l'IEEE 802.11.

Comme Bluetooth, l'IEEE 802.11 utilise des fréquences qui se situent aux alentours de 2.4 GHz. Cette bande appartient à l'ISM (Industrial, Scientific and Medical), Elle ne demande pas de licence pour être exploité, cependant dans certains pays elle n'est pas totalement libre, quoique le processus de libérations soit en cours.

Dans ce type de réseau local sans fil, les communications peuvent se faire de deux manières : soit directement, de terminal à terminal, mais sans possibilité de relayer les paquets vers un autre terminal, soit en passant par une station de base. Les débits varient suivant la technique de codage utilisée et la bande spectrale du réseau, exp. 1Mbit/s avec le BPSK, 2Mbit/s avec le QPSK.

La technique d'accès utilisée pour accéder au support physique est un protocole MAC (Medium Access Control) qui a été spécialement conçu pour l'IEEE 802.11.

Ce support physique a la particularité de s'adapter à tous les supports physiques des Ethernet Hertziens. De nombreuses options sur l'interface radio sont disponibles rendant la mise en œuvre de cette technique d'accès, au canal radio, assez complexe. Cette technique d'accès provient du CSMA/CD (Carrier Sense Multiple Access/ Collision Detection) qui est utilisé dans les réseaux Ethernet pour l'accès au support physique. Cependant, comme la détection de collision dans l'environnement hertzien n'est pas possible, on utilise un algorithme CSMA/CA (Carrier Sense Multiple Access/ Collision Avoidance) qui permet d'éviter ces collisions.(cf. annexe B)

3.2 Architecture de l'IEEE 802.11

L'architecture du réseau 802.11 est cellulaire. Un groupe de terminaux, un ou plusieurs, qui s'associent pour établir des communications directes forme une BSS (Basic Set Service) à ne pas confondre avec le BSS (Base Station Subsystem) du réseau GSM. La zone occupée par les terminaux d'un BSS peut être une BSA (Basic Set Area) ou une cellule.

Comme le montre la figure 3.1, le standard 802.11 offre deux modes de fonctionnement : le mode infrastructure et le mode ad-hoc.

Le mode infrastructure est utilisé pour fournir aux différentes stations des services spécifiques sur une zone de couverture déterminée par la taille du réseau. Les réseaux d'infrastructure sont établis en utilisant des points d'accès: AP (Access Point) qui jouent le rôle de station de base pour une BSS. Lorsqu'un réseau est composé de plusieurs BSS, chacune d'elle est reliée à un système de distribution : DS (Distribution System) par l'intermédiaire de leur point d'accès. Cet ensemble de BSS interconnecté par un système de distribution forme, comme le montre la figure 3.1 une ESS (Extended Set Service).
Le système de distribution correspond, en général, à un réseau Ethernet classique utilisant du câblage métallique. Ce réseau est responsable du transfert des paquets entre les différentes BSS d'une même ESS. La DS est implémenté de façon indépendante de la structure hertzienne, c'est la raison pour laquelle le système de distribution peut correspondre à un réseau Ethernet classique, mais aussi a un réseau Token Ring, FDDI (Fiber Distributed Data Interface) voir même un autre réseau 802.11.
L'ESS peut fournir aux différentes stations mobiles une passerelle d'accès vers un réseau fixe, tel que l'Internet. Cette passerelle permet de connecter le réseau 802.11 à un autre réseau. Si ce réseau est du type 802.x, la passerelle incorporera des fonctions similaires à celle d'un pont.

En mode ad-hoc, le réseau est composé d'un groupe de terminaux qui communiquent sans l'aide d'une quelconque infrastructure tel un point d'accès ou une connexion à un système de distribution. Cet ensemble de terminaux forment une IBSS (Independent Basic Set Service). Dans ce mode chaque station peut établir une communication directe avec n'importe quelle autre station dans l'IBSS. Comme il n'y a pas de point d'accès dans ce mode, les stations n'intègrent qu'un certain nombre de fonctionnalités, telles les trames utilisées pour la synchronisation. Ce mode se révèle très utile pour mettre facilement en place un réseau sans fil lorsqu'une infrastructure fixe ou sans fil fait défaut.

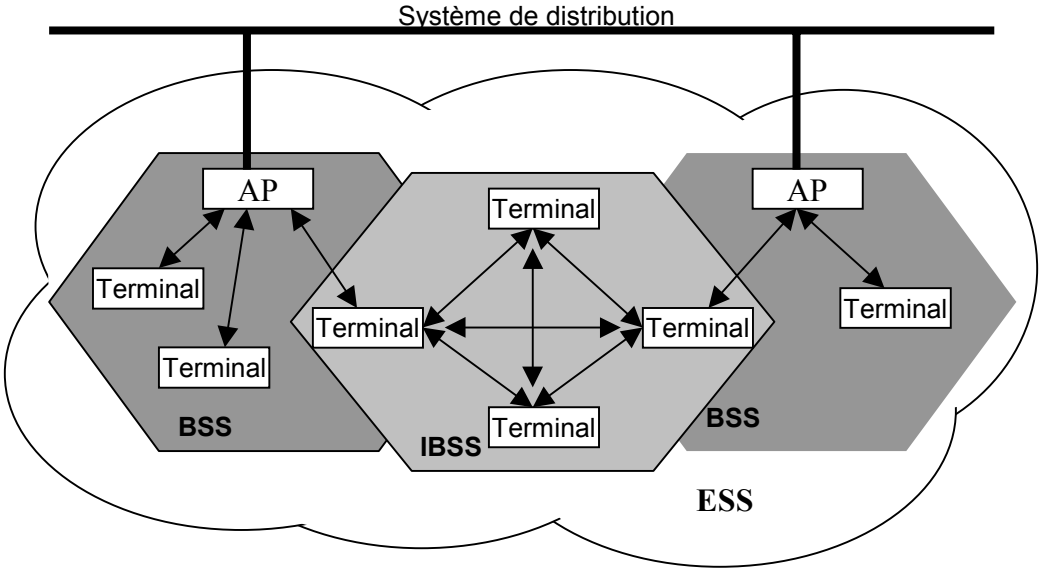

Figure 3.1: Architecture d'un réseau 802.11

3.3 Les couches de L'IEEE 802.11

Le standard IEEE 802.11 couvre les deux premières couches du modèle OSI (Open System Interconnection). Une des caractéristiques essentielles de ce standard est qu'il définit une couche MAC commune à toutes les couches physiques. Donc, de futures couches peuvent êtres ajoutés sans avoir besoin de modifier la couche MAC.

Couche Liaison de données	LLC 802.2			
	MAC 802.11			
Couche Physique	FHSS	DSSS	IR	….

Figure 3.2: Les deux couches de l'IEEE 802.11

3.3.1 La couche physique

Le rôle de cette couche est de transporter les signaux 0 et 1 d'un émetteur à un récepteur. Cette couche est divisée, comme le montre la figure 3.3, en deux sous-couches : PLCP et PMD. La sous-couche PMD s'occupe de l'encodages des données, alors que la PLCP s'occupe de l'écoute du support et fournit un CCA (Clear Channel Assessment) qui est utilisé par la couche MAC pour savoir si le support est occupé ou non.

Couche Physique	Physical Layer Convergence Protocol (PLCP)
	Physical Medium Dependent (PMD)

Figure 3.3 : Les sous-couches de la couche Physique 802.11

L'IEEE 802.11 définit trois types de couches physiques différentes: le FHSS (Frequency Hopping Spread Spectrum), le DSSS (Direct Sequence Spread Spectrum) et l'IR (InfraRouge).

Le FHSS et le DSSS utilisent la bande des 2.4 GHz de l'ISM qui est disponible aux Etats-Unis et en partie en Europe et au Japon. L'infrarouge (IR)est utilisé dans le cas ou les distances entre les différentes stations sont faibles.

Pour qu'un signal soit reçu correctement, la portée ne doit pas dépasser les 150m dans un environnement fermé avec obstacle, les 600m sans obstacle et les 1.5Km dans un environnement extérieur avec une antenne externe. Cependant, pour avoir une très bonne qualité de réception, la portée maximale ne doit pas dépasser les 50m.

Les trois premières couches, définit par le standard IEEE 802.11, permettent d'atteindre au maximum un débits de 2Mbits/s.

En juillet 1998, une nouvelle couche physique a été ajoutée au modèle existant, suivant l'amendement IEEE 802.11b High Rate. Cette quatrième couche permet d'atteindre des débits de 5.5 et 11 Mbit/s.

Une cinquième couche physique (amendement IEEE 802.11a) a été définie dans la bande des 5.2GHz. Elle permet d'atteindre des débits compris entre 6 et 54 Mbit/s grâce au codage OFDM (Orthogonal Frequency Division Multiplexing). C'est le premier standard qui utilise un codage OFDM pour une communication de type paquet. OFDM était, jusqu'à présent, utilisé pour des systèmes de transmission de données continues tels que DVB (Digital Video Broadcasting) ou DAB (Digital Audio Broadcasting). Cette couche est toujours en phase de développement et ne sera commercialisé qu'en 2002.

3.3.2 La couche Liaison de données

Cette couche est essentiellement composée de deux sous-couches: LLC (Logical Link Control) et MAC. La couche LLC utilise les mêmes propriétés que la couche LLC 802.2, ce qui permet de relier le WLAN a tout autre réseau local appartenant à un standard de l'IEEE. Tandis que le couche MAC est spécifique à l'IEEE 802.11.

Le rôle de la couche MAC 802.11 est assez similaire a celui de la couche MAC 802.3. En effet les terminaux écoutent la porteuse avant d'émettre. Si la porteuse est libre le terminal émet sinon il attend. Cependant, la couche MAC 802.11 intègre un grand nombre de fonctionnalités qui n'existe pas dans la version 802.3.

Le standard IEEE 802.11 définit deux méthodes d'accès au niveau MAC fondamantalement ifférentes :

- **DCF** (Distributed Coordination Function) : la méthode d'accès est similaire au réseau traditionnel supportant le best effort. Elle a été conçue pour prendre en charge le transport de données asynchrones. Cette méthode se base sur le protocole CSMA/CA Carrier Sense Multiple Acces/Collision Avoidance) définit ci dessous.

- **PCF** (Point Coordination Function): la méthode d'accès est fondée sur l'interrogation à tour de rôle des terminaux qui sont contrôlés par le point d 'accès. Cette méthode a été spécialement conçue pour la transmission de données sensibles, qui demandent une gestion du délai utilisée pour des applications de types temps réel comme la voix ou la vidéo.

Un réseau de type ad-hoc utilise uniquement la méthode DCF, alors que le réseau en mode infrastructure utilise les deux méthodes d 'accès: DCF et PCF.

3.4 Fonctionnement d'IEEE 802.11

Comme l'IEEE 802.11 est un réseau local sans fils, le passage d'une cellule à une autre sans interruption (handover) paraît évident or cela ne fut pas le cas dans les premières versions, il a du être introduit dans les nouvelles versions. De même, la sécurité a été renforcée pour éviter qu'un client ne prenne la place d'un autre ou qu'il écoute les communications des autres utilisateurs.

3.4.1 Les handovers

Dans les réseaux sans fil, les handovers surviennent lorsqu'un terminal se déplace d'une cellule à une autre sans qu'il y ait une interruption de la communication. Ces handovers se font, a quelques nuances près, de la même manière que dans la téléphonie mobile. Dans les réseaux sans fil, le handover se fait entre deux transmissions de paquets et non pas au milieu d'un dialogue comme dans la téléphonie mobile. Les déconnexions dans les réseaux sans fil peuvent entraîner des pertes de performance du réseau.

Le standard ne fournit pas un mécanisme de handover mais définit certaines règles, telles la synchronisation, l'écoute passive et active ou des mécanismes d'association et de réassociation qui permettent aux stations de choisir le point d'accès auquel elles veulent se rattacher. Lorsqu'un terminal se déplace (change de cellule) ou qu'il soit en mode d'économie d'énergie, il doit rester synchronisé pour pouvoir communiquer. Dans une BSS, les stations synchronisent leur horloge avec le point d'accès. Pour garder la synchronisation, le point d'accès envoie périodiquement des trames balisées, appelées Beacon Frames, qui contiennent la valeur de l'horloge du point d'accès. Lorsque la station reçoit cette trame elle met à jour son horloge.

Quand un terminal veut accéder après un allumage, retour d'un mode veille ou d'un handover à une BSS ou ESS contrôlé par un ou plusieurs point d'accès, il choisit un point d'accès, en fonction d'un certain nombre de critère comme la puissance du signal, le taux d'erreur des paquets ou la charge du réseau.

3.4.2 Sécurité

La sécurité fournit par l'IEEE 802.11 se base sur le protocole WEP (Wired Equivalent Privacy) qui se base sur l'authentification des stations et le chiffrement des données. L'authentification est le moyen par lequel on vérifie que la station est autorisée à communiquer avec une autre station dans une zone de couverture donnée. Dans le mode infrastructure, l'authentification se fait entre chaque station et le point d'accès.

Le standard fournit un algorithme pour le chiffrement des données pour empêcher toute écoute clandestine sur le support. Chaque station possède une clé secrète de 40 bits partagée. Cette clé est concaténée avec un code de 24 bits, le vecteur d'initialisation ou IV (Initialisation Vector), qui est réinitialisé à chaque transmission. La nouvelle clé de 64 bits est placée dans un générateur de nombre aléatoire appelé PRNG (RS4) qui provient de l'algorithme RSA (Rivest Shamir Adelman). Ce générateur détermine une séquence de clés pseudo-aléatoires, qui permet de chiffrer les données. Une fois chiffrée, la trame peut çtre envoyée avec son IV. Pour le déchiffrage l'IV sert à retrouver la séquence de clés qui permet de déchiffrer les données. Seul les données de la trame MAC sont chiffrer et non l'en-tête de la trame de la couche physique.

Les techniques d'authentification utilisées sont les suivantes : Open System Authentification (OSA) et Shared Kay Authentification (SKA). La méthode OSA est le système d'authentification par défaut. Il n'y a pas d'authentification explicite, n'importe quel terminal peut s'associer avec n'importe quel point d'accès et écouter toutes les données qui sont échangés au sein d'une BSS. Tandis que la méthode SKA fournit un meilleur système d'authentification puisqu'il utilise un mécanisme de clé secrète partagée.

Pour restreindre encore la possibilité d'accéder à un point d'accès, ce dernier peut posséder une liste d'adresse MAC, appelée Access Control List, qui permet de fournir l'accès qu'aux stations dont l'adresse MAC est spécifiée dans la liste.

