

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

***Réseaux de micro-contrôleurs à faible
consommation d'énergie embarquant des capteurs :
Premières expériences et développement d'une
nouvelle interface paramétrable et programmable***

Yannis Mazzer — Bernard Tourancheau

N° 6450

Février 2008

Thème COM



***rapport
technique***

**Réseaux de micro-contrôleurs à faible
consommation d'énergie embarquant des
capteurs :
Premières expériences et développement
d'une nouvelle interface paramétrable et
programmable**

Yannis Mazzer , Bernard Tourancheau

Thème COM — Systèmes communicants
Équipe-Projet GRAAL

Rapport technique n° 6450 — Février 2008 — 35 pages

Résumé : Ce document est un rapport technique, sur la mise en place d'un réseau sans fil de capteurs, basé sur la norme 6LoWPAN en utilisant la solution Archrock. Il décrit aussi l'implémentation d'une interface supplémentaire qui se greffe à l'interface web Archrock.

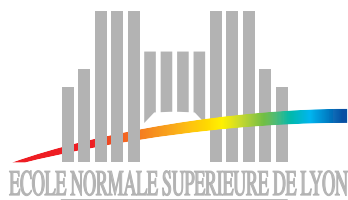
Mots-clés : mote, capteur, réseau de capteur, sans fil, archrock, webaccess, 6LoWPAN, IPv6

micro-chips network with sensors.

Abstract: This document is a technical report, dealing with the creation of a wireless sensor network, based on the 6LoWPAN standard using the Archrock system. It describes the implementation of Webaccess, an add-on interface to the Archrock web interface.

Key-words: node, sensor, sensors network, wireless, archrock, webaccess, 6LoWPAN, IPv6

Réseaux de micro-contrôleurs à faible
consommation d'énergie embarquant des
capteurs :
Premières expériences et développement d'une
nouvelle interface paramétrable et
programmable



INRIA



LIP UMR 5668 CNRS-ENS-INRIA-UCB
ENS-Lyon, 46 allée d'Italie 69364 Lyon Cedex 07

M. Yannis Mazzer
Sous la direction de M. Bernard Tourancheau

13 février 2008

Table des matières

1	Présentation des plate-formes de micro-contrôleurs	6
1.1	Existant Hardware	6
1.1.1	Atmel	6
1.1.2	Texas Instrument	7
1.1.3	Sun Microsystems	7
1.2	Existant Software dans le domaine des micro-contrôleurs en réseau	8
1.2.1	Zigbee	8
1.2.2	TinyOS	11
1.3	Existant Réseau	12
1.3.1	WiFi	12
1.3.2	LRPAN ou WPAN	13
1.3.3	Réseau Scatternet	15
1.3.4	Mesh Routing	15
1.3.5	Ad-hoc On Demand Distance Vector	16
1.3.6	ANA6	18
1.4	Internet Protocol	18
1.4.1	IPv4	18
1.4.2	IPv6	20
1.5	6LoWPAN	22
1.6	L'approche Archrock	22
1.6.1	L'approche hardware	22
1.6.2	L'approche software	24
1.6.3	Interface SOAP	24
2	Webaccess, une interface web pour exploiter efficacement le monitoring via Archrock	
2.1	Architecture	26
2.2	Réalisation	27
2.3	Tests et performances	27
3	Perspectives d'une approche plus ouverte pour le monitoring	31
3.1	Architecture	31
3.2	Architecture d'une implémentation de 6LoWPAN open-source	32

Introduction

Du WiFi[33], utilisant la norme 802.11[4], au bluetooth, 802.15.5[21], en passant par les réseaux GSM[24], les réseaux wireless sont aujourd'hui omniprésents dans la vie de tous les jours. Cependant ses utilisateurs doivent faire face à certaines contraintes techniques qui dans certains cadres d'utilisation se révèlent rédhibitoires.

Effectivement, l'utilisation des systèmes sans-fil, notamment le WiFi, nécessite une puissante source d'énergie. Par conséquent, son utilisation n'est pas envisageable dans les systèmes embarqués. Les constructeurs de téléphones portables, par exemple, ont choisi de développer des batteries de plus en plus performantes, comme les batteries au Lithium (Li-Ion, Li-Po/Li-Ion[23]), mais aussi de plus en plus chères, pour s'adapter au réseau GSM.

Dans une politique d'économie d'énergie qui intègre les préoccupations environnementales, cette stratégie n'est peut-être pas la meilleure. Dans l'élaboration d'un réseau sans-fil de capteurs à moindre coût, il n'est clairement pas possible de participer à la course à la puissance. La solution est donc de réduire la consommation des appareils embarqués en utilisant un autre protocole réseau, et en optimisant les temps d'éveil des capteurs.

Une solution a déjà été envisagée par une communauté de chercheurs. Ce projet, intitulé 6LoWPAN[1] n'est toutefois qu'à un stade de recherche et développement et est loin d'être achevé. Celui-ci a pour but d'appliquer les qualités de la norme IPv6[32] et du mesh routing[25][18] à un réseau basse consommation de capteurs de norme physique 802.15.4[20].

Effectivement, si l'on veut greffer les réseaux de capteurs à l'internet, il faut prévoir une identification pour chacun d'entre eux. Quoi de mieux donc que l'IPv6 qui permet d'adresser chaque atome de l'univers? On peut ainsi déployer des réseaux massifs de capteurs sans se soucier du problème de l'adressage. Solution qui s'avère d'autant plus utile lorsque l'on commence à parler de mesh routing. En effet, le réseau maillé permet d'avoir une route principale pour accéder au serveur principal, mais aussi des routes "fallback". Cela favorise non seulement la mobilité de l'unité, mais aussi le mesh routing lui-même, plus il y a de routes vers la destination (gateway) plus la connectivité de ces réseaux sera bonne.