D'après le standard, le protocole WEP est définit de manière optionnelle : les stations et les AP ne sont pas obligés d'implémenter ce protocole

3.4.3 Economie d'énergie

Les réseaux sans fils, peuvent posséder des terminaux fixes et /ou mobile; or le principal problème de terminaux mobiles c'est l'autonomie. Pour cela le standard a prévu deux modes:

- **Continuous Aware Mode :** Le premier mode correspond au fonctionnement par défaut. La station est tout le temps allumée et écoute continuellement le support.

- **Power Saving Mode :** Le second mode permet d'économiser l'énergie. Dans ce cas là, le point d'accès garde à jour une liste de toutes les stations qui sont en mode d'économie d'énergie et stocke toutes les données qui leur sont adressées. Toutes les stations qui sont en veille s'activent, en même temps, à des périodes de temps régulières pour recevoir une trame particulière, appelée TIM (Traffic Information MAP), envoyée par le point d'accès puis retournent en mode veille. Cette trame informe les terminaux mobiles s'ils ont de données stockées dans le point d'accès. Lorsqu'un terminal s'aperçoit que le point d'accès contient des données qui lui sont destinées, il envoie a ce dernier une requête, appelée Polling Request Frame, pour mettre en place le transfert de ces données. Dès que le transfert se termine, le terminal retourne en mode veille jusqu'à la prochaine trame TIM.

3.5 Récapitulatif

Le standard 802.11, à été conçu pour créer un réseau local sans fil ou WLAN (Wireless Local Aera Network) afin de permettre à différents utilisateurs de communiquer. La mise en place de ce réseau local est rapide et permet d'effectuer des économies exp. financière, d'administration,...

L'IEEE 802.11 supporte le mode infrastructure et le mode sans infrastructure (mode ad-hoc) ce qui lui permet de s'adapter à n'importe quel contexte.

C'est le premier standard qui permet aux réseaux locaux sans fils d'atteindre des débits de 11Mbits/s avec la version 802.11b et bientôt 54 Mbps avec la future version 802.11a.

Cependant la sécurité fournit par IEEE 802.11 reste assez légère. En effet, le protocole WEP chargé de la sécurité dans les réseaux 802.11 présente des failles puisque la sécurité n'est juste assurée qu'entre les stations et pas de bout en bout ce qui fait qu'un utilisateur qui a accès aura accès a l'ensemble du réseau. D'autre part des failles dans le protocole WEP ont été découvertes, comme la vulnérabilité de l'algorithme RC4 [6] et la révélation de la clé WEP associé [7].

4 Conclusion

Un réseau ad-hoc est une collection d'entités mobiles, interconnectées par une technologie sans fil formant un réseau temporaire sans l'aide d'une quelconque infrastructure préexistante ou administration centralisée. Bluetooth et IEEE 802.11 sont deux technologies différentes qui supportent les réseaux ad-hoc.

La différence entre les débits offerts par les deux technologies est flagrante. En effet dans la version actuelle de Bluetooth, version 1.0, le débit total offert ne peut dépasser les 1Mbit/s alors que dans 802.11, la versions actuelle, le débit total avoisine les 11Mbit/s et les nouvelles versions pourraient atteindre des débits de 54Mbit/s. La nouvelle version de Bluetooth devrait y remédier en apportant un débit total de 10Mbits par liaison, mais ceci reste insuffisant comparé aux débits qui seront offert par la nouvelle version de l'IEEE 802.11 (IEEE 802.11a). Cependant, il faudrait voir dans quel contexte on se situe, en effet Bleutooth à été conçu dans le but d'interconnecter des équipements domestiques (téléphone GSM, imprimante,…) et des terminaux de télécommunications (PC fixe et portable, PDA) d'un même utilisateur créant ainsi un réseau personnel sans fil ou WPAN (Personal Aera Network) alors que IEEE 802.11 ne s'intéresse qu'à interconnecter des terminaux (PDA, PC portable et fixes,…) qui désirent échanger des informations au niveau personnel ou professionnel.

Un point positif pour Bluetooth vient, malgré la faiblesse du débit qu'il offre, de la mise en place de nouveaux sous-groupes de travail de l'IEEE 802.15, qui prennent comme base la version 1.0 de bluetooth, pour normaliser l'exploitation de Bluetooth dans différents contextes [5,23].

Partie II

Le protocole COPS

Dans cette partie, nous allons étudier dans un premier temps le protocole COPS qui est un protocole développé par l'IETF (RFC2748), pour assurer des fonctions de contrôles d'admission en les fondants sur des règles. Par la suite, nous allons faire le tour quelques type de clients COPS.

1 Le protocole COPS

Le protocole COPS (Common Open Policy Service) est un simple protocole de requête/réponse qui permet aux serveurs de politiques ou PDP (Policy Decision Point) de communiquer les décisions de politiques aux équipements du réseau ou PEP (Policy Enforcement Point). Deux modèles (principaux) sont supporté par le protocole COPS: modèle *Outsourcing* et *provisioning*. Le but de ce protocole est l'administration, la configuration et l'exécution des politiques.

Le modèle *Outsourcing* est utilisé quand il y a des évènements déclencheurs "*Trigger events*" au niveau du PEP qui nécessitent une décision politique (exemple les requêtes dynamiques pour l'admission de nouveau flux). Le PEP délègues les décisions à un serveur de règles externe « PDP ». Il envoie un message de requête au PDP, et attend la décision de ce dernier avant d'admettre un nouveau flux (cf. fig. 1). Le modèle *Outsourcing* est utilisé pour le type de client RSVP [9].

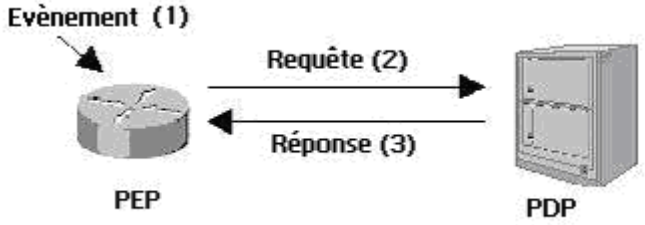

Figure 1: Modèle Outsourcing

En ce qui concerne le modèle de *Provisioning* (configuration) [10] le PDP configure pro-activement le PEP de telle sorte que ce dernier sache comment gérer les mécanismes de la QOS (Quality of Service). Le mécanisme pour l'échange des informations de configuration et pour le stockage de ces informations est basée sur la définition de la « PIB » (Policy Information Base). Dans ce modèle, soit il n'y a pas d'évènements déclencheur *Trigger events* au niveau du PEP (exemple : seulement les paquets de classification, de marquage, de planification… sont effectués par le PEP) soit ces évènements doivent être traités par l'intermédiaire d'informations locale (c.à.d. décrite dans les ressources disponibles approvisionner par le PDP).

Le protocole COPS est un protocole flexible, car il est conçu pour supporter plusieurs type de client de politique. Chaque type de client doit être décrit dans une proposition envoyer à l'IETF « draft ».

Dans les modèles du protocoles COPS, l'échange entre PEP et le PDP se fait par l'intermédiaires de messages.

2 Format des messages COPS

Dans ce paragraphe on va décrire le format de messages et des objets échangés entre un PEP et le PDP distant.

Chaque message COPS est composé d'une entête « Common Header » suivit d'un certain nombre d'objets (cf. fig. 2.1).

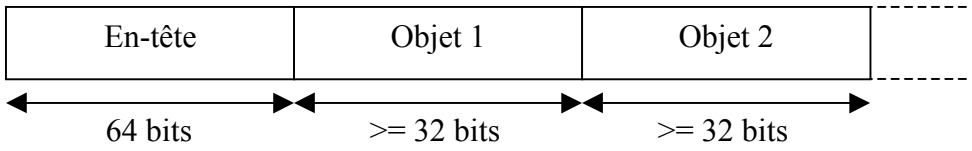

Figure 2.1: Message COPS

2.1 Description de l'Entête du message COPS

Chaque message COPS contient une Entête de 64 bits (8 octets). Cette Entête est composée de cinq champs (cf. fig. 2.2).

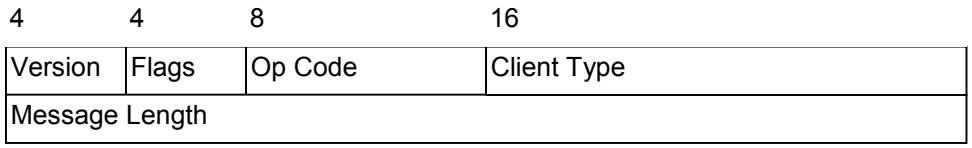

Figure 2.2: Entête du message COPS

Version 4 bits
Ce champ indique le numéro de la version de COPS. Actuellement la version est 1.

Flags 4 bits
C'est la valeur du drapeau. Il indique si le message est sollicité ou non-sollicité.
Quand un message est sollicité par un autre message la valeur du drapeau est « 0001 » sinon elle est « 0000 ». Par exemple, si le PEP envoie une requête au PDP, la décision envoyée par le PDP est un message sollicité donc le *flags* est « 0001 ». Par contre si le PDP envoie une décision pour une mise à jour ou pour effacer une politique, le *flags* du message est « 0000 » car c'est un message non-sollicité.

Op Code 8 bits
C'est le champ Opération. Il identifie les différents types de messages de COPS. Il existe, comme le montre le tableau 2.1, dix types de messages. Pour plus de détails sur les messages se référer au §B de l'annexe C.

Client Type 16 bits
C'est le champ type de client. Il identifie le politique du client à appliquer. Par exemple RSVP est le type de client 1. L'interprétation de tous les objets encapsulés est relative au type-client. Pour les messages KA « Keep-Alive », le champ type-client doit être mis à 0 car ce message est utilisé pour vérifier la connexion et pas pour vérifier la session du client.

Message Length 32 bits

Ce champ indique la taille en octets du message COPS. Il inclue l'Entête standard COPS ainsi que tous les objets encapsulés.

Code	Message	Sigle	Direction
1	Request	REQ	PEP → PDP
2	Decision	DEC	PDP → PEP
3	Report State	RPT	PEP → PDP
4	Delete Request State	DRQ	PEP → PDP
5	Synchronize State Req	SSQ	PDP → PEP
6	Client-Open	OPN	PEP → PDP
7	Client-Accept	CAT	PDP → PEP
8	Client-Close	CC	PDP ←→ PEP
9	Keep-Alive	KA	PDP ←→ PEP
10	Synchronize Complete	SSC	PEP → PDP

Tableau 2.1: Les différents type de messages dans COPS

2.2 Format de l'objet

Tous les objets suivent le même format. Chaque Objets est composé d'une Entête de 32 bits (4 octets) et d'un ou plusieurs mots de 32 bits (4 octets) (cf. fig. 2.3). L'entête de l'objet est composée de trois champs : Length, C-num et C-type.

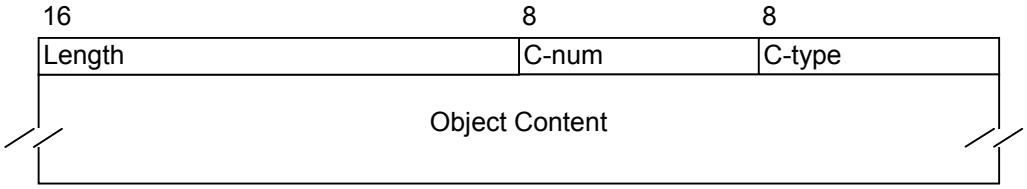

Figure 2.3: Format général de l'objet

1- Length (16 bits)

Il décrit le nombre d'octet (y compris l'entête de l'objet) qui constitue l'objet. Avant d'envoyer l'objet il faut vérifier que la longueur en octet de l'objet forme un multiple de 4 octets sinon il faut ajouter du bourrage à la fin du message afin de constituer un mot de 32 bits (4 octets).

2- C-num (8 bits)

Il identifie la classe de l'information contenue dans l'objet. Il y a 16 classes d'objets COPS (cf. tab.2).

3- C-type (8 bits)

Il identifie le sous-type de la classe d'objet. La combinaison entre C-type et C-num définit un type d'objet unique. Il existe actuellement 29 types d'objets (cf. tab.2).

Le champ *objet content* est le dernier champ de l'objet COPS, c'est le contenu de l'objet.

C-Num	Classe de l'objet	C-Type	Objet
1	Handle	1	Client – Handle
2	Context	1	Context
3	In Interface (IN-Int)	1	IPv4 address + Interface
		2	IPv6 address + Interface
4	Out Interface (OUT-Int)	1	IPv4 address + Interface
		2	IPv6 address + Interface
5	Reason Code (Reason)	1	Reason Code
6	Decision (Decision)	1	Decision Flags
		2	Stateless Data
		3	Replacement Data
		4	Client Specific Decision Data
		5	Named Decision Data
7	LPDP Decision	1	Decision Flags
		2	Stateless Data
		3	Replacement Data
		4	Client Specific Decision Data
		5	Named Decision Data
8	Error (Error)	1	Error
9	Client Specific Info (ClientSI)	1	Signaled ClientSI
		2	Named ClientSI
10	Keep-Alive Timer (KATimer)	1	Keep-alive timer value
11	PEP Identification (PEPID)	1	PEP Identification
12	Report Type (Report-Type)	1	Report-Type
13	PDP Redirect Address (PDPRedirAddr)	1	IP V4 +TCP Port
		2	Ipv6 + TCP port
14	Last PDP Address (LastPDPAddr)	1	IPv4 Address
		2	IPv6 Address
15	Accounting Timer	1	Accounting timer value
16	Message Integrity	1	HMAC digest

Tableau 2.2: Les differents objets COPS

Pour plus de détails sur les différents objets se référer au § A de l'annexe C.

3 La communication dans COPS

Le Protocole COPS est utilise dans un modèle de gestion de réseau à base de politique qui sont échangées entre le serveur de politique PDP et le point d'application de politique PEP.

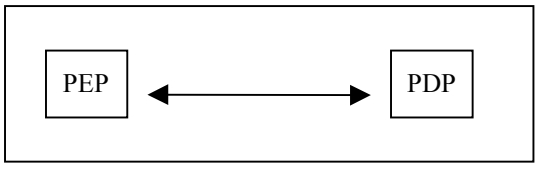

Figure 3.1 : Echange entre le serveur et le client

3.1 Les principaux composant de COPS

Le PDP et le PEP sont les principaux composant du protocole COPS. Le client peut supporter un LPDP (Local PDP) qui est un PDP local. Ce dernier permet au client de prendre des décisions locales, pendant une période prédéterminée, lorsque le PEP ce dé-connecte du PDP.

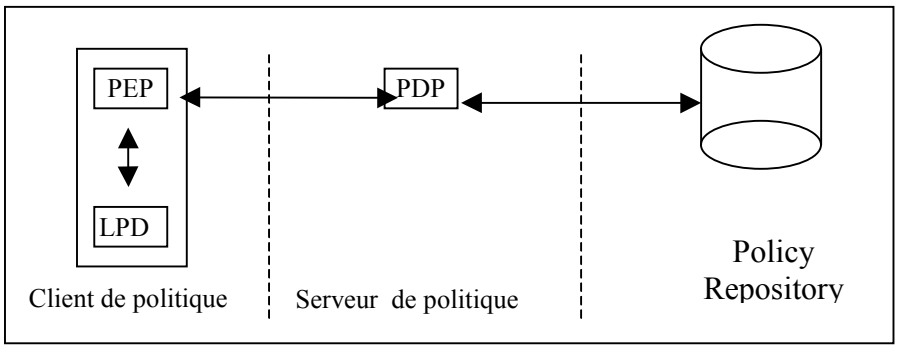

Figure 3.2: Illustration des échanges dans COPS

PDP : Policy Decision Point. C'est le serveur de politique qui contrôle directement le PEP. Il peut supporter un ou plusieurs type de client[5].

PEP : Policy Enforcement Point. C'est n'importe quel équipement du réseau qui reçoit et applique les décisions du PDP.Le PEP est capable de supporter plusieurs type de client

LPDP : Local PDP. Il permet de prendre des décisions politiques locale durant l'absence du PDP. Le LPDP doit par la suite rendre compte au PDP des décisions prises, le PDP les accepter où les refuser. Sa présence est facultative.

Policy Repository: C'est l'endroit où les règles de politiques sont stockées.

3.2 Fonctionnement de COPS

3.2.1 Etablissement de la connexion

Au début, une connexion[6] TCP[7] est établie entre le client PEP et le serveur PDP. Il est à noté que le PDP écoute sur un port TCP connu « 3288 ». C'est sur cette connexion TCP que le PEP va envoyer des requêtes et va recevoir des décisions du PDP

La localisation du PDP peut être soit configurer soit obtenu par l'intermédiaire du protocole de localisation de service 'SLP'(Service Location Protocol) [16].

Une fois la connexion établie, le PEP envoie un message d'ouverture de client (OPN) et précise le type de client (RSVP, Provisioning,..). Si un PEP supporte plusieurs type de client, il peut envoyer un message d'ouverture de client pour chaque type de client sur une ou plusieurs connexion TCP, comme le montre la figure ci-dessous.

Le PDP est capable d'accepter ou de refuser chaque type client indépendamment.

[5] Le support de plusieurs type de client n'est par fournit par le protocole COPS mais plutôt laissé à l'architecture du programme qui supporte le serveur [1]
[6] C'est le PEP qui est responsable de l'initiation de la connexion TCP avec le PDP; cette connexion est persistante .
[7] Le protocole COPS utilise TCP comme protocole de transport afin de fiabilisé l'échange de messages entre les clients de politiques et le serveur.

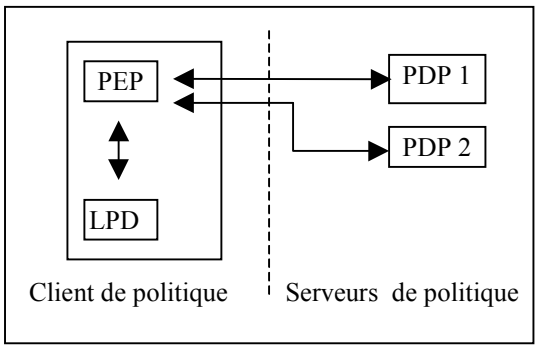

Figure 3.3 : Connexions d'un client avec plusieurs PDPs.(un PDP/type de client)

Si le PDP supporte ce type de client, il répond avec le message d'acceptation de client (CAT), sinon le PDP répond par un message de fermeture de client (CC). Dans le cas où le client serait refusé, le PDP peut éventuellement le diriger vers un autre PDP en lui spécifiant l'adresse IP et le port TCP du nouveau PDP. Parfois ce dernier peut être le même PDP mais avec un numéro de port diffèrent.

Dés quel le PEP reçoit le message d'acceptation de client (CAT), il peut commencer à envoyer des requêtes au serveur et attend la décision du PDP.

Le PEP utilise les messages de rapport (RPT) pour informer le PDP du succès ou de l'échec de l'application des décisions.

Il est à noter, que pour chaque message émis, le type de client est indiqué dans le champ *Client Type* (cf. §2.1) car chaque type de client peut avoir des données différentes qui peuvent nécessiter des décisions différentes.

3.2.2 Identification des requêtes

Chaque requête a un *Handle* unique qui l'identifie. Ce *Handle* est choisit par le PEP et il est opaque au PDP. Les *Handle* utilisés sur une connexion TCP sont uniques.

Le PEP utilise le *Handle* pour mettre à jours ou pour effacer une requête déjà installée et pour envoyer des messages de rapport (RPT) au PDP. De même le PDP l'utilise pour identifier la requête afin de lui envoyer des décisions sollicitées ou non sollicitées.

3.2.3 Modèle Outsourcing

Dans le modèle *OutSourcing,* lorsqu'un événement qui nécessite une nouvelle décision politique arrive au PEP, un message de requête est envoyé par le PEP au PDP. Le PDP prend une décision et la renvoie au PEP dans un message décision[8]. Dans le cas ou la requête est acceptée, le PDP la mémorise.

[8] Les réponses du PDP aux nouvelles requêtes du PEP peuvent être différentes parce qu'il prend en considérations les requêtes et les décisions déjà installées.

Une requête déjà installée peut être mise à jour par le PEP, en envoyant un nouveau message de requête contenant le *Handle* (identificateur) de la requête. Un PDP peut aussi, modifier la décision qu'il a prise pour une requête donnée, en envoyant un nouveau message de décision (qui n'est pas sollicité) qui contient le *Handle* de la requête.

Il est à noter que c'est de la responsabilité du PEP de notifier tout changement d'état dans la requête ou d'effacer la requête lorsqu'elle n'est plus requise. La notification se fait par le biais des messages rapports (RPT).

3.2.4 Modèle de provisionning

Dans ce modèle comme dans le modèle *Outsourcing* le PEP fait une demande de configuration au PDP pour une interface, un module ou une fonctionnalité particulière qui peut être spécifiée dans le *Named ClientSI* (cf. § A.9 de l'annexe C).

Le PDP, en fonction de la demande faite, enverra plusieurs décisions contenant des unités de configurations nommées au PEP qui les installera et les utilisera localement. Une configuration nommée peut être modifiée (ajouter/retirer) par le PDP en envoyant de nouveau messages de décisions.

Le PEP peut notifier le PDP du statut des états installés en utilisant les messages de rapport (RPT). Ce message peuvent être utilisés pour commencer la facturation, notifier les actions prises ou mettre à jours périodiquement de la comptabilité et la surveillance.

3.2.5 Maintient de la liaison

Le protocole COPS utilise des messages *Keep-Alive* (KA) pour vérifier que la connexion PEP-PDP est toujours valable même quand aucun message n'est échangé entre le PEP et le PEP.

Le PEP génère ces messages de façon aléatoire durant le période de temps comprise entre le ¼ et les ¾ du plus petit intervalle de temps KA qui est spécifié dans les messages d'acceptation de client envoyés par le PDP. A la réception de ce message, le PDP répond par un message *Keep-Alive*.

Si un des deux cotés (PDP ou PEP) ne reçoit pas de l'autre un message *Keep-Alive* ou n'importe quel autre message durant le plus petit intervalle de temps KA, la connexion est considérée comme rompu.

3.2.6 Synchronisation

Quand le PEP détecte que la connexion est rompu, il doit essayer de se reconnecter à son PDP principale ou à un PDP alternatif/secondaire. Entre temps il peut prendre des décisions locales grâce au LPDP (PDP locale). Dés le rétablissement de la connexion le PEP doit informer le PDP des tous les évènements qui ont traversé le contrôle d'admission locale durant son absence.

L'interruption du service peut être minimisée si le PEP mémorise[9] « caches » les décisions prisent auparavant et continue à les utilisés pendant un certain temps (défini).

Lorsque le PEP arrive à se reconnecter à un PDP (n'importe lequel) et qu'il mémorise toujours les décisions de l'ancien PDP alors il doit informer le nouveau PDP. Ceci est fais en incluant l'objet *LastPDPAddr* (qui contient le port et l'adresse de l'ancien PDP) dans le message d'ouverture de client (OPN).

Une fois la connexion établie, le PDP peut demander au PEP de synchroniser un/tous les états ou de ne pas synchroniser les états (dans le cas ou c'est le même PDP et que l'interruption de service a été momentanée. Dans ce cas le PDP contient toujours la liste complète des états de ce PEP).

Pour la synchronisation le PDP envoie un message *Synchronize State Request* (SSQ). Lorsque le PEP termine la synchronisation il envoie le message *Synchronize State Complete* (SSC).

3.2.7 Fermeture

Quand un PEP détecte une rupture de la connexion qui est due à la non réception du message *Keep-Alive* dans les délais, il doit envoyer un message de fermeture de client (CC) pour chaque type de client ouvert en spécifiant comme code d'erreur: échec de la communication. Puis il termine la connexion avec le PDP et essayer de se reconnecter à nouveau avec ce PDP ou avec un PDP alternatif/secondaire.

Les messages de fermeture de client (CC) sont utilisés pour réfuter les messages d'ouverture de client ou pour notifier l'autre cote (PEP ou PDP) que ce type de client n'est plus supporté ou actif. D'autre par le PDP se sers des messages de fermeture de client lorsqu'il veut se déconnecter.

3.2.8 Sécurité

Le protocole COPS peut fournir une sécurité pour les messages échangés entre le PEP et le PDP. L'utilisation de cette sécurité est facultative; elle doit être, lorsque requise, négocié durant la phase initiale d'échange des messages Ouverture de client (OPN) / Acceptation de client (CAT) en spécifiant zéro comme type de client[10] (au premier message d'ouverture de client échangé entre un PEP et un PDP) sinon elle est refusée. Une fois la sécurité négociée, tous les messages échangés entre le PEP et le PDP contiendront l'objet Intégrité (qui assure la sécurité)

L'objet Intégrité contient une clé *Key ID* et un numéro de séquence. Cette clé identifie une clé et un algorithme. Ces deux derniers servent à calculer le message digeste nécessaire pour l'authentification et l'intégrité. Le numéro de séquence permet d'éviter les attaques de re-joue (replay attacks). Pour plus de détails sur la sécurité voire dans COPS [8]

[9] Des règles spécifiques pour un tel comportement doivent être définies dans l'extension appropriée des spécifications des types de clients de COPS

[10] Le type de client « zéro » est réservé pour négocier la sécurité de la connexion et pour vérifier la connexion

4 Quelques types de clients

Dans cette partie nous allons traiter trois types de client : COPS-RSVP, COPS-PR et COPS-SLS.

4.1 COPS-RSVP

COPS-RSVP[11] utilise le modèle *Outsourcing*, où chaque nouveau flux est traité séparément. En effet, un contrôle d'admission par politique est accompli pour chaque flux RSVP. La façon dont la réservation (style de réservation) est faite influe sur la définition du flux (cf. § 4.1.1)

Certains objets du protocole COPS sont affectés : *Context*, *ClientSI* et *Decision,* quand le protocole RSVP est utilisé.

a- ***Objet Contexte :*** *Cet objet devra indiquer les messages RVSP suivant:* Path, Resv, PathErr *et* ResvErr *; que COPS ne supporte pas. [9]*
b- ***Objet ClientSI :*** *L'objet* Signaled ClientSI *encapsule tous les objets reçus dans un message RSVP*
c- ***Objet de décision :*** *Certaines options de cet objet ne sont pas utilisées.*

4.1.1 Le protocole RSVP

RSVP est un protocole de signalisation pour la réservation de ressources. Il est orienté récepteur car c'est le récepteur du flux de données qui est responsable de l'initiation de la réservation de ressources. La réservation est faite dans une seule direction (émetteur vers récepteur).

D'autre part RSVP fournit différent style de réservation pour les flux. Trois styles sont définis:

a- **WF (Wilcard-Filter):** Une réservation est partagée par un groupe d'émetteurs (la réservation correspond à la demande la plus importantes).
b- **SE (Shared-Explicit):** Identique à WF sauf que les émetteurs sont déclarés explicitement.
c- **FF (Fixed-Filter):** Une réservation distincte par émetteur (Il n'y a pas de partage et la réservation correspondant à la somme des demandes).

Le style est associé à une réservation, ce qui permet d'effectuer des agrégations entre les différents flots d'une session.

La fonction de RSVP est d'établir et de maintenir des réservations de ressources sur un chemin, et cela de façon indépendante de la manière par laquelle le chemin a été crée.

RSVP utilise 2 messages principalement: PATH et RESV.

Le message *PATH* est transmis par l'émetteur vers le récepteur, il établit un état dans les nœuds traversés et transmet la description du flux de l'émetteur au récepteur. Ce message suit

[11] COPS-RSVP est le premier type de client de COPS

le même chemin que les données et construit un arbre de parcours inverse que suivront les messages *RESV*.

Le message *RESV* est émis par le récepteur vers l'émetteur par le chemin défini avec le message *PATH*. La réservation est réalisée a l'aide des messages *RESV* en une passe ou mieux.

Des messages « *Keep Alive* » sont émis périodiquement afin de maintenir les chemins et les réservations utilisées.

4.1.2 Fonctionnement de COPS-RSVP

Dans RSVP, l'admission d'une nouvelle réservation doit passer par deux phases l'accord administratif (policy control) et le contrôle de la capacité d'admission. Une fois que la réservation traverse ces deux phases elle peut être installée.

Si l'installation de la réservation se fait avec succès, alors le PEP notifie le PDP en envoyant un message de rapport indiquant le succès (*commit type*[12]), sinon si l'installation échoue due à un manque de ressources le PEP doit émettre un message de rapport indiquant l'échec (no-commit type).

Vu que le protocole COPS est *Stateful*[13], le PEP n'a pas besoins d'envoyer constamment, au PDP distant, des rafraîchissements pour les messages RSVP: *Path* et *Resv*. Donc le PEP mémorise les décisions retournées par le PDP et les utilise pour les futures mises à jour.

Quand le PEP détecte des changements dans les messages *Resv* et *Path* , il met à jour le PDP par l'intermédiaire de requêtes. Entre temps le PEP peut continuer à utiliser les décisions sauvegardées jusqu'à la réception de la réponse du PDP.