1 Présentation des plate-formes de micro-contrôleurs

Le mot "mote" est dérivé de l'anglais "node", fréquemment utilisé pour désigner un membre d'une architecture réseau. Dans la communauté des développeurs de réseaux de capteurs, ce mot est aussi employé pour désigner un membre d'une architecture réseau, cependant comme ce réseau n'est, dans notre cas, constitué que de capteurs, le mot "mote" désigne alors un capteur. Mais de quoi est donc constitué ce capteur ? Encore un abus de langage, un capteur désigne un composant électronique capable de mesurer une grandeur physique. Ici, le capteur est en fait un petit ordinateur embarqué alimenté par des piles ou des batteries et capable d'interagir avec son environnement. Il est donc composé d'un micro-contrôleur qui va gérer les entrées et sorties des périphériques, tels que l'antenne, les capteurs (électroniquement parlant), les DELs, etc ...

1.1 Existant Hardware

Quelques firmes commencent à commercialiser des motes, certaines, qui existent depuis la naissance des Intel[12] 486, d'autres, qui sont plus jeunes et se spécialisent dans les réseaux de capteurs. Aujourd'hui trois grandes firmes sont présentes sur ce créneau : Atmel[5], fabricant réputé de micro-contrôleurs, MoteIV[17], fabricant de systèmes embarqués, et Sun Microsystems[13], entreprise qu'il n'est plus nécessaire de présenter.

1.1.1 Atmel



FIG. 1 – Atmel mote

Atmel propose une plate-forme de mote basée sur les ATmegaX, rendus très populaires par leur faible prix, avec une largeur de bus de 8 bit, une mémoire de 4k, et un espace de stockage flash de 128k. Contrairement à d'autres plate-formes, le chipset radio n'est pas intégré au micro-contrôleur, on y trouve donc l'intérêt de pouvoir choisir son chipset selon la consommation, la portée et le budget. Toutefois, l'ajout de modules d'extension comme des capteurs de pression ou encore des Leds supplémentaires est limité, on compte seulement 8 canaux digitaux 10 bit.

1.1.2 Texas Instrument



FIG. 2 – Archrock mote

Le micro-contrôleur Texas Instrument[11] MSP430 sert de base à la plate-forme produite par la société MoteIV (désormais appelée Sentilla). Ses performances sont largement supérieures à celles de l'ATmega, et il intègre par défaut un chipset radio 802.15.4. La largeur de bus est de 16 bit, sa mémoire est de 10k, et son espace de stockage est de 48k. La tension de ce chipset seul ne dépasse pas les 3,6V (maximum!) pour une fréquence de 8Mhz. Le MSP430 permet notamment de brancher 4 extensions binaires, 4 extensions analogiques, une autre antenne (très utile pour étendre le réseau à plusieurs centaines de mètres), et permet aussi d'alimenter, dans la mesure du raisonnable, ces extensions à partir de la batterie de la mote.

1.1.3 Sun Microsystems

Le géant de l'informatique Sun Microsystems a lui aussi un projet de recherche et développement de réseaux de senseurs. La différence majeure outre les caractéristiques hardware grandement supérieures aux autres (cf. plus bas), réside dans la facilité de programmation. En effet, aujourd'hui tout le monde connaît la puissance des machines virtuelles java, c'est pourquoi Sun Microsystems a choisi d'utiliser une machine virtuelle adaptée plutôt que d'utiliser les deux principaux softwares existants (TinyOS[8] et Zigbee[2]). Il



FIG. 3 – SunSPOT

faut tout de même avouer qu'avec un micro-contrôleur ARM920T cadencé à 180MHz en 32bit, 512k de mémoire et 4M de stockage, il est plus facile de faire tourner une machine java. En outre, la gestion de l'alimentation est faite par un Atmel Atmega88, ce qui réduit la charge du contrôleur principal. De plus, non-content d'outrepasser les concurrents en terme de puissance et d'ergonomie, SunSpot[14] dispose de 7 extensions binaires et de 6 extensions analogiques.

1.2 Existant Software dans le domaine des micro-contrôleurs en réseau

1.2.1 Zigbee

Zigbee est une Alliance industrielle spécialisée dans les réseaux de monitoring sans fil à faible coût énergétique.

Applications La spécification initiale de ZibBee propose un protocole lent dont le rayon d'action est relativement faible, mais dont la fiabilité est assez élevée, le prix de revient faible et la consommation considérablement réduite.

On retrouve donc ce protocole dans des environnements embarqués où la consommation est un critère de sélection. Ainsi, la domotique et les nombreux capteurs qu'elle implémente apprécient particulièrement ce protocole en plein essor et dont la configuration du réseau maillé se fait automatiquement en fonction de l'ajout ou de la suppression de noeuds. On retrouve aussi Zigbee dans les contrôles industriels, les applications médicales, les détecteurs de fumée et d'intrusion.

Les noeuds sont conçus pour fonctionner plusieurs mois (jusqu'à deux ans pour les moins consommant) en autonomie complète grâce à une pile alcaline de 1,5 V.

Présentation de la stack Zigbee L'essence même d'un protocole est de donner un cadre de fonctionnement à une communication. Ainsi, l'IEEE définit dans le cadre de sa norme IEEE 802.15.4 le cadre de ce protocole pour les couches basses (physique et mac). Il est donc nécessaire d'implémenter les couches de plus haut niveau (réseau et applicatif dans notre cas) afin que ce modèle soit parfaitement fonctionnel. Les protocoles Zigbee peuvent donc fonctionner, en théorie, sur plusieurs supports mac mais sont en général présentes pour le média 802.15.4. C'est la Zigbee Alliance qui s'occupe de cette partie du protocole en fournissant une stack (ou pile) de référence. Celle-ci est réservée aux membres de l'alliance qui doivent l'implémenter dans leurs solutions. Si l'on désire mettre en place un dispositif pouvant se connecter à un réseau Zigbee, il faudra suivre l'un des trois cas suivants :

- Soit faire partie de la Zigbee Alliance et donc bénéficier de ses apports technologiques, notamment concernant cette stack de communication. Il faut savoir que l'inscription à cette structure coûte environ \$3500 pour une entreprise.
- Soit reprendre un produit développé par l'un des membres de la Zigbee Alliance et disposer de la stack, spécifique à ce produit, développée par le constructeur choisi.
- Soit développer sa propre stack en accord avec les dernières spécifications disponibles. Cela représente bien entendu un travail très important qui doit s'adapter au système (hardware et software) retenu pour mettre en place le dispositif. Si ce développement est effectué à des fins commerciales, il devra être validé par la Zigbee Alliance.