Si la connexion entre le PEP et le PDP est perdue, les décisions mémorisées (cachée) par le PEP peuvent être utiliser pendant un certain temps par ce dernier pour les flux déjà admit. Une fois ce temps dépassé le PEP doit purger toutes les décisions mémorisées. Lorsque la connexion est rétablie avec un PDP (ancien ou nouveau), ce dernier peut faire une demande de synchronisation SSQ au PEP. Dans ce cas là le PEP doit émettre une requête qui correspond à l'état RSVP courant (c'est comme si tous les états avaient été mis à jours) ainsi que les décisions qui sont toujours mémorisées par le LPDPDecision.

Dans RSVP les messages sont élaborés en trois étapes distinctes : entrée, allocation de ressources et sortie, nécessitant chacune une décision.

En cas d'erreur, RSVP utilise l'objet *ERROR_SPEC*[14] dans les messages *PathErr* et *ResvErr* pour rapporter l'erreur. Le contenu (sub-codes) de l'objet *ERROR_SPEC* est fournit par le PDP.

En ce qui concerne la sécurité COPS-RSVP n'utilise que la sécurité fournit par COPS.

---

[12] Le message de rapport *Commit Type* indique quand la facturation doit commencer
[13] Mémorise les états
[14] Cet objet est définit dans [4] [8]

4.2 COPS-PR

COPS-PR est un modèle de configuration ou le PDP provisionne le PEP afin qu'il réagisse aux événements. Le provisionnement peut se faire en portion (ex. mise à jours d'un élément diffserv) ou en une seule fois (ex. configuration entière d'un routeur).

La plupart du provisionnement des ressources du réseau est basé sur les statiques SLA[15] (Service Level Agreements).

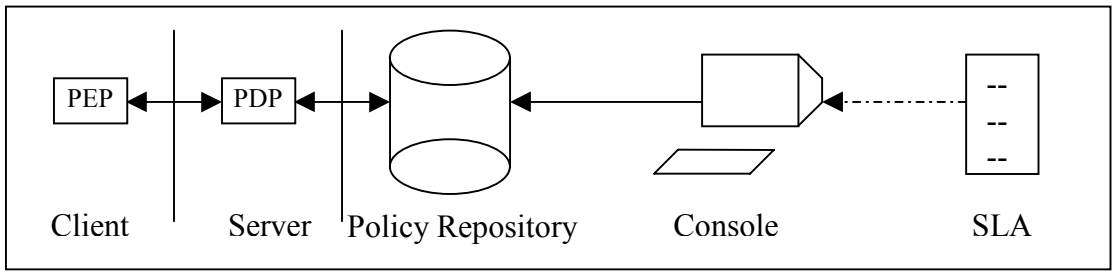

Figure 4.1: Schéma général

Dans COPS–PR, les requêtes de politiques envoyées par le PEP, servent à donner une description du PEP et de ses paramètres de configuration. Si ces paramètres changent une requête de mise à jours est envoyée.

Il est à noter que les décisions ne sont pas forcément liées directement aux requêtes. Elles sont plutôt issues quand le PDP répond à des évènements externe ou à des évènements de PDP (Policy/SLA updates).

Le model (COPS-PR) est basé sur le concept des *Policy Information Bases* (PIBs). La PIB définit les donnés politiques.

On peut avoir une ou plusieurs PIBs par région de politique. Les différentes régions de politiques peuvent avoir un ensemble de PIBs différent.

Pour supporter un modèle à plusieurs PIBs le type de client spécifié par le PEP au PDP doit être unique par région (c.à.d. un seul type de client (exp. QoS) sera utiliser pour toutes les PIBs qui existent dans cette région).

Le protocole COPS-PR encapsule six nouveaux objets dans les objets *Named Client Specific information* (Named ClientSI) et *Named Decision Data.* Les nouveaux objets sont les suivants: PRID, PPRID, EPD, GPERR, CPERR et ErrorPRID (cf. Annexe C).[10]

[15] Accord entre un administrateur de réseau et un client (accord technique, économique,…)

4.2.1 Interaction entre le PEP et le PDP.

Quand un équipement est mis en marche, il ouvre une connexion TCP avec son PDP primaire. Une fois que la connexion est établie, le PEP envoie un message d'ouverture de client (OPN) avec comme type de client ‘‘COPS-PR’’.
Si le PDP supporte ce type de client, il émet un message d'acceptation de client (CAT) sinon il émet un message de fermeture de client (CC). Ce dernier message peut éventuellement contenir l'adresse d'un serveur alternatif qui supporte ce type de client.

Une fois le type de client accepté, le PEP envoie au PDP un message de requête de configuration qui contient des informations sur lui (exemple: type de Hardware, type de logiciel, informations de configuration). Ces informations sont contenues dans l'objet *Named ClientSI* (cf. § A.9 de l'Annexe C). Il est à noter que durant cette phase le client (PEP) peut spécifier la taille maximale des messages COPS-PR supportés.

Le PDP répond à cette requête en envoyant dans un message de décision toutes les politiques provisionnées applicables à cet équipement.

Dés que le client reçoit ces politiques provisionnées, il les installe et notifie le PDP du résultat (succès/échec)de l'installation par l'intermédiaire de messages de rapports (RPT).

Le PDP peut modifier (installer, mettre à jour, enlever) les politiques provisionnées par l'intermédiaire de messages de décision (non sollicités).

Le PDP peut aussi demander au PEP d'ouvrir une nouvelle requête ou d'effacer une requête existante grâce à un nouveau drapeau défini dans COPS-PR.

Tout changement de l'état du PEP peut être communiquer au PDP par l'intermédiaire de messages de requêtes (non sollicités) de mise à jours de la configuration.

4.2.2 Policy Information Base (PIB)

Les données transportées par COPS-PR sont un ensemble de données politiques. Le protocole COPS-PR supporte une structure de données nommée « PIB » qui identifie le type et le but des informations non sollicitées qui sont ‘poussées’ par le PDP au PEP ou envoyées au PDP par le PEP comme notification.

Le nom de la PIB est commun au PEP et au PDP. Les instances de données comprises dans cet espace sont uniques pour un type de client et pour la requête d'état dans une connexion TCP entre un PEP et un PDP.

Dans le cas ou le PEP contient plusieurs types de client, chaque type de client aura une instance unique. Il n'y a pas de partage de données entre les instances des types de client implémenter par le PEP

La PIB est composée d'un ensemble de *Provisioning Classes* (PRCs). Chaque PRC peut contenir un ensemble d'instance de provisionnement « *Provisioning Instances »* (PRIs). (Exemple: si on veut installer un filtre pour un contrôle d'accès multiple, le PRC pourrait représenter le filtre d'accès commun et chaque PRI pourrait représenter un filtre d'accès individuel).

Chaque instance de classes de politiques (PRI) est identifiée par une Provisioning Instance Identifier (PRID). Le PRID identifie une instance particulière de la classe. Il est transporté dans les objets COPS *Named ClientSI* et *Named Decsion Data*.

Les PIBs peuvent être modifier pour répondre aux nouveaux besoins. Les modifications peuvent se faire de plusieurs façon:

- Ajouter/supprimer une PRC.
- Les attributs d'une PRC (existante) peuvent être ajouter/supprimer.
- Une PRC (existante) peu être agrandit avec une nouvelle PRC qui est définit dans une autre PIB.

4.3 COPS-SLS

4.3.1 Définition d'une SLS

Une SLS (Service Level Specification) est un ensemble de paramètres et de leurs valeurs. Ces paramètres et valeurs définissent le service offert à un flux dans un domaine Diffserv [13].

4.3.2 Utilisation de COPS-SLS

COPS-SLS est utilisé par les terminaux (end points) pour la négociation de la qualité de service cela est fait dans le but de rendre le terminal plus concerné du trafic qu'il génère sur le réseau et de lui permettre de s'adapter à l'évolution de la charge du trafic dans le réseau.

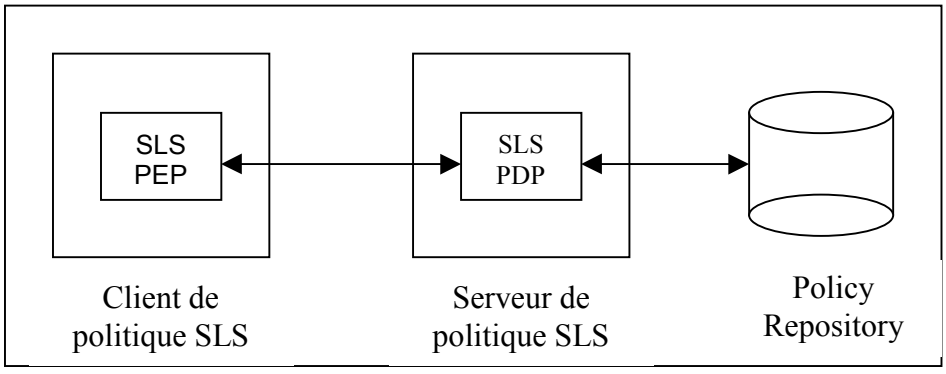

Figure 4.2: Schéma général

COPS-SLS utilise le protocole COPS pour transporter des informations SLS du client au serveur. Dans COPS-SLS, le client est considéré (PEP-SLS) comme un abonné d'un fournisseur de service Internet ISP (Internet Service Provider). Un client peut être :

- Un utilisateur connecté directement à un ISP par le biais d'un modem.
- Une passerelle d'un réseau locale (LAN) connecté à un ISP.
- Un ISP connecté à un autre ISP.

Lors de l'activation du module PEP-SLS, le client essayera de se connecter au serveur (PDP-SLS). Une fois la connexion établie le client pourra demander au PDP le niveau de service requis par l'intermédiaire du message de requête (REQ).

C'est le serveur (PDP-SLS) gère toutes les SLS d'un domaine administratif. Il est capable d'accepter ou de refuser une requête SLS, et aussi de suggérer une autre SLS en cas de refus. Les règles de politiques d'un PDP-SLS sont stockées dans un Policy Repository.

Le protocole COPS-SLS permet au serveur, pour bien configurer le réseau, d'interagir avec d'autres entités (comme le PDP de COPS-PR ou l'entité qui gère le *Policy Repository* du PDP de COPS-PR).

Les deux modèles *Outsourcing* et *Provisionning* supportés par COPS sont utilisés par le protocole COPS-SLS. Ce dernier transporte deux types d'informations: les informations de configurations et les informations de signalisations.

Les informations de configurations servent à définir comment le PEP peut négocier une SLS (Exemple: mode de négociation, l'intervalle de temps durant lequel le PEP ne peux envoyer une requête pour modifier le SLS négocié,…) alors que les informations de signalisations définissent une SLS spécifique.

4.3.3 Les objets du protocole COPS-SLS

Les objets utilisés par COPS-SLS suivent le même format des objets qui sont définis dans COPS-PR. COPS-SLS reprend les six objets définis par COPS-PR: PRID, PPRID, EPD, GPERR, CPERR et ajoute un septième objet: SLSERR.. L'objet SLSERR est utilisé pour indiquer que le PEP n'accepte pas la SLS suggérée.

4.3.4 Déroulement de la communication dans COPS-SLS

Quand le module PEP-SLS est activé, le PEP se connecte à son PDP primaire puis envoie un message d'ouverture de client (OPN) dont le type de client est 'COPS-SLS'. Si le PDP accepte ce PEP, il envoi un message d'acceptation de client (CAT) sinon c'est un message de fermeture de client (CC).

Une fois la connexion établie, le PEP peut commencer à envoyer des messages de requêtes. Deux messages sont nécessaires pour configurer le PEP. Le premier message est un message de configuration dont le but est d'informer le PDP des capacités du PEP (Exemple: les PRC que le PEP comprends, le mode de négociation que le PEP supporte,…) et de tous les SLS qui sont prédéfinis[16], alors que le deuxième message est utilisé pour négocier une SLS.

Il existe deux modes: prédéfini et non prédéfini. Dans le premier cas le PDP installe toutes les SLS prédéfinis dans le PEP. Dans ce mode là le PEP ne peut faire que des requêtes SLS que si elles sont conformes aux SLS prédéfinis tandis que dans le deuxième mode (non prédéfini), le PEP peux faire une requête SLS avec n'importe quelle valeur de paramètres.

[16] Le SLS prédéfini est une SLS qui est déjà défini avec des valeurs.

Le PDP répond au premier message du PEP par un message de décision pour le configurer. A la réception de ce message, le PEP installe la configuration envoyée par le PDP, puis envoie un message de rapport pour informer le PDP du résultat de l'installation (succès ou échec).

Si l'installation de la configuration est réussie alors le PEP pourra envoyer un nouveau message de requête (le deuxième) pour faire une demande d'allocation de ressources (demander une SLS). En réponse à ce message, le PDP envoie un message de décision.

Le PDP peut accepter ou refuse la SLS demandée, il peut aussi proposer une autre SLS.

Si le PEP reçoit un message DEC acceptant/rejetant la requête et arrive à installer la décision, il envoie un message de rapport pour informer le PDP du succès de l'installation, sinon il envoie un message de rapport pour informer le PDP de l'échec.

Dans le cas où le message reçu par le PEP suggérerait une autre SLS, le PEP peut soit l'accepter soit la refuser. Il est à noter que les paramètres d'une SLS négociée peuvent être modifier; pour cela, il suffit que le PEP envoie un message de requête.

Le PEP peut effacer une SLS négociée auparavant par l'intermédiaire du message d'effacement de requête.

Le PDP peut à n'importe quel moment envoyer un message de décision non sollicité soit pour mettre à jours les informations politique, soit pour avilir quelques SLS négociées.

Si Le PEP reçoit un message de synchronisation (SSQ), il doit ré-émettre des requêtes pour toutes les SLS installées puis envoyer un message de fin de synchronisation (SSC) pour indiquer la fin de la synchronisation.

4.4 Conclusion

COPS est un protocole qui a été développé par l'IETF pour assurer des fonctions de contrôle d'admission en les fondants sur des règles. C'est un protocole sous forme de requête-réponse simple permettant l'échange d'informations de politique entre un client ou PEP et le serveur de politique ou PDP pour un type de client particulier.

Des différents type de clients de COPS, COPS-SLS se démarque. En effet, il présente quelques avantages qui le rendent plus flexibles par rapports aux autres clients de COPS. Pour ce client, le terminal de l'utilisateur est considéré comme un PEP, ce qui fait que c'est le terminal qui négocie directement avec le serveur et non avec un équipement intermédiaire comme c'est le cas avec COPS-RSVP. La négociation que propose COPS-SLS est interactive puisque le PDP peut proposer d'autres services que ceux qui sont demandés par le terminal, ces services peuvent être accepté ou refusé. D'autre part, le terminal est capable de modifiés les paramètres qu'il avait négocié préalablement sans être obliger de se déconnecter permettant ainsi un gain de temps.

Partie III

Notre Proposition

1 Introduction

Dans un réseau mobile et plus spécialement un réseau ad-hoc, tous les utilisateurs partagent un même support de transmission. Pour éviter que les conversations soient écoutées ou que les données soient espionnées, il est nécessaire d'incorporer un mécanisme qui sécurise l'envoi de l'information. Des mécanismes qui s'appuient sur le cryptage des données et l'authentification des stations ont été introduits dans les réseaux mobiles, dans le seul but de protéger le contenu des flux qui voyagent sur l'interface radio. Or ces mécanismes restent insuffisants pour assurer une sécurité efficace pour une communication de bout en bout. Si on prend l'exemple du Standard IEEE 802.11, la sécurité de données est assurée par le WEP (Wired Equivalent Privacy) qui est une fonctionnalité facultative de la sous-couche de MAC. Le WEP permet juste d'assurer une sécurité entre les stations, mais pas de bout en bout d'où la fragilité de ce système de sécurité. En effet, n'importe quel utilisateur qui a accès au réseau à accès à l'ensemble du réseau. [A] Le réseau ad-hoc 'bluetooth' a aussi des problèmes de sécurités : la sécurité prévue par la norme Bluetooth repose sur une clef de chiffrage secrète or cette clé est partagée par les participants d'où le problème. En effet si quelqu'un connaît la clé secrète il est capable de la déchiffrer, décoder votre conversation. Il peut aussi se faire passer pour vous car la sécurité dans Bluetooth permet d'identifier les machines mais pas les personnes.

D'un autre coté, de plus en plus d'applications sont en train d'intégrer des fonctions multimédias (vidéo conférence, visiophonie, video on demand…) qu'on veut absolument fournir aux utilisateurs de ces réseaux. Ces applications consomment énormément de ressources et ne permettent pas une utilisation efficace et équitable du support de communication surtout quand ils coexistent avec des services de données qui sont caractérisés par des transferts en rafales (Burst). Comme la bande passante des réseaux sans fils est très limitée, il est nécessaire, pour bien exploiter cette ressource, d'utiliser des mécanismes permettant d'assurer une certaine qualité de service (QoS) particulière pour chaque station partageant le support et pour chaque genre de trafic véhiculé sur le médium.

La tendance actuelle, dans les réseaux fixes, pour assurer la sécurité et la qualité de service est d'utiliser un serveur de politique ou PDP (Policy Decision Point) dont on a parle dans le chapitre précédent. Ce serveur prend des décisions et les transmet à ses clients appelés PEP (Policy enforcement Point) qui appliquent les politiques. Le protocole de signalisation utilisé pour transporter les décisions prisent par le PDP est le protocole COPS (Commnon Open Policy Service) qui est un protocole de requête-réponse simple et qui permet l'échange des informations de « politique » entre le PEP et le PDP, pour un type de client particulier.

L'utilisation d'un tel concept qui permettra d'assurer un contrôle sécurité de bout en bout ainsi qu'une bonne qualité de service (QoS), dans les réseaux ad-hoc va nécessiter une réelle adaptation. Car contrairement aux réseaux fixes, qui eux reposent sur une infrastructure et qui contiennent des serveurs dédiés (routeurs et de terminaux), les réseaux ad-hoc ne reposent sur aucune infrastructure et ne sont composés que de terminaux mobiles.

En premier, nous expliquerons dans ce chapitre comment nous allons adapter le modèle par politique, dont on a parlé au chapitre précédent et qui a été définie pour un réseau fixe, à un réseau purement ad-hoc ainsi que les divers cas de figures qui peuvent se produire et donc démontrer la **robustesse** de notre proposition.

Dans un deuxième temps nous expliquerons les différents Algorithmes proposés (Calcul du poids, Nouvel Adhérent,…) et finalement nous donnerons une proposition de l'utilisations d'un client COPS (COPS-SLS) et nous expliquerons notre choix.

2 Adaptation du modèle par politique au réseau Ad-hoc

Notre travail consiste donc à adapter ce modèle par politique aux réseaux ad-hoc et donc expliquer comment introduire un serveur de politique (PDP) dans un tel réseau. Cette solution va permettre de contrôler, à base de politiques, ce réseau sans que cela aille à l'encontre de la définition ad-hoc.

Le problème qui se pose est que le réseau ad-hoc, est un réseau composé que de nœuds mobiles et qui ne repose sur aucune infrastructure préexistante. Notre proposition consiste à élire un des terminaux du réseau pour qu'il devienne serveur de politique. Cette élection se fera par l'intermédiaire d'une entité qui va déterminer le degré de capacité pour un terminal à être PDP et qu'on appelle "poids du terminal".

2.1 Poids du terminal

Le calcul du poids du terminal suit des critères précis. Ces critères vont permettre de déterminer si le terminal a les ressources nécessaires pour devenir un serveur de politique PDP ou pas sans avoir à pénaliser l'utilisateur. En effet, le terminal qui va jouer le rôle de serveur de politique ne doit, en aucun cas, être dédié seulement à cette tâche. L'utilisateur ne se connecte pas rien pour gérer le réseau mais pour pouvoir communiquer et échanger des informations avec d'autres utilisateurs. Pour cette raison, le choix des critères doit être rigoureux.

Voici une liste non exhaustive de critères :

- multitâche
- processeur
- ram
- mémoire de stockage
- autonomie,…

Pour bien déterminer les capacités du terminal, nous avons séparé ces différents critères en deux grandes catégories :

1) **Les critères liés à des conditions nécessaires.** Cette catégorie contient les critères, contient les critères qui sont indispensables au terminal afin de lui permettre de supporter un serveur de politique (exp. multitache). En effet, si un de ces critères n'est pas supporter par le terminal, il ne pourra pas être élu comme serveur de politique car cela risque de dégrader ses performances pénalisant ainsi l'utilisateur. Dans le cas ou, le terminal ne supporte pas un des critères liés aux conditions nécessaires la valeur "zéro" est affecté au poids.

2) **Les critères liés à des aspects de performances.** Cette catégorie, comprend les critères qui vont servir à départager les terminaux éligibles au poste de PDP (les terminaux qui supportent tous les critères de la première catégorie). En effet, ces critères (exp. RAM, mémoire de stockage, processeur,..) sont liés aux performances de la machine et vont permettre de calculer le poids du terminal. Pour calculer le poids on utilise la formule suivante :

$$\text{Poids} = \frac{1}{\sum_{i=1}^{n} a_i} \sum_{i=1}^{N} a_i \, f_i(x) \qquad (1)$$

où :
N : est le nombre de critères.
a_i : est le poids du critère, sa valeur dépendra l'importance du critère.
$f_i(x)$: est une fonction qui dépend du critère.

Dans ce qui suit, nous allons voir comment calculer ce poids, comment se fait la connexion d'un nouveau terminal ainsi que les différents cas de figures qui peuvent se produire.

Ce poids va être calculer par le terminal en exécutant l'algorithme "*calcul du poids*" qui sera détaillé dans la section 3.1 ci-dessous, dés sa première connexion au réseau ad-hoc. Il permettra donc non seulement de déterminer si le terminal est capable de supporter d'être un PDP mais en plus de choisir le meilleur des terminaux dans le réseau comme PDP(celui avec le plus grand poids).

2.2 Connexion au réseau

Des la connexion d'un nouveau terminal, son poids est calculé et va donc essayer de détecter la présence d'un serveur de politiques PDP. La détection va se faire par l'intermédiaire du protocole SLP[17] (Service Locator Protocol) ou SDP[18] (Service Discovery Protocol). Le choix de l'un de ces deux protocoles dépendra du choix de l'implémentation.

Trois cas de figure, se présentent :

1) Un PDP est déjà présent dans le réseau ad-hoc.

2) Aucun PDP n'est présent dans le réseau ad-hoc.

3) Un PDP est entrain de se déclarer.

[17] SLP est un protocole utilisé par le protocole COPS pour localiser le serveur de politique (cf. Partie II).
[18] SDP est un protocole utilisé par les réseaux ad-hoc pour permettrent aux applications de découvrir quels sont les services qui sont offerts par les autres terminaux (cf. Partie I).

On va détailler ces trois cas dans ce qui suit.

1) Cas où le terminal détecte la présence d'un PDP : Dans ce cas, le terminal va directement se connecter au serveur en tant que client ou PEP. Le terminal va suivre la même méthode de connexion décrite par le protocole COPS.

Au début, le terminal commence par établir une connexion TCP avec le serveur. Une fois cette connexion établie, le client envoie un message d'ouverture de client au serveur en précisant le type de client (On parlera, plus loin, du type de client). Le serveur répond, s'il accepte, par un message d'acceptation de client.

Une fois que la connexion établie avec le serveur, le terminal lui envoie son poids. Le serveur va mémoriser ce poids, si différent de 'zéro', pour l'utiliser ultérieurement. Par exemple : chaque quelque temps le PDP compare le meilleur poids, qu'il a mémorisé, avec son propre poids afin de déterminer s'il existe, sur le réseau, un terminal qui offre de meilleures capacités que lui pour supporter un serveur de politique.

2) Si aucun PDP n'est présent dans le système : le terminal se réfère à son poids, calculé auparavant, pour déterminer s'il peut ou non supporter le rôle d'un PDP. Dans le cas où, le terminal aurait les capacités nécessaires (poids $\neq 0$), il va déclencher la procédure de serveur pour se déclarer comme PDP sinon il va attendre qu'un serveur se déclare pour se connecter en tant que PEP.

3) Un PDP est en train de se déclarer : Ce troisième cas a été ajouté pour éviter d'avoir plusieurs PDP qui se déclarent en même temps. En effet, il ne serait pas convenable d'avoir cette situation. Pour pallier ce problème, le terminal qui se déclare comme PDP devra indiquer, à tout terminal qui essaye de détecter la présence d'un PDP, qu'il est entrain de se déclarer comme serveur. Le terminal qui sera informé ne pourra plus se déclarer comme PDP et devra attendre que le serveur soit mis en place.

Pour pouvoir se connecter le PDP va exécuter un algorithme "*Nouvel adhérent*" qui est décrit dans la section 3.2 ci-dessous.

2.3 Durée de vie d'un PDP

Dans cet paragraphe, on va relater les différents cas de figures que peut prendre un serveur dans un réseau ad-hoc. Trois cas se présentent :

1) Changement de PDP;
2) PDP quitte le réseau;
3) PDP disparaît de façon brusque;

2.3.1 Changement de PDP

Le poids mémorisé par le serveur sera utilisé pour déterminer s'il existe, sur le réseau, un terminal offrant de meilleures capacités pour supporter un PDP que le terminal actuel. Lorsque le serveur détecte un terminal offrant de meilleures performances, il effectue alors un transfert du PDP vers ce terminal. Ce transfert sera effectué seulement si la condition suivante est vérifiée :

$$\text{Taux} = \frac{\text{Poids du terminal le plus élevé}}{\text{Poids du PDP actuel}} > \alpha \qquad (2)$$

Où α : est une valeur au-delà de laquelle un transfert serait nécessaire

Pour éviter un changement fréquent du PDP (à chaque fois qu'il y a un terminal meilleur que le PDP actuel on fait un transfert) qui va causer une instabilité du système on définit un temporisateur. Tant que ce temporisateur n'a pas expiré, aucun transfert n'est possible. A l'expiration du temporisateur, le PDP vérifie la formule (2), si elle est vérifiée alors le transfert aura lieu sinon le PDP re-déclenche le temporisateur.

2.3.2 Le PDP quitte le réseau

C'est le cas ou le terminal qui supporte le PDP veut quitter le réseau c'est donc le cas où l'utilisateur terminé son travail et veut éteindre sa machine. Cette opération doit être précédée par un transfert du PDP vers un autre terminal afin d'éviter que le réseau ne se retrouve sans serveur et donc sans contrôle.

Le transfert va se faire de la façon suivant :

1- Avant de quitter le réseau, le serveur consulte la table dans laquelle il a mémorisé le poids de tous ses clients.

2- Il choisit le client qui a le poids le plus élevé pour lui envoyer un message de décision qui contient une commande. Cette commande oblige le terminal à démarrer la procédure du serveur.

3- Le PDP attend de recevoir un message de rapport du PEP, afin de garantir la présence du PEP. Si le serveur reçoit ce message il envoie un message de fermeture de client à tous ses PEPs afin de les diriger vers le nouveau PDP sinon il n'y aura pas de transfert.

Afin de pouvoir effectuer ce transfert un algorithme " PDP quitte" a été mis au point. Cet algorithme sera décrit dans la section 3.4 ci-dessous.

2.3.3 Le PDP disparaît

C'est le cas où le PDP disparaît brusquement du réseau par exemple le cas où le terminal plante. Les PEPs vont détecter, par l'intermédiaire des messages *Keep Alive* du protocole COPS, la disparition du PDP. Dés qu'un PEP détecte la disparition du serveur, il va essayer de se reconnecter à son serveur (ceci est prévu par le protocole COPS). Mais vu qu'on se positionne dans un réseau ad-hoc cette solution n'est plus suffisante car comme on l'a déjà cité : le PDP est supporté par un terminal qui est mobile. Il a fallut donc modifier le comportement du PEP en le forçant à recommencer une nouvelle connexion et donc, c'est comme s'il venait de se connecter au réseau. Un algorithme "*PDP disparait*" permettant de traiter ce cas là est décrit dans la section 3.5 ci-dessous

Après avoir décrit notre proposition, qui consiste à élire un des terminaux du réseau ad-hoc pour qu'il devienne serveur de politique, nous allons dans la section suivante détailler les algorithmes qui vont traiter cette proposition.

3 Les Algorithmes

3.1 Algorithme "calcul du poids"

Début

Si (toutes les conditions nécessaires = = Vraie)alors /* vérification si le terminal peut Devenir un PDP */
{

Poids = $\frac{1}{\sum_{i=1}^{n} a_i} \sum_{i=1}^{N} a_i f_i(x)$;

/* calcul du Poids du terminal */
/* a_i Poids donné au critère */
/* $f_i()$ est une fonction qui dépend du critère */
/* x est la valeur du critère */

}

Sinon /* Cas ou le terminal ne ne peut devenir un PDP*/
{
Poids = 0 ;
}

Fin.

Explication

Cet Algorithme permet au terminal de calculer son poids.

La première chose qu'il fait c'est de vérifier que tous les conditions requises, pour qu'un terminal puisse supporter un serveur de politique, soient remplis. Si la première condition n'est pas remplie le poids du terminal est mis automatiquement à "0" sinon le poids est calculé par l'intermédiaire de la formule (1).