Routage au niveau réseau Au niveau de la couche réseau, le routage est soit direct, soit indirect. Le routage est direct lorsqu'un dispositif voulant transmettre des données connaît l'adresse réseau du destinataire. Cette adresse est donc transmise dans la trame pour atteindre et agir sur le dispositif prévu. Dans le cas contraire, le routage indirect se fait lorsqu'un dispositif ne connaît pas l'adresse du destinataire. Un équipement de type routeur ou coordonnateur fait la relation avec le vrai destinataire d'après la table de routage et la table de découvertes des routes. Un dispositif qui n'a pas les capacités de routages (LED) doit router les données suivant le routage hiérarchique (il remonte l'arbre). La table de routage contient les données sur les destinataires. Il s'agit de l'adresse de destination de la route et le prochain dispositif à atteindre pour se rapprocher du destinataire. La table de découverte d'une route contient les informations sur les sources du message. Elle stocke l'adresse d'origine du dispositif qui a fait la demande et l'adresse du dispositif qui va transmettre les données en tant qu'intermédiaire (entre

la source et la destination). Elle contient aussi les coûts de transmission entre la source jusqu'au noeud actuel et du noeud jusqu'au destinataire. Elle peut donc adapter la route pour être plus performante en mettant à jour les adresses à utiliser. Le choix d'une route, lorsque plusieurs routeurs en parallèle relaient l'information, se fait par rapport au routeur (ou coordinateur). Lors de la demande de création de route, la table va recevoir plusieurs demandes à partir de la même adresse d'origine. Elle va alors comparer les coûts de transmission pour choisir le chemin ayant le coût le plus faible. L'algorithme de routage suggéré par la Zigbee Alliance pour les réseaux maillés est AODV (Ad hoc On-Demand Vector Routing)[30]. C'est un protocole de routage dit "réactif" : une route est établie uniquement sur demande. L'avantage est qu'il ne charge pas le trafic.

Liaison au niveau applicatif La connexion au niveau applicatif se fait grâce à la table de liaison, contenu dans le coordinateur ou un routeur. Les liaisons permettent de créer des liens logiques entre des dispositifs d'application complémentaires et des éléments de fins (capteurs). La table de liaison permet aussi d'associer à un attribut d'un dispositif en entrée plusieurs attributs de dispositifs en sortie ou l'inverse. La table de liaison est implémentée dans le coordinateur Zigbee. Le choix de ce dispositif vient du fait que le coordinateur Zigbee est nécessaire au réseau. Le second intérêt est que comme le coordinateur est indispensable au réseau, il doit être (en général) alimenté par le secteur. Ces deux raisons font que la table de liaison sera toujours accessible. La table de liaison se repose sur trois critères normalisés par la Zigbee Alliance :

1. **Un profile** permet de créer une application inter-opérable et distribuée. Il s'agit donc de définir des formats de messages et le traitement des actions pour permettre à des dispositifs de demander, transmettre des données et savoir les interpréter. Les profiles sont développés par les entreprises pour permettre de répondre à des besoins spécifiques. Par exemple, le premier profile existant est fait pour gérer les lampes et des interrupteurs (home control lighting). Ce profile permet six types d'échanges de messages de contrôle. Les profiles permettent de créer aussi une norme autour de chaque application pour permettre l'interopérabilité des systèmes.
2. **Les clusters** sont associés avec des flots de données entrant ou sortant. Les identificateurs de clusters sont uniques dans un profile. Les clusters permettent de lier deux dispositifs par l'association d'un cluster en entrée et d'un cluster en sortie en supposant qu'ils appartiennent au même profile. En fait deux dispositifs sont liés s'ils partagent le même

besoin (côté récepteur) et la même ressource (côté émetteur). La table de liaison (binding table) contient pour chaque cluster un identifiant pour le définir (sur 8 bits) et l'adresse des deux dispositifs (source et destination).

3. **Un attribut** définit un capteur ou un actionneur. C'est l'élément qui décrit de façon la plus précise l'utilisation du dispositif (par exemple un capteur de mouvement, un buzzer, une lampe, etc.). La table de liaison est la couche applicative qui permet de gérer la table de routage et la table de découverte de routes. C'est elle qui va permettre d'associer le relevé d'un capteur sur un dispositif à une action spécifique sur un autre dispositif à travers toutes les couches du protocole Zigbee. C'est une façon de simplifier l'accès lorsque le réseau contient beaucoup de connexions et de dispositifs : la reconnaissance entre les dispositifs qui dialoguent se fait par rapport à leurs familles (les profiles et clusters) et leurs qualités (les attributs) communes.

1.2.2 TinyOS

Présentation TinyOS est un système d'exploitation open-source conçu pour des réseaux de capteurs sans-fil. Il respecte une architecture basée sur une association de composants, réduisant la taille du code nécessaire à sa mise en place. Cela s'inscrit dans le respect des contraintes de mémoires qu'observent les réseaux de capteurs. Pour autant, la bibliothèque de composant de TinyOS est particulièrement complète puisqu'on y retrouve des protocoles réseaux, des pilotes de capteurs et des outils d'acquisition de données. L'ensemble de ces composants peut être utilisé tel quel, il peut aussi être adapté à une application précise. En s'appuyant sur un fonctionnement événementiel, TinyOS propose à l'utilisateur une gestion très précise de la consommation du capteur et permet de mieux s'adapter à la nature aléatoire de la communication sans fil entre interfaces physiques.

Disponibilité et sources : TinyOS est un système principalement développé et soutenu par l'université américaine de Berkeley, qui le propose en téléchargement sous la licence BSD et en assure le suivi. Ainsi, l'ensemble des sources sont disponibles pour de nombreuses cibles matérielles.

Event-driven : Le fonctionnement d'un système basé sur TinyOS s'appuie sur la gestion des événements se produisant. Ainsi, l'activation de tâches, leur interruption ou encore la mise en veille du capteur s'effectue à l'apparition d'événements, ceux-ci ayant la plus forte priorité. Ce fonctionnement événementiel (event-driven) s'oppose au fonctionnement dit

temporel (time-driven) où les actions du système sont gérées par une horloge donnée.