3.2 Algorithme "Nouvel adhérent"

```
Début :

Rechercher de la présence d'un PDP.                    /* Peut se faire via des protocoles comme
                                                          SLP ou SDP*/

Si  (PDP_possible  = = Vrai) alors                     /* Un terminal démarre le service serveur
{                                                         de politique */
        Déclenchement d'un temporisateur 'X' ;         /* attente que le PDP qui se déclare se
                                                          mette en route */
        Re-déclenchement de l'Alog"Nouvel adhérent" ;   /* re-démarrage de l'algorithme*/
}

Si (PDP_existe = = Vrai) alors                         /* PDP présent dans le réseau */
{
        la machine se connecte en tant que PEP         /* Echange des messages d'ouverture de
                                                          client avec le serveur */
        Emission du Poids de la machine au PDP         /* Poids calculé par l'algo "Calcul du
}                                                         poids" */

Sinon                                                  /* Pas de PDP présent dans le    système
{                                                         */
        Si ( Poids  ≠ 0)                               /* La machine est capable de devenir un
        {                                                 serveur */
                PDP_possible = Vrai ;                  /* Pour indiquer aux autres terminaux
                                                          qu'il démarre la procédure de serveur */
                Alors il devient PDP;                  /* Le terminal déclenche la procédure de
        }                                                 démarrage du serveur */

        sinon                                          /* la machine ne peut devenir un     PDP
        {                                                 */
                Tant que (PDP = faux)                  /* Boucle pour détecter la présence d'un
                {                                         serveur */
                        Recherche de la présence d'un PDP ; /* Peut se faire via des protocoles
                                                               comme SLP ou SDP*/
                        Si {PDP_existe = = Vrai)       /* Boucle pour la recherche d'un
                        {                                 PDP */
                                PDP = vrai;
                                                       /* Pour sortir de la boucle vu que le
                                la machine se connecte-en  PDP a été détecté */
                                tant que PEP;          /* Echange des messages d'ouverture
                        }                                 de client avec le serveur */
                }
        }
}
Fin
```

Explication

Au début, l'algorithme essaye de détecter la présence d'un serveur de politique. Trois réponses seront possibles :

1- **Un PDP est entrain de se déclarer :** Dans le cas ou un PDP est entrain de se déclarer, un temporisateur est déclenché afin de permettre au serveur qui se déclare de se mettre en route. A l'expiration du temporisateur l'algorithme re-démarre.

2- **Le PDP existe :** C'est le cas où un PDP est détecté la machine va dans un premier temps se connecter à ce serveur. Dans un deuxième temps, le poids de la machine calculé par l'algorithme "*calcul du poids*" définie dans la section précédente, est envoyé au serveur.

3- **Le PDP n'existe pas :** Dans ce cas il existe deux possibilités :

   a- Soit le terminal a la possibilité de supporter un serveur (poids □ 0). Dans le cas, la « possibilité de PDP » est déclarée puis la procédure de démarrage de PDP est mise en route.

   b- soit le terminal n'a pas la possibilité de supporter un serveur (poids = 0). Dans ce cas, on rentre dans une boucle pour détecter un serveur. Une fois que le serveur est détecté la machine se connecte en tant que PEP.

3.3 Algorithme "changement de PDP"

Début

```
Tant que (PDP = = Faux)                                   /* Y c'est le temps qui doit
{                                                          s'écouler avant de pouvoir faire un
        Attendre un temps 'Y';                             changement de PDP (c'est pour la
                                                           stabilité) */
        PDP =  Faux ;

        Taux = Poids du terminal  le plus élevé / Poids du PDP actuel ;   /*  calcul du taux pour le terminal
                                                           qui a le poids le plus élevé */
        Si (Taux  > α )                                    /*  Si le taux calculer
                                                           précédemment est supérieur a une
        {                                                  certaine valeur 'α' alors il peut y
                PDP = = Vrai ;                             avoir un changement */
                Exécution  de l'algorithme "PDP quitte";   /* pour sortir de la boucle */
        }
}
```

Fin

Explication

Cet Algorithme, permet que le serveur de politique soit transféré vers une autre machine si cette dernière possède de meilleures performances.
L'algorithme rentre dans une boucle qui va permettre de détecter si un terminal a de meilleures performances que lui. Au début l'algorithme déclenche un temporisateur, qui à pour rôle d'éviter que le PDP change trop souvent de main car ceci va rendre le système instable. Par la suite, il vérifie si l'un des poids déjà reçus de la part de ses clients susceptibles de devenir PDP est largement supérieur à son poids. Il calcule donc un taux à part de son poids et du poids reçu le plus élevé. Si ce taux est assez grand ($> \alpha$) le transfert de PDP pourra avoir lieu vers ce dernier terminal (le terminal avec le plus grand poids dans le réseau ad-hoc).

3.4 Algorithme "PDP quitte"

```
Début

Si (PDP_quitte = = Vrai)
{
        Envoie d'un message au PEP qui a le meilleur poids  pour
        l'informer qu'il devra enclencher la procédure PDP dès
        qu'il recevra un message de fermeture de cleint;

        Envoie un message de fermeture de client contenant  l'adresse du
        PDP alternatif à tous ses PEPs ;

        Fermeture du PDP ;
}
Fin
```

Explication

Cet Algorithme traite le cas où le terminal qui supporte le PDP désire quitter le réseau, exp.la machine est en train de s'éteindre.

3.5 Algorithme "PDP disparait"

```
Début

Si (PDP_existe  = = Faux)  alors
{
        Exécution de l'algorithme "nouvel Adhérent" ;
}
```

/* La détection de la perte du PDP se fait par l'intermédiaire des messages Keep Alive */

Explication

Cet algorithme traite le cas ou le serveur de politique disparaît de façon imprévue.
Nous venons d'expliquer comment nous proposons d'élire un serveur de politique dans un réseau ad-hoc et nous avons détaillé les divers cas de figures qui peuvent se produire cependant nous n'avons pas encore parlé du type de client COPS qui pourrait être utilisé.

4 Le type de client

Le type de client COPS qui semble, actuellement, convenir le mieux aux réseaux ad-hoc est COPS-SLS. En effet, ce client présente de nombreux avantages par rapport aux autres types de client, comme COPS-RSVP ou COPS-PR. Un des avantages de COPS-SLS c'est qu'il considère le terminal de l'utilisateur comme PEP ceci dans le but de le rendre plus concerné du trafic qu'il génère sur le réseau. C'est donc le terminal qui négocie le service qu'il désire directement avec le serveur et non plus par l'intermédiaire d'un PEP comme c'est le cas dans COP-RSVP.

Un autre avantage c'est qu'il permet une négociation assez malléable. En effet il permet au serveur PDP de proposer lors de la négociation d'autres services que ceux qui sont demandés par le client d'où l'avantage. Il est à noter que le client peut après accepter ou refuser ce service. D'autre part, COPS-SLS offre la possibilité aux PEPs de pouvoir modifier les paramètres du service qui ont été négociés préalablement, ce qui permet aux PEPs de gagner du temps et éviter d'interrompre le service entre le PEP et le PDP.

Afin que notre proposition puisse fonctionner avec COPS –SLS, on propose d'ajouter, pour l'instant, deux nouveaux objets :

- Le premier objet devra permettre de transporter le poids du terminal du client au serveur PDP. On l'apeller objet **PoidsT**
- Le deuxième objet devra transporter une commande du PDP au PEP pour obliger le PEP à devenir serveur. **CommandeS**

Dans les sections ci-dessous on va expliquer quand et comment ces objets doivent être transmis.

4.1 L'objet PoidsT

<table>
<tr><td></td><td>16</td><td>8</td><td>8</td></tr>
<tr><td>Longeur
(=8)</td><td colspan="2">S-Num =PoidsT
(à définir)</td><td>S-Type = BER
(=1)</td></tr>
<tr><td>Valeur
(contiendra le poids du PEP)</td><td colspan="3">XXXXXXXX</td></tr>
</table>

Voici comment l'objet PoidsT sera envoyé par le PEP au PDP pour lui faire parvenir son poids.

La première chose que va faire un PEP, après avoir établit une connexion avec le serveur, est d'envoyer un message d'ouverture de client avec le type de client « COPS-SLS ». Le serveur répondra, dans le cas positif par un message d'acceptation de client, sinon le serveur répondra par un message de fermeture de client. Pour l'instant, rien de nouveau, c'est le comportement classique de COPS.

Une fois que le PEP reçoit le message d'acceptation de client, il envoie un message de requête de configuration et va attendre la décision du PDP. En outre des informations que le message de requête contient, on envoie aussi le poids du terminal qui sera mémorisé par le PDP. L'objet **Poids** qui sera encapsulé dans le l'objet optionnel *Named ClientSI*.

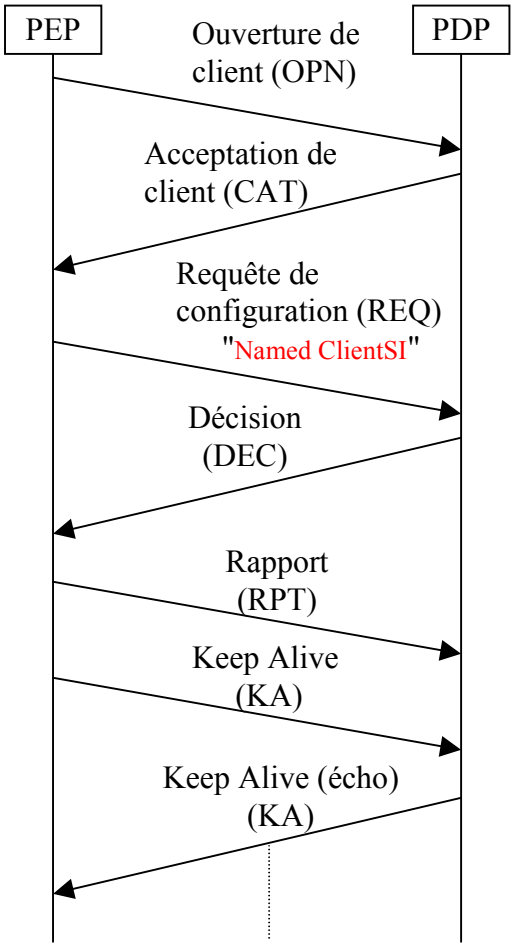

Figure 1: Etablissement de la connexion avec échange du poids

4.2 L'objet commandeT

16	8	8
Longeur (=8)	S-Num = CommandT (à définir)	S-Type = BER (=1)
Valeur (contiendra le poids du PEP)	XXXXXXXX	

Voici comment l'objet **CommandT** sera envoyé par le PDP au PEP pour l'obliger à démarrer le serveur.

Lorsque le PDP détecte, après avoir comparer son poids avec le poids du PDP le plus élevé qu'il avait mémorisé, qu'il existe un terminal qui possède de meilleure capacité que lui, il lui envoie un message de décision pour l'informer qu'il va être PDP. Ce message de décision ne contiendra pas de décision, mais plutôt l'objet optionnel *Named Decision Data* qui transportera l'objet **CommandT** qui va obliger le PEP à démarrer la procédure de PDP. Part la suite le PDP envoie à tous ses PEPs un message de fermeture de client qui contiendra l'objet *PDPredirect* pour les diriger vers le nouveau PDP. Avant d'envoyer ce message de fermeture le PDP attend de recevoir un message de rapport de ce client. Ce message permet au PDP de s'assurer de la présence du PEP dans le système.

Dans le cas ou le PDP est entrain de quitter, il envoie un message de décision, ne contenant que l'objet optionnel *Named Decision Data,* et pas d'objet de décision. Cet objet encapsulera le même objet utilisé dans le cas ou le PDP effectue un transfert. Une fois ce message envoyé, le PDP envoie à tous ses clients un message de fermeture de client qui contient l'objet *PDPredirect*. Cet objet servira à rediriger les PEPs vers le nouveau PDP.

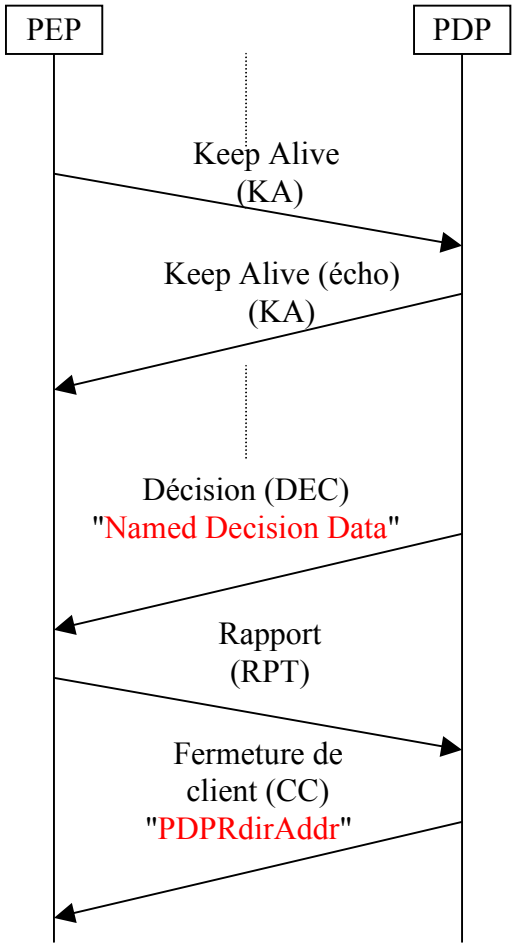

Figure 2 : Transfert du PDP dans le cas il existe un terminal offrant de meilleure capacité que le PDP actuel

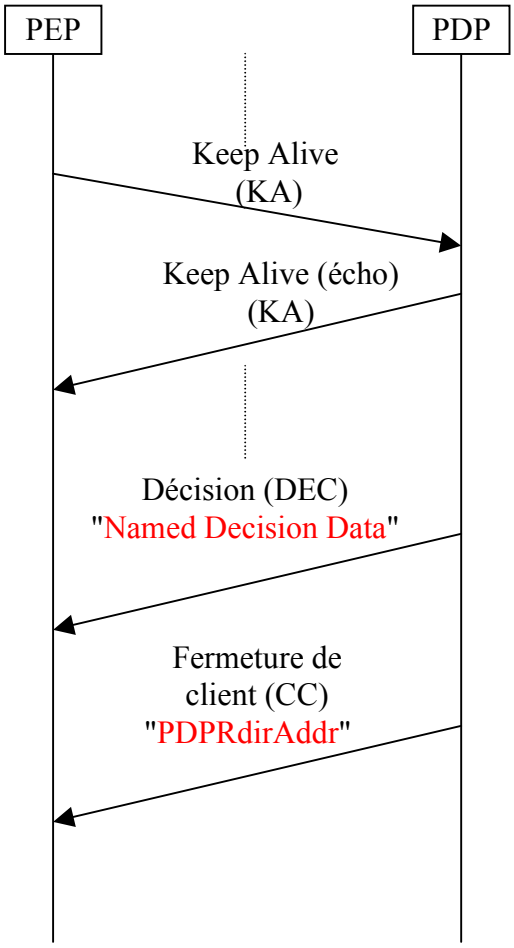

Figure 3 : Transfert du PDP dans le cas où le PDP quitte le réseau

5 Conclusion

Dans cette partie nous avons expliqué comment nous nous sommes pris pour adapter le modèle de contrôle par politique dans les réseaux ad-hoc. La méthode proposée permet de mettre en place un serveur de politiques (PDP) en se basant sur le poids des nœuds du réseau. Ce poids permet, en un premier temps, d'éliminer tout terminal qui ne possèderait pas les ressources nécessaires pour supporter le serveur. Dans un second temps, le poids permet de choisir le meilleur nœud du réseau capable de supporter le PDP. Cette méthode utilise, comme type de client, COPS-SLS car il semble convenir le mieux aux réseaux ad-hoc à cause des avantages qu'il propose par rapport aux autres types de clients.