Langage : TinyOS a été programmé en langage NesC[6]

Préemptif : Le caractère préemptif d'un système d'exploitation précise si celui-ci permet l'interruption d'une tâche en cours. TinyOS ne gère pas ce mécanisme de préemption entre les tâches mais donne la priorité aux interruptions matérielles. Ainsi, les tâches entre elles ne s'interrompent pas mais une interruption peut stopper l'exécution d'une tâche.

Temps réel : Lorsqu'un système est dit "temps réel", celui-ci gère des niveaux de priorité dans ses tâches permettant de respecter des échéances données par son environnement. Dans le cas d'un système strict, aucune échéance ne tolère de dépassement contrairement à un système temps réel mou. TinyOS se situe au-delà de ce second type car il n'est pas prévu pour avoir un fonctionnement temps réel.

Consommation : TinyOS a été conçu pour réduire au maximum la consommation en énergie du capteur. Ainsi, lorsqu'aucune tâche n'est active, il se met automatiquement en veille.

TinyDB TinyDB est un système d'exécution de requêtes sur un réseaux de motes, intégré à TinyOS. Son interface est proche de l'interface SQL, et permet d'accéder aux données sans avoir à coder une pseudo-base de données ponctuelle. TinyDB a été codé dans la même optique d'économie d'énergie que TinyOS, et ne se contente pas seulement de stocker les données, mais les traite, les filtre, les regroupe, et les envoie sur le réseau périodiquement. Une API java, permet à partir d'un PC de récupérer directement ces données et de les traiter très aisément.

1.3 Existant Réseau

1.3.1 WiFi

La norme IEEE 802.11 est un standard international décrivant les caractéristiques d'un réseau local sans fil (WLAN). Le nom wifi correspond initialement au nom donné à la certification délivrée par la WECA (Wireless Ethernet Compatibility Alliance), l'organisme chargé de maintenir l'interopérabilité entre les matériels répondant à la norme 802.11. Par abus de langage (et pour des raisons de marketing) le nom de la norme se confond aujourd'hui avec le nom de la certification (c'est du moins le cas en France, en Espagne, au Canada et aux États-Unis). Ainsi un réseau wifi est en réalité un réseau répondant à la norme 802.11. Dans d'autres pays (en Allemagne

par exemple) de tels réseaux sont correctement nommés WLAN. Grâce au wifi, il est possible de créer des réseaux locaux sans fil à haut débit. Dans la pratique, le wifi permet de relier des ordinateurs portables, des machines de bureau, des assistants personnels (PDA), des objets communicants ou même des périphériques à une liaison haut débit (de 11 Mbit/s en 802.11b à 54 Mbit/s en 802.11a/g et 540 Mbit/s pour le futur 802.11n) sur un rayon de plusieurs dizaines de mètres en intérieur (généralement entre une vingtaine et une cinquantaine de mètres). Dans un environnement ouvert, la portée peut atteindre plusieurs centaines de mètres voire dans des conditions optimales plusieurs dizaines de kilomètres (pour la 'variante' WIMAX ou avec des antennes directionnelles). Ainsi, des fournisseurs d'accès internet commencent à irriguer des zones à forte concentration d'utilisateurs (gares, aéroports, hôtels, trains, etc.) avec des réseaux sans fil connectés à Internet. Ces zones ou point d'accès sont appelées bornes wifi et en anglais "hot spots". Les iBooks d'Apple, Inc. furent, en 1999, parmi les premiers ordinateurs grand public à proposer un équipement wifi intégré (sous le nom d'AirPort), bientôt suivis par le reste de la gamme. à partir de 2003, on voit aussi apparaître des modèles de PC portables bâtis autour de la technologie Intel Centrino, qui leur permettent une intégration similaire. Les autres modèles de PC doivent encore s'équiper d'une carte d'extension adaptée (PCMCIA, USB, Compact Flash, SD, PCI, MiniPCI, etc.).

Structure La norme 802.11 s'attache à définir les couches basses du modèle OSI pour une liaison sans fil utilisant des ondes électromagnétiques, c'est-à-dire :

- la couche physique (notée parfois couche PHY), proposant trois types de codage de l'information
- la couche liaison de données, constituée de deux sous-couches :
 - le contrôle de la liaison logique (Logical Link Control, ou LLC)
 - le contrôle d'accès au support (Media Access Control, ou MAC).

La couche physique définit la modulation des ondes radioélectriques et les caractéristiques de la signalisation pour la transmission de données, tandis que la couche liaison de données définit l'interface entre le bus de la machine et la couche physique, notamment une méthode d'accès proche de celle utilisée dans le standard Ethernet et les règles de communication entre les différentes stations. La norme 802.11 propose en réalité trois couches physiques, définissant des modes de transmission alternatifs :

- DSSS (Direct-Sequence Spread Spectrum)
- FHSS (Frequency-Hopping Spread Spectrum)

- IR (Infra-Red)

Il est possible d'utiliser n'importe quel protocole de transport sur un réseau sans fil wifi au même titre que sur un réseau Ethernet.

1.3.2 LRPAN ou WPAN

Le standard LRPAN ou WPAN se décompose en différentes normes :

- IEEE 802.15.1 définit le standard Bluetooth 1.x permettant d'obtenir un débit de 1 Mbit/sec
- IEEE 802.15.2 propose des recommandations pour l'utilisation de la bande de fréquence 2.4 GHz (fréquence utilisée également par le WiFi). Ce standard n'est toutefois pas encore validé
- IEEE 802.15.3 est un standard en cours de développement visant à proposer du haut débit (20 Mbit/s).
- IEEE 802.15.4 est un standard pour des applications telles que Zigbee et PAN. Il possède deux modes, CSMA et Beacon, et deux niveaux, FFD et RFD.

Les éléments fondamentaux d'un produit 802.15 sont définis dans les deux premières couches protocolaires, la couche radio et la couche bande de base. Ces couches prennent en charge les tâches matérielles comme le contrôle du saut de fréquence et la synchronisation des horloges.

La couche radio (la couche la plus basse) est gérée au niveau matériel. C'est elle qui s'occupe de l'émission et de la réception des ondes radio. Elle définit les caractéristiques telles que la bande de fréquence et l'arrangement des canaux, les caractéristiques du transmetteur, de la modulation, du receveur, etc. La bande fréquences utilisée est disponible au niveau mondial et s'étend sur 83,5 MHz (de 2,4 à 2,4835 GHz). Cette bande est divisée en 79 canaux séparés de 1 MHz. Le codage de l'information se fait par sauts de fréquence. La période est de 625 μ s, ce qui permet 1 600 sauts par seconde.

Il existe 3 classes de modules radio Bluetooth sur le marché ayant des puissances différentes et donc des portées différentes :

Classe	Puissance	Portée
1	100mW (20dBm)	100 mètres
2	2,5mW (4dBm)	15 à 20 mètres
3	1mW (0 dBm)	10 mètres

La bande de base (ou baseband en anglais) est également gérée au niveau matériel. C'est au niveau de la bande de base que sont définies les adresses matérielles des périphériques (équivalent à l'adresse MAC d'une carte réseau). Cette adresse est nommée BD_ADDR (Bluetooth Device Address) et est

codée sur 48 bits. Ces adresses sont gérées par la IEEE Registration Authority. C'est également la bande de base qui gère les différents types de communication entre les appareils. Les connexions établies entre deux appareils Bluetooth peuvent être synchrones ou asynchrones. La bande de base peut donc gérer deux types de paquets :

- les paquets SCO (Synchronous Connection-Oriented)
- les paquets ACL (Asynchronous Connection-Less)

Réseau Piconet

Un piconet est un mini réseau qui se crée de manière instantanée et automatique quand plusieurs périphériques Bluetooth sont dans un même rayon. Un piconet suit une topologie en étoile : 1 maître / plusieurs esclaves.

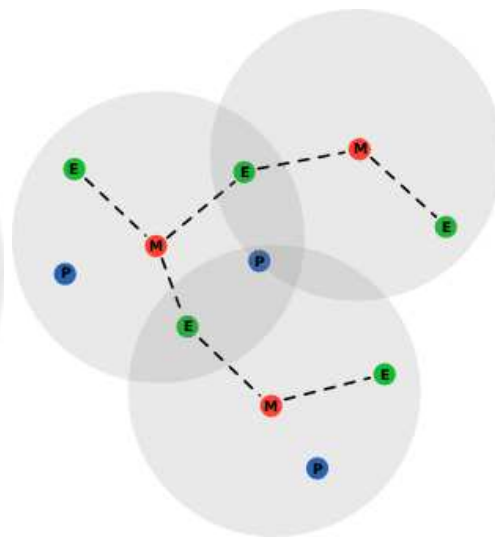
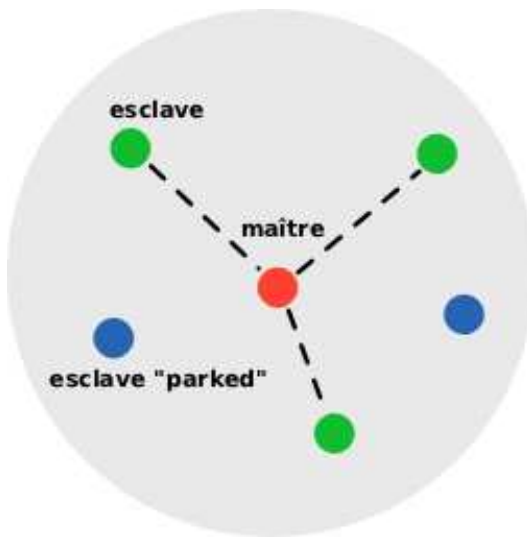


FIG. 4 – Schéma du réseau Piconet FIG. 5 – Schéma du réseau Scatternet

La communication est directe entre le maître et un esclave. Les esclaves ne peuvent pas communiquer entre eux. Tous les esclaves du piconet sont synchronisés sur l'horloge du maître. C'est le maître qui détermine la fréquence de saut pour tout le piconet.

1.3.3 Réseau Scatternet

Les périphériques esclaves peuvent avoir plusieurs maîtres, les différents piconets peuvent donc être reliés entre eux. Le réseau ainsi formé est appelé un scatternet.

1.3.4 Mesh Routing

La topologie Mesh (terme anglais signifiant maille ou filet), qualifie les réseaux (filaires ou non) dont tous les hôtes sont connectés de proche en proche sans hiérarchie centrale, formant ainsi une structure en forme de filet. Cela permet d'éviter d'avoir des points sensibles, qui en cas de panne, coupent la connexion d'une partie du réseau. Si un hôte est hors service, ses voisins passeront par une autre route.

Cette architecture est issue de la recherche militaire.

C'est une technologie de rupture comparée aux solutions centralisées classiques sans-fil avec station de base. Les solutions "Mesh" autorisent un déploiement rapide et simplifié, une grande évolutivité de la couverture et, de par leur maillage, une forte tolérance aux pannes et aux interférences, réduisant significativement les coûts d'installation et d'exploitation des réseaux. Ces solutions reproduisent l'architecture de l'Internet tout en l'optimisant pour le sans-fil. L'implémentation d'une telle topologie est appelée réseau maillé.

Les spécifications du standard international "Mesh" OLSR (Optimized link state routing protocol - RFC 3626[10]) de l'Internet Engineering Task Force (IETF) ont été produites par l'INRIA dans le cadre ouvert du groupe de travail Mobile Ad hoc NETwork (MANET) de l'IETF.

OLSR permet de plus le maintien de la connectivité avec une montée à l'échelle du nombre de hôtes en mobilité, la compatibilité avec différentes technologies radio (Wi-Fi, WiMax, ?) et filaires (PLC, fibre, Ethernet) du fait de son positionnement au niveau IP (Internet Protocol) ainsi que l'optimisation en sans fil mobile : Peer-to-Peer (P2P) & Machine-to-Machine (M2M).

1.3.5 Ad-hoc On Demand Distance Vector

L'algorithme de routage AODV (Ad hoc On Demand Distance Vector) est un protocole de routage destiné aux réseaux dit mobiles.