Afin de rendre notre proposition plus robuste, nous avons traité le cas de la contention entre deux machines pour le rôle de PDP et le cas d'instabilité causé par un changement trop fréquent de PDP. Nous avons aussi essayé de rendre cette méthode assez flexible pour qu'elle puisse s'adapter à la dynamicité du réseau.

Conclusion et perspectives

Les problèmes de sécurité et de qualité de services QoS sont des problèmes rencontrés non seulement dans les réseaux fixes mais dans les réseaux ad-hoc aussi. Pour pallier à ce type de problèmes, la tendance actuelle pour les réseaux fixes repose sur l'utilisation d'un serveur de politiques PDP qui applique des politiques de sécurité et/ou de qualité de service QoS sur des PEPs.

Nous avons essayé dans ce travail d'adapter cette approche aux réseaux ad-hoc. Le plus grand problème, pour la mise en place d'une infrastructure de contrôle à base de politiques, dans ce genre de réseaux est la localisation du serveur de politiques PDP. En effet, vu que les réseaux ad-hoc ne reposent sur aucune infrastructure préexistante et que les nœuds sont mobiles, on a été amené à proposer une méthode flexible capable de s'adapter à la dynamicité de ce réseau.

La méthode repose sur l'élection d'un des nœuds du réseau pour qu'il supporte le rôle d'un serveur de politiques (PDP). Cette élection va se faire par l'intermédiaire du poids du nœud qui est calculé localement à partir d'un certain nombre de critères. Ces critères ont pour rôle d'éliminer, de la phase d'élection, tout terminal ne présentant pas les ressources nécessaires pour supporter un PDP.

Au cours de notre étude, différents scénarios pouvant se produire dans un réseau ad-hoc ont été traités permettant ainsi de rendre notre méthode plus flexibilité :

- Un PDP est présent dans le réseau ;
- Aucun PDP n'est présent dans le réseau ;
- Un PDP est entrain de se déclarer ;
- Le nœud supportant le PDP est entrain de quitter le réseau ;
- Le nœud supportant le PDP a disparu de façon brusque suite à un dysfonctionnement ;
- Le PDP est transféré vers un nœud qui présente de meilleures capacités que le le PDP actuel ;

D'autre part, pour fournir une robustesse à la méthode qu'on propose, les deux cas suivants ont été étudiés:

- Contention entre deux machines pour le rôle de PDP ;
- Instabilité du réseau due à un changement trop fréquent du PDP ;

Le type de client du protocole COPS qui est COPS-SLS, semble mieux convenir à notre approche. Cependant certaines modifications ont du être faite pour adapter ce type de client à notre proposition, comme l'ajout de deux objets (un pour le poids du terminal et un pour la commande qui force le PEP à démarrer le serveur). L'avantage que ce client offre par rapport aux autres clients COPS est le fait que le PEP soit dans le terminal, permettant ainsi d'offrir une négociation interactive entre le terminal et le PDP ainsi que la possibilité de modifier les paramètres négociés préalablement.

Une validation de cette proposition serait intéressante afin de déterminer sa robustesse et sa faisabilité.

Afin de compléter la mise en place d'une infrastructure qui sera chargé de contrôler un réseau ad-hoc, un certain nombre de points tels que la sécurisation du PDP par rapport à l'utilisateur et la localisation du *Policy Repository* doivent être étudiés. L'étude de politiques propres aux réseaux ad-hoc pour la sécurité et la qualité de service QoS rentre aussi dans les perspectives de cette étude.

Abréviations

A
ACL: Asynchronous Connection Less
ACK: Acknowledge
AM_ADDR: Active Member Address
AODV: Ad-hoc On demand Distant Vector
AP: Access Point
AR_addr: Access Request Address
ARQN: Acknowledge indication

B
BD_addr: Bluetooth Device Address
BLUETOOTH: Bluetooth
BPSK: Binary Phase Shift Keying
BSS: Basic Set Service

C
CCA: Clear Channel Assessment
CAC: Channel Access Code
CC: Call Control
CL: ConnextionLess
CRC: Cyclic Redundancy Check
CSMA/CA: Carrier Sense Multiple Access / Collision Avoidance
CSMA/CD: Carrier Sense Multiple Access / Collision Detection

D
DAB: Digital Audio Broadcasting
DAC: Device Access Code
DCF: Distributed Coordination Function
DIAC: Dedicated Inquiry Access Code
DIFS: DCF Inter Frame Spacing
DVB: Digital Video Broadcasting
DSDV: Destination Sequence Distant Vector
DSR: Dynamic Source Routing
DSSS: Direct Sequence Spread Spectrum

F
FHSS: Frequency Hopping Spread Spectrum)

G
GIAC: General Inquiry Access Code
GM: Group Management
GSM: Global System for Mobile Communication

H
HCI: Host Controller Interface
HEC: Header Error Check

I
IAC: Inquiry Access Code
IBSS: Independent Basic Set Service
IEEE: Institte of Electrical and Electronics Engineers
IETF: Internet Engineering Task Force
IFS: Inter Frame Spacing
IP: Internet Protocol
IR: InfraRouge
IrDA: Infrared Data Association
IrOBEX: Infrared Object Exchange Protocol
ISM: Industrial Scientific and Medical
ITU: International Telecom Union
IV: Initialisation Vector

L
L2CAP: Logical Link Control and Application Protocol
LAN: Local Area Network
LAP: lower address part
LLC: Logical Link Control
LM: Link Manager
LMP: Link Manager Protocol
LSB: Least Significant Bit

M
MAC: Medium Access Control
MANET: Mobile Ad-hoc NETwork
MSB: Most Significant Bit

N
NACK: Negative Acknowledge
NAP: non-significant address part
NAV: Network Allocation Vector

O
OBEX: Object Exchange Protocol
OLSR: Optimized Link State Routing Protocol
OSA: Open System Authentification
OSI: Open System Interconnection
OFDM: Orthogonal Frequency Division Multiplexing

P
PAN: Personal Area Network
PDA: Personal Digital Assistant
PCF: Point Coordination Function
PDU: Paquet Data Unit
PIFS: PCF Inter Frame Spacing
PLCP: Physical Layer Convergence Protocol
PMD: Physical Medium Dependent
PSM: Protocol/Service Multiplexer
PSTN: Public Switch Telephony Network
PM_Addr: Parked Member Address

Q
QoS: Quality of Service
QPSK: Quadrature Phase Shift Keying

R
RFCOMM

S
SAR: Segmentation and Reassembly
SB: station de base
SCO: Synchronous Connection Oriented
SDP: Service Discovery Protocol
SEQN: Sequence Number
SIG: Special Interest Group
SIFS: Short Inter Frame Spacing
SKA: Shared Kay Authentification
SLP: Service Location Protocole

T
TCS: Telephony Control Specification
TDD: Time Division Duplex
TIM: Traffic Information MAP
TTL: Time To Live

U
UAP: upper address part
UM: Unité mobile

W
WEP: Wired Equivalent Privacy
WAP: Wireless Application Protocol
WLAN: Wireless Local Aera Network

Références

[1] "The universal radio interface for ad hoc, wireless connectivity", Jaap Haartsen.

[2] « Aman Kansal », 'http://www.Bluetooth.amankansal.com'

[3] Bluetooth Specifications V1.1 CORE, http://www.Bluetooth.com/developer/specification/Bluetooth_11_Specifications_Book.pdf

[4] "The Bluetooth Radio System", Jaap Haartsen.

[5] Edition Eyrolles "Réseaux de mobiles & réseaux sans fil", Khaldoun Al Agha, Guy Pujolle, Guillaume Vivier

[6] http://www.stelvio.nl/documents/wep-solution.pdf

[7] http://xforce.iss.net/alerts/advise84.php

[8] RFC 2748, "The COPS (Common Open Policy Service) Protocol", J. Boyle, R.Cohen, D. Durham, S. Herzog, R. Rajan, A. Sastry; January 2000.

[9] RFC 2749, "COPS usage for RSVP", J. Boyle, R. Cohen, D. Durham, S. Herzog, R. Rajan, A. Sastry; January 2000.

[10] RFC 3084 "COPS Usage for Policy Provisioning (COPS-PR)", K. Chan, J. Seligson, D. Durham, S. Gai, K. McCloghrie, S. Herzog, F. Reichmeyer, R. Yavatkar, A. Smith; March 2001.

[11] RFC 2205, "Resource ReSerVation Protocol (RSVP) – Functional Specification", Braden, R. Zhang, L., Berson, S., Herzog, S. and S. Jamin, September 1997

[12] T.M.T. Nguyen, G. Pujolle "Université de Paris 6", N. Boukhatem "ENST Paris", COPS Usage for SLS negotiation (COPS-SLS), <draft-nguyen-rap-cops-sls-00.txt>; June 2001.

[13] Dan Grossman "Motorola, Inc."Expires, New Terminology for Diffserv, <draft-ietf-diffserv-new-terms-05.txt>; August, 2001.

[14] S. Salsano, F. Ricciato, M. Winter, G. Eichler, A. Thomas, F. Fuenfstueck, T. Ziegler, C. Brandauer, "Definition and usage of SLSs in the AQUILA consortium", <draft-salsano-aquila-sls-00.txt>; November 2000, Work in pogress.

[15] RFC 2750, "RSVP Extensions for Policy Control", Herzog, S.; January 2000.

[16] RFC 2608, "Service Location Protocol, Version 2", Guttman, E., Perkins, C., Veizades, J. and M. Day; June 1999.

[17] http://www.ietf.org/html.charters/manet-charter.html

[18] "Ad hoc On-Demand Distance Vector (AODV) Routing" <draft-ietf-manet-aodv-09.txt>, C. Perkins, E. Belding-Royer, S. Das, 9 November 2001.

[19] "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)",<draft-itef-manet-dsr-06.txt>, D. Johnson, David A. Maltz "AON Networks", J. Jetcheva ; November 2001

[20] "Optimized Link State Routing Protocol", <draft-ietf-manet-olsr-05.txt>, T. Clausen, P. Jacquet et al., 31 October 2001.

[21] Perkins, C.E. and Bhagwat, P., "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers", ACM SIGCOMM, vol.24, no.4, October 1994.

[22] Q.931, “Digital Subscriber Signalling System No. 1(DSS 1) – ISDN User-Network interface Layer 3 Specification for Basic Call Control”, 03/93

[23] New Release on Bluetooth/802.15 Relationship “Bluetooth spec serves as foundation for the IEEE 802.15.1 Standard”, Ian Grifford. http://grouper.ieee.org/groups/802/15/pub/2000/Mar00/00094r1P802-15_WG-Bluetooth-802-15-Relationship.doc

Annexe A

A) Les différents états

Dans Bluetooth il y a deux principaux états "**ATTENTE**" (STANDBY), "**CONNEXION**" (CONNECTION) et sept sous états "**page**", "**page scan**", "**inquiry**" (enquête), "**inquiry scan**", "**master response**", "**slave response**" et "**inquiry response**" (cf. fig. A1).
Les sous états sont des états intermédiaires utilisés pour ajouter des nouveaux esclaves à une piconet.

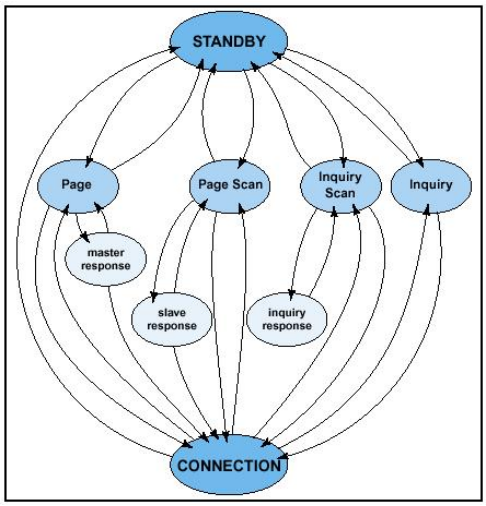

Figure A1: Diagramme des différents états [3].

A.1) Etat Standby

C'est l'état par défaut dans Bluetooth. Dans cet état, l'équipement BLUETOOTH fonctionne en mode faible consommation. Il ne participe à aucune piconet et attend d'être réveillé par le maître ou par un autre équipement bluetooth. Il peut passer à un des états suivant : *Page, Page Scan, Inquiry* ou *Inquiry Scan* comme indiqué dans la figure 7.
De l'état *Page Scan* il peut passer à l'état 'Connexion' en tant qu 'esclave, par contre s'il passe de l'état *Inquiry Scan* il est considéré comme maître.

A.2) Etat Connexion

Dans cet état là le maître et l'esclave s'échangent des paquets d'informations. On distingue quatre modes :

1 - mode actif : l'esclave participe activement sur le canal.
2 - mode sniff : l'activité d'écoute de l'esclave peut être réduite suite à la demande de ce dernier ou du maître. Son adresse de membre actif est gardée dans ce mode.
3 - mode hold : les liens ACL sont mis en suspension ce qui permet de libérer de la capacité. Alors que les liens SCO continueront à être supporté. Dans ce mode l'esclave garde son adresse de membre actif (AM_Addr).
4 - mode park : les esclaves sont toujours synchronisés au canal de la piconet mais ne participe pas. Ce mode est utilisé pour connecter plus de sept esclaves par piconet. Seul sept sont actifs mais en faisant des échanges entre esclaves actifs et les esclaves *park*s, on peut virtuellement avoir, grâce à ces nouvelles adresses, aux alentours de 255 esclaves.

A.3) Les sous-états

Les sous-états *page* et *inquiry* servent à l'établissement de nouvelles connections.

Le sous-état *inquiry* (enquête) est utilisée pour découvrir les unités BLUETOOTH qui sont à portées, ainsi que leurs adresses et horloges. Le sous-état *inquiry scan* sert à écouter s'il y a des messages de *Inquiry* contenant son code d'accès d'enqête. Dans le cas positif il n'y a qu'une seule réponse *Inquiry response* qui revient de l'esclave. Le maître se contente de lire ces réponses (inquiry response) entre les messages *Inquiry* qu'il envoie.

Avec la procédure *page* on peut établir une connexion. Le sous-état *page scan* permet à l 'esclave d'écouter s'il y a des messages de *scan* contenant son code d'accès. Quand le message est reçu par l'esclave, une grossière synchronisation se fait entre le maître et l'esclave. Le maître et l'esclave entrent respectivement dans le sous-état *master response* et *slave response* afin d'échanger des informations vitales pour l'établissement de la connexion.

B) Les différents types de paquets

Les données sur le canal de la piconet sont envoyées par paquets. Le paquet standard est composé de trois champs tels que présentés dans la figure A2.

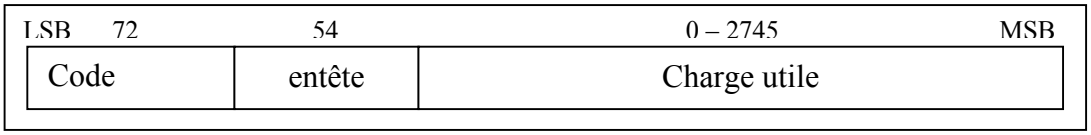

Figure A2: Format du paquet Standard.

Il existe trois grands types de paquets dans Bluetooth:

1- les paquets de contrôle : Sont utilisés pour gérer les connexions entre les terminaux bluetooth.
2- les paquets SCO : Ces paquets correspondent aux communications synchrones.
3- les paquets ACL. Ces paquets correspondent aux transferts de données asynchrones

Dans chacun de ces types de paquets plusieurs sous-catégories sont distinguées :

- **Les paquets DV** (Data voice), qui portent à la fois des données et de la parole.
- **Les paquets DMx** (Data Medium) pour les paquets ACL en mode asynchrone avec une encodage permettant la correction des erreurs en ligne. La valeur de x, qui vaut 1, 3, 5, indique la longueur du paquet en nombre de slots.
- **Les paquets DHx** (Data High) pour les paquets ACL en mode asynchrone sans correction des erreurs permettant ainsi un débit effectif plus élevé. Comme précédemment, x indique la longueur du paquet.
- **-Les paquets HVy** (High quality Voice) pour les paquets SCO en mode synchrone sans correction d'erreur (sans CRC (Cyclic Redundancy Check). Ces paquets ne sont jamais retransmis. La valeur y indique le type de contrôle d'erreur dans le paquet. Si y = 1, un FEC (Forward Error Correction) de 1/3 est utilisé. Dans ce cas, le corps du paquet contient une redondance par l'émission de trois fois la même information. Si y = 1, un FEC de 2/3 est utilisé. Dans ce cas, on transforme a l'aide d'un code la suite d'éléments binaires à transmettre de façon à détecter et corriger les erreurs. Si y = 3, aucune correction n'est utilisée.

Annexe B

A) Le protocole CSMA/CA

Dans un WLAN 802.11, la détection des collisions est impossible du fait de ce qu'on appelle le problème "near/far". Pour détecter une collision, une station doit être capable de transmettre et d'écouter en même temps. Or, dans les systèmes radio, il ne peut y avoir transmission et écoute simultanées. Pour prendre en compte cette différence, le standard IEEE 802.11 fait appel à un protocole légèrement modifié, baptisé CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance).

Ce protocole CSMA/CA fonctionne de la manière suivante : une station qui souhaite émettre explore les ondes et, si aucune activité n'est détectée, attend un temps aléatoire avant de transmettre si le support est toujours libre. Si le paquet est intact à la réception, la station réceptrice émet un accusé de reception ACK qui, une fois reçue par l'émetteur, met un terme au processus. Si la trame ACK n'est pas détectée par la station émettrice (parce que le paquet original ou le paquet ACK n'a pas été reçu intact), une collision est supposée et le paquet de données est retransmis après attente d'un autre temps aléatoire.

Autre problème de la couche MAC, spécifique au sans fil, celui du "nœud caché", où deux stations situées de chaque côté d'un point d'accès peuvent entendre toutes les deux une activité du point d'accès, mais pas de l'autre station, problème généralement lié aux distances ou à la présence d'un obstacle. Pour résoudre ce problème, le standard 802.11 définit sur la couche MAC un protocole optionnel de type RTS/CTS (Request to Send/Clear to Send). Lorsque cette fonction est utilisée, une station émettrice transmet un RTS et attend que le point d'accès réponde par un CTS. Toutes les stations du réseau peuvent entendre le point d'accès, aussi le CTS leur permet-il de retarder toute transmission prévue, la station émettrice pouvant alors transmettre et recevoir son accusé de réception sans aucun risque de collision. Du fait que le protocole RTS/CTS ajoute à la charge du réseau en réservant temporairement le support, il est généralement réservé aux plus gros paquets, dont la retransmission s'avérerait lourde du point de vue de la bande passante.

Enfin, la couche MAC 802.11 offre deux autres caractéristiques de robustesse : les sommes de contrôle CRC et la fragmentation des paquets. Pour chaque paquet, une somme de contrôle est calculée et rattachée afin d'assurer que les données n'ont pas été corrompues durant leur transit. La fragmentation des paquets permet de casser les gros paquets en unités de plus petite taille lorsqu'ils sont transmis par radio, ce qui s'avère particulièrement utile dans les environnements très congestionnés ou lorsque les interférences posent problème, puisque les gros paquets courent plus de risque d'être corrompus

Annexe C

A- Les Objets COPS

Dans ce paragraphe nous allons détailler brièvement les 16 objets du protocole COPS. Pour plus de détails sur les objets se référer au RFC 2748.

1- Objet Handle
L'objet « Client – Handle » identifie une requête particulière d'un PEP pour un type de client. Il est unique. C'est le PEP qui choisit la valeur de cet objet. Il apparaît dans les messages de requête (REQ), de décision (DEC), de rapport (RPT) et d'effacement de requête (DRQ).

2- Objet Context
Cet objet spécifie l'événement qui a déclenché la requête. Il est requis pour les messages requête (REQ).

3- Objet In-Interface (IN-Int)
Cet objet est utilisé pour identifier :

- l'interface d'entrée sur laquelle est adressé une requête particulière.
- l'adresse de provenance de ce message.

Dans COPS-RSVP, cet objet est encapsulé dans le message de requête (REQ). Par contre, il n'est jamais utilisé dans COPS-PR.

4- Objet Out-Interface (OUT-Int)
Cet objet est utilisé pour identifier l'interface de sortie sur laquelle la requête particulière s'applique ainsi que l'adresse de destination (vers où le message doit être envoyer).

5- Objet Reason
Cet Objet spécifie la raison pour laquelle la requête à été effacée. Il est encapsulé dans le message d'effacement de requête (DRQ)

6- Objet Decision
L'objet décision représente les décisions prisent par le PDP. Ces objets apparaissent dans les messages de décisions (DEC). Il existe cinq types d'objets décision :

- Decision Flags (Obligatoire)
- Decision Stateless Data (facultative)
- Decision Replacement Data (facultative)
- Client Specific Decision Data (facultative)
- Named Decision Data (facultative)

L'objet *Decision Flags* est obligatoire dans tous les messages de décisions (DEC), les autres sont facultatis. Il indique au PEP s'il faut installer ou enlever des informations de configurations. Les quatre autres objets sont facultatifs et leur longueur est variable. Le format de ces objets dépend du contexte et du type de client.

Par exemple, dans COPS-RSVP, l'objet *decision Stateless Data* est encapsulé dans le message de décision (DEC) pour indiquer le niveau de priorité du flux alors que l'objet *Decision Replacement Data* est encapsulé dans le message de décision (DEC) pour transporter des objets RSVP à remplacer dans la requête RSVP originale avant de l'envoyer à l'interface de sortie. Par contre, pour COPS-PR, l'objet *Named Decision Data* est encapsulé pour indiquer des PRIDs (Policy Rule Identifier) que le PEP doit enlever ou installer pour répondre aux demandes de configurations.

7- Objet LPDP Decision
Cet objet est utilise lorsque le LPDP (Local Policy Enforcement Point) du PEP prend des décisions. Il apparaît dans les messages de requête (REQ).

8- Objet Error
Cet Objet permet d'identifier les erreurs. Il est encapsulé dans les messages de décisions (DEC) et fermeture de Client (CC).

9- Objet ClientSI (Client Specific Information)
Il existe deux types d'objet *ClientSI* : *Signaled ClientSI* et *Named ClientSI.* Ces deux objets sont de longueurs variables et facultatives. Ils sont encapsulés, si nécessaire, dans les messages de requête (REQ), de rapport (RPT) et d'ouverture (OPN). Ils contiennent des informations spécifiques sur le type de client. Par exemple, dans COPS-RSVP l'objet *Signaled ClientSI* est encapsulé dans le message de requête (REQ) pour transporter au PDP les objets RSVP du message RSVP arrivant. Alors que, COPS-PR encapsule l'objet *Named ClientSI* dans le message de requête (REQ) pour transporter au PDP des paramètres configurables au routeur Diffserv.

10- Objet Keep Alive Timer (KA Timer)
Cet objet spécifie l'intervalle de temps maximal entre l'envoie ou réception de deux messages COPS. Il est utilisé dans les messages d'acceptation de client (CAT).

11- PED Identification Object (PEPID)
Cet objet identifie de façon unique le PEP dans le domaine de politique. Il n'est encapsulé que dans les messages d'ouverture de client. Le PEPID pourrait être, par exemple, l'adresse IP ou le nom de DNS attribué.

12- Objet Report Type
Cet objet sert à informer le PDP du résultat de l'application de la décision ou à transporter des informations de comptabilité au PDP. Il est encapsulé dans le message de rapport (RPT).

13- Objet PDP Redirect Address (PDPRedirAddr)
Cet objet peut être utilisé par le PDP dans le message de fermeture de client (CC) pour orienter le PEP vers un autre PDP.

14- Objet Last PDP Address (LastPDPAddr)
Cet objet est utilisé par le PEP pour spécifier au nouveau PDP, le dernier PDP avec qui il s'était connecté avec succès depuis sa dernière mise en marche. Il est utilisé dans le message d'ouverture de client (OPN). Si aucun PDP n'a été utilisé depuis on dernier démarrage alors cet objet ne sera pas inclus dans le message OPN.

15- Objet Accounting Timer (AcctTimer)
Cet objet, si encapsulé par le PDP dans le message d'acceptation de client (CAT), permet au PDP de contrôler la quantité de messages de rapport générée.

16- Objet Integrity
C'est un objet facultatif qui est utilisé pour la sécurité. Il permet d'authentification et de valider l'intégrité du message COPS.

B- Les messages COPS

Dans ce paragraphe nous allons détailler brièvement les 10 messages du protocole COPS. Pour plus de détails sur les messages voire se référer au RFC 2748.

1- Message Request (REQ)
Le message requête *request* est un message envoyé du PEP au PDP pour demander une décision politique.

2- Message Decision (DEC)
Le message décision est envoyé du PDP au PEP en réponse à une requête du PEP ou pour une mise à jour des décisions déjà installée. Ce message peut contenir un ou plusieurs objets décisions qui sont relatifs à un même objet contexte. La décision prise par la PDP dépend du contexte et du type de client.

3- Message Report State (RPT)
Ce message est généré par le PEP à chaque fois qu'il reçoit un message de décision. IL est utilisé par le PEP pour informer le PDP du succès ou de l'échec de l'application de la décision. D'autre part, il sert aussi, en fonction du type de client, à transporter au PDP des informations périodique associées à la comptabilité.

4- Message Delete Request State (DRQ)
Ce message est utilise par le PEP pour indiquer au PDP que l'état de requête identifié par le *Client Handle* n'existe plus.

5- Message Synchronize State Request (SSQ)
Ce message permet au PDP de demander au PEP de renvoyer un ou tous ses états actifs.

6- Message Client Open (OPN)
Ce message est utilisé par le PEP pour ouvrir une connexion avec le PDP pour un type de client. Ce message contient le type de client que le PEP supporte, l'adresse du dernier PDP qui a été utilisé pour ce type de client.

7- Message Client Accept (CAT)
Ce message est envoyé par le PDP en réponse au message *Client Open*, si le PDP accepte l'ouverture de la connexion.

8- Message Client Close (CC)
Ce message permet au PEP/PDP de notifier l'autre que ce type de client n'est plus supporté.

9- Message Keep Alive Timer
Ce message permet de vérifier que la connexion fonctionne.

10- Message Synchronize State Complete (SSC)
Ce message est envoyé par le PEP au PDP pour le notifier que la synchronisation des états est terminée. Ce message est généré suite au message *Synchronize State Request*.

C- Les six nouveaux objets de COPS-PR

Dans ce paragraphe nous allons expliquer brièvement les six nouveaux objets ajouter par COPS-PR. Pour plus de détails sur ces objets se référer au RFC 3084.

b. Objet Provisioning Instance Identifier (PRID)
Cet objet est utilisé pour transporter un identificateur de l'instance de provisionnement ou PRID.

c. Objet Prefix PRID (PPRID)
Cet objet n'est utilisé dans le protocole COPS que pour des opérations de retrait de décisions en masse qui utilisent des préfixes.

d. Objet Encoded Provisioning Instance Data (EPD)
Cet objet est utilisé pour transporter des valeurs d'instances de provisionnement codées.

e. Objet Global Provisionning Error (GPERR)
Cet Objet Il permet de communiquer des erreurs qui ne sont pas décrites dans une spécifique PRC.

f. Objet PRC Class Provisionning Error Object (CPERR)
Cet objet permet de communiquer les erreurs liées à des PRCs spécifiques et doit avoir un objet d'erreur PRID associé.

g. Objet Error PRID (ErrorPRID)
Cet objet permet de transporter des l'identificateur d'une instance de provisionnement ou le PRID qui a causé une erreur d'installation ou qui n'a pas pu être installer ou enlever.