AODV est capable à la fois de routage unicast et multicast[15]. C'est un algorithme à la demande c'est-à-dire qu'il ne construit des routes entre les noeuds que lorsqu'elles sont demandées par des noeuds sources. Il maintient ces routes aussi longtemps que les sources en ont besoin. De plus, AODV constitue des arborescence connectant les membres des groupes multicast. Les arbres sont composés des membres des groupes et des noeuds nécessaires pour connecter les membres. AODV utilise un numéro de séquence pour assurer la fraîcheur des routes. Il est libre de boucle, auto-démarrant et s'accommode d'un grand nombre de noeuds mobiles (ou intermittents).

AODV construit les routes par l'emploi d'un cycle de requêtes route request / route reply. Lorsqu'un noeud source désire établir une route vers une destination pour laquelle il ne possède pas encore de route, il broadcast un paquet route request (RREQ) à travers le réseau. Les noeuds recevant le paquet mettent à jour leur information relative à la source et établissent des pointeurs de retour vers la source dans les tables de routage. Outre l'IP de la source, le numéro de séquence courant et l'ID de broadcast, le RREQ contient également le numéro de séquence de la destination le plus récent connu de la source. Un noeud recevant un RREQ émettra un paquet route reply (RREP) soit si il est la destination, soit si il possède une route vers la destination avec un numéro de séquence supérieur ou égal à celui repris dans le RREQ. Si tel est le cas, il envoie (unicast) un paquet RREP vers la source. Sinon, il rebroadcast le RREQ. Les noeuds conservent chacun trace des IP sources et des ID de broadcast des RREQ. Si ils reçoivent un RREQ qu'ils ont déjà traité, ils l'écartent et ne le transmettent pas.

Les noeuds établissent des pointeurs de propagation vers la destination alors que les RREP reviennent vers la source. Une fois que la source a reçu le RREP, elle peut commencer à émettre des paquets de données vers la destination. Si, ultérieurement, la source reçoit un RREP contenant un numéro de séquence supérieur ou le même mais avec un compteur de hop plus petit, elle met à jour son information de routage vers cette destination et commencera à utiliser la meilleure route.

Une route est maintenue aussi longtemps qu'elle continue à être active. Une route est considérée active tant que des paquets de données transitent périodiquement de la source à la destination selon ce chemin. Lorsque la source stoppe d'émettre des paquets de données, le lien expire et est alors effacé des tables de routages des noeuds intermédiaires. Si le lien se rompt alors qu'une route est active, le noeud extrémité du lien rompu émet un paquet route error (RERR) vers le noeud source pour le notifier de la destination désormais inatteignable. Après réception de RERR, si la source désire toujours la route, elle peut réinitier un processus de découverte de route.

Les routes multicast sont établies de la même manière. Un noeud désireux de rejoindre un groupe multicast diffuse (broadcast) un RREQ avec l'IP de destination positionnée à celle du groupe multicast et le flag J (join) positionné pour indiquer la volonté de rejoindre le groupe. Tout noeud membre de l'arborescence multicast recevant cet RREQ et ayant un numéro de séquence récent est susceptible d'émettre un RREP. Les noeuds établissent des pointeurs de routage alors que les RREP voyagent vers la source. A la réception des RREP, la source retient la route avec le numéro de séquence le plus récent ou à défaut avec le plus petit compte de hops. Après une période donnée de découverte, le noeud source émettra un message d'activation mul-

unicast (MACT) au hop suivant sélectionné. Ce message active la route. Un noeud ayant préparé un routage multicast qui ne recevrait pas de MACT dans la période spécifiée expirera et effacera le pointeur. Si un noeud recevant le MACT ne fait pas partie de l'arborescence, il aura à conserver la trace de la meilleure route pour les RREP reçus. Dès lors il doit également émettre unicast un MACT vers son prochain hop et ainsi de suite jusqu'à atteindre un noeud de l'arborescence.

AODV maintient les routes aussi longtemps que celles-ci sont actives. Ceci inclut la maintenance d'une arborescence multicast aussi longtemps que le groupe est actif. Du fait de l'intermittence des noeuds, il est plus que probable que de nombreuses pertes de liens auront lieu le long de la route durant sa durée de vie. Les publications en annexe étudient comment gérer ces ruptures de liens. Le papier WMCSA décrit AODV sans multicast et inclut les résultats détaillés d'une simulation d'un réseau d'un millier de noeuds. Le papier Mobicom décrit les opérations multicast d'AODV et détaille des simulations qui illustrent son comportement correct. Le draft internet contient la spécification tant du routage unicast que du multicast. Il mentionne également comment la Qualité de Service (QoS) et l'agrégation de sous-réseaux peuvent être utilisés en conjonction avec AODV. Enfin le papier IEEE 'Personal Communications' et le papier Infocom détaillent une étude approfondie de simulations comparant AODV et DSR (dynamic source routing protocol) et envisage les forces et faiblesses respectives des deux protocoles.

1.3.6 ANA6

ANA6[7] est une nouvelle architecture d'adressage IPv6 pour micro-contrôleurs qui intègre le support des réseaux ad-hoc spontanés et qui permet une fusion entre ceux-ci et les réseaux sans fils. ANA6 est basé sur le concept d'identification des motes, qui s'avère être une base du fonctionnement des réseaux ad-hoc. Il se situe au niveau 2.5 de la couche OSI, entre le niveau 2 (MAC) et le niveau 3 (IP). Il offre une vision "Ethernet" au niveau 3, ce qui permet de réutiliser un niveau 3 classique. De plus, ANA6 introduit une structure d'adressage et définit un routage en dehors du niveau 3.

1.4 Internet Protocol

1.4.1 IPv4

L'Internet Protocol version 4 ou IPv4[31] est la première version d'IP à avoir été largement déployée, et forme encore la base (en 2007) de l'Internet. Elle est décrite dans la RFC numéro 791 (RFC 791).

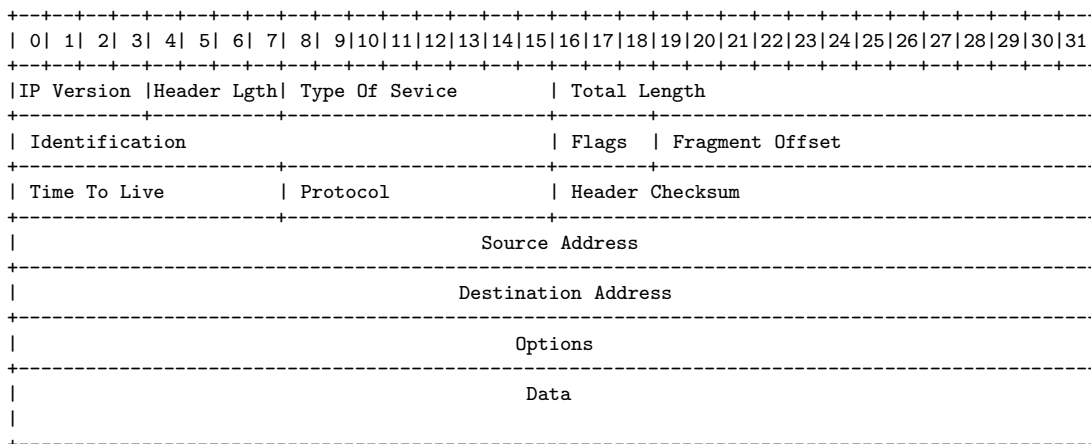


FIG. 6 – Schéma IPv4

IPv4 utilise une adresse IP sur 32 bits, ce qui est un facteur limitant à l'expansion d'Internet puisque "seulement" 4 294 967 296 adresses sont possibles. Cette limitation conduit à la transition d'IPv4 vers IPv6, actuellement en cours de déploiement, qui devrait progressivement le remplacer. Cette limitation est pour l'instant contournée grâce à l'utilisation de techniques de translation d'adresses NAT ainsi que par l'adoption du système CIDR.

Représentation d'une adresse IPv4 Le plus souvent, une adresse IP est représentée sous la forme de quatre nombres décimaux séparés par des points comme par exemple 193.43.55.67. Chacun des nombres représente un octet. Comme un octet est composé de 8 bits, cela fait 32 bits.

Fragmentation Une trame a une taille maximale, appelée Maximum Transmission Unit ou MTU. Lorsque la longueur du paquet (datagramme) est supérieure, l'information sera fragmentée. La taille maximum supportée par IPv4 (car codée sur 16 bits) est de 64ko mais les réseaux ne prennent pas en charge de trames de telles longueurs, typiquement, on a MTU de l'ordre de 1ko(Arpanet), 1,5ko(Ethernet) etc...

Pour reconstituer l'information, on utilise les champs suivants dans l'entête IPv4 :

- Version (4 bits) : Version d'IP utilisée. Ici, 4.
- Longueur de l'entête (4 bits) : Nombre de mots de 32 bits, soit 4 octets. (Ou nombre de lignes du schéma.) La valeur est comprise entre 5 et 15, car il y a 20 octets minimum et on ne peut dépasser 40 octets d'option. (Soit en tout, 60 octets.)

- Type de service (8 bits) : Rarement utilisé. Ce champ permet de distinguer différentes qualité de service différenciant la manière dont les paquets sont traités. Composé de 3 bits de priorité (donc 8 niveaux) et trois indicateurs permettant de différencier le débit, le délai ou la fiabilité.
- Longueur totale en octets (16 bits) : Nombre total d'octets du datagramme, entête IP comprise. Donc, la valeur maximale (est 2^{16})-1 octets.
- Identification (16 bits) : Numéro permettant d'identifier les fragments d'un même paquet.
- Flag (3 bits) :
 1. (Premier bit) actuellement Inutilisé.
 2. (Deuxième bit) DF (Don't Fragment) : Lorsque ce bit est positionné à 1, il indique que le paquet ne peut pas être fragmenté. Si le routeur ne peut acheminer ce paquet (taille du paquet supérieure à la MTU), il est alors rejeté.
 3. (Troisième bit) MF (More Fragments) : Quand ce bit est positionné à 1, on sait que ce paquet est un fragment de données et que d'autres doivent suivre. Quand il est à 0, soit le fragment est le dernier, soit le paquet n'a pas été fragmenté.
- Fragment offset (13 bits) : Position du fragment par rapport au paquet de départ, en nombre de mots de 8 octets.
- Durée de vie ou TTL Time To Live (8 bits) : initialisé par l'émetteur, ce champ est décrémenté d'une unité généralement à chaque saut de routeur. Quand TTL = 0, le paquet est abandonné et un message ICMP est envoyé à l'émetteur pour information.
- Protocole (8 bits) : Numéro du protocole au-dessus de la couche réseau : TCP = 6, UDP = 17, ICMP = 1.
- Somme de contrôle de l'entête ou Checksum ou encore CRC pour Contrôle de Redondance Cyclique (16 bits) : Vérification de l'intégrité de l'entête seulement. Si le CRC est invalide, le paquet est abandonné sans message d'erreur.
- Adresse source (32 bits) : Adresse IP de l'émetteur sur 4 octets ou 32 bits.
- Adresse destination (32 bits) : Adresse IP du récepteur sur 4 octets ou 32 bits.
- Options (0 à 40 octets ou 0 à 320 bits par mots de 32 bits ou 4 octets) : facultatif.

- Bourrage : de taille variable comprise entre 0 et 7 bits. Il permet de combler le champ option afin d'obtenir une entête IP multiple de 32 bits. La valeur des bits de bourrage est 0.

1.4.2 IPv6

Pourquoi un nouveau protocole IP ? Le protocole IPv4 permet d'utiliser un peu plus de quatre milliards d'adresses différentes pour connecter les ordinateurs et les autres appareils reliés au réseau. Du temps des débuts d'Internet, quand les ordinateurs étaient rares, cela paraissait plus que suffisant. Il était pratiquement unimaginable qu'il y aurait un jour suffisamment de machines sur un unique réseau pour que l'on commence à manquer d'adresses disponibles.

Une grande partie des quatre milliards d'adresses IP théoriquement disponibles ne sont pas utilisables, soit parce qu'elles sont destinées à des usages particuliers (par exemple, le multicast), soit parce qu'elles appartiennent déjà à des sous-réseaux importants. En effet, d'immenses plages de 16,8 millions d'adresses, les réseaux dits de classe A, ont été attribuées aux premières grandes organisations connectées à Internet, qui les ont conservées jusqu'à aujourd'hui sans parvenir à les épuiser. Les Nord-Américains, et dans une moindre mesure les Européens, se sont partagé les plus grandes plages d'adresses, relativement peu nombreuses, tandis que les régions connectées plus tardivement, comme l'Amérique du Sud et surtout l'Asie, sont restées sur la touche.

En conséquence, il y a aujourd'hui, principalement en Asie, une pénurie d'adresses que l'on doit compenser par des mécanismes comme la Traduction d'adresse et de port réseau (NAPT) et l'attribution dynamique d'adresses, et en assouplissant le découpage en classes des adresses (CIDR).

Au vu de l'importance et de la croissance d'Internet, cette situation pose de plus en plus de problèmes, le Japon par exemple possède un nombre très limité d'adresses IPv4. Il est de plus prévisible que la demande d'adresses Internet va augmenter dans les années à venir, même dans les régions du monde épargnées jusqu'ici, suite à des innovations comme les téléphones mobiles (et bientôt, sans doute, les automobiles et divers appareils) connectés à Internet.

C'est principalement en raison de cette pénurie, mais également pour résoudre quelques-uns des problèmes révélés par l'utilisation à vaste échelle d'IPv4, qu'a commencé en 1995 la transition vers IPv6. Parmi les nouveautés essentielles, on peut citer :

- l'augmentation de 2^{32} (soit environ 10^{10}) à 2^{128} (soit environ 10^{38}) du nombre d'adresses disponibles

- des mécanismes de configuration et de renumérotation automatique très importants
- IPsec, QoS et le multicast implémentés nativement
- la simplification des en-têtes de paquets, qui facilite notamment le routage.

Adresses IPv6 Une adresse IPv6 est longue de 16 octets, soit 128 bits, contre 4 octets (32 bits) pour IPv4. On dispose ainsi d'environ $3,4 \times 10^{38}$ adresses, soit 340 282 366 920 938 463 463 374 607 431 768 211 456, soit encore, pour reprendre l'image usuelle, plus de 667 132 000 milliards ($6,67 \times 10^{17}$) d'adresses par millimètre carré de surface terrestre.

On abandonne la notation décimale pointée employée pour les adresses IPv4 (par exemple 172.31.128.1) au profit d'une écriture hexadécimale, où les 8 groupes de 16 bits sont séparés par un signe deux-points :

1fff :0000 :0a88 :85a3 :0000 :0000 :ac1f :8001

La notation canonique complète ci-dessus comprend exactement 39 caractères.

Les 64 premiers bits de l'adresse IPv6 (préfixe) servent généralement à l'adresse de sous-réseau, tandis que les 64 bits suivants identifient l'hôte à l'intérieur du sous-réseau : ce découpage joue un rôle un peu similaire aux masques de sous-réseau d'IPv4.

Différentes sortes d'adresses IPv6 jouent des rôles particuliers. Ces propriétés sont indiquées par le début de l'adresse, appelé préfixe.

L'Internet IPv6 est défini comme étant le sous-réseau 2000 : :/3 (les adresses commençant par un 2 ou un 3). Seules ces adresses peuvent être routées. Toutes les autres adresses ne peuvent être utilisées que localement sur un même réseau physique (de niveau 2), ou par un accord privé de routage mutuel. Parmi les adresses de 2000 : :/3, on distingue :

- Les adresses 6to4 (2002 : :/16) permettant d'acheminer le trafic IPv6 via un ou plusieurs réseaux IPv4
- Les adresses du 6bone (3ffe : :/16) pour l'expérimentation des interconnexions de réseaux IPv6. (Le 6bone n'est plus opérationnel depuis le 06 juin 2006)

1.5 6LoWPAN

6LoWPAN, alias IPv6 over Low power Wireless Personal Area Networks, est le nom d'un groupe de travail créé afin de développer un réseau de communication pour des systèmes embarqués à puissance limitée, sur la couche physique 802.15.4. Effectivement, Des différences majeures sont remarquables entre un réseau IPv6 standard et IEEE802.15.4 :

- IPv6 définit une MTU (Maximum Transmission Unit) de 1280 bytes, tandis qu'un paquet IEEE802.15.4 ne compte que 127 octets. Il est donc nécessaire d'implémenter une fragmentation de chaque paquet IPv6 avant de l'envoyer sur la couche 802.15.4, et naturellement, d'implémenter la défragmentation à la réception.
- Sur un réseau IPv6, des machines dédiées procèdent à l'acheminement des paquets à travers le réseau hiérarchiquement organisé. 6LoWPAN prévoit un réseau maillé à l'instar du protocole IRC (Internet Relay Chat), ainsi, chaque noeud pourra être programmé pour rediffuser les paquets qui ne lui sont pas adressés.
- L'ordre des priorités n'est pas le même, l'économie d'énergie et de lignes de code reste la principale préoccupation des développeurs 6LoWPAN. A contrario, Les performances et la gestion des problèmes caractéristiques – comme la congestion – sont primordiales pour un réseau IPv6.

1.6 L'approche Archrock

La société Archrock[3] basée à Los Angeles, propose une solution permettant l'exploitation des motes Telosb. Cette solution est composée d'un Gateway (hardware), qui sert de passerelle entre le réseau principal ethernet, et le réseau maillé 802.15, pour lequel il est aussi serveur DNS et DHCP, et d'un middleware sous forme de portail web.

1.6.1 L'approche hardware

côté gateway Le gateway est largement fourni au niveau hardware, il ne risque pas d'avoir de fortes latences par rapport aux tâches qui lui incombent. De plus l'espace de stockage est suffisant pour garder précieusement plusieurs années de mesures dans la base de données, avec la fréquence et les capteurs par défaut.

Descriptif technique du gateway :

Processeur	: Intel Celeron 1.6GHz
Mémoire	: 512Mb de RAM
Disque Dur	: 40Gb
Boîtier	: ITX, 146x250x42mm

côté mote Comme décrit plus haut, les motes sont dotées d'un micro-contrôleur Texas Instrument[11] MSP430. de 10kB de RAM, et d'un chipset

radio 802.15.4 intégré. Toutes ces caractéristiques sont regroupées sous l'appellation Telosb. Appellation reprise dans TinyOS1.2.2 pour la programmation, ainsi TinyOS peut inclure les bibliothèques correspondant au hardware de la mote.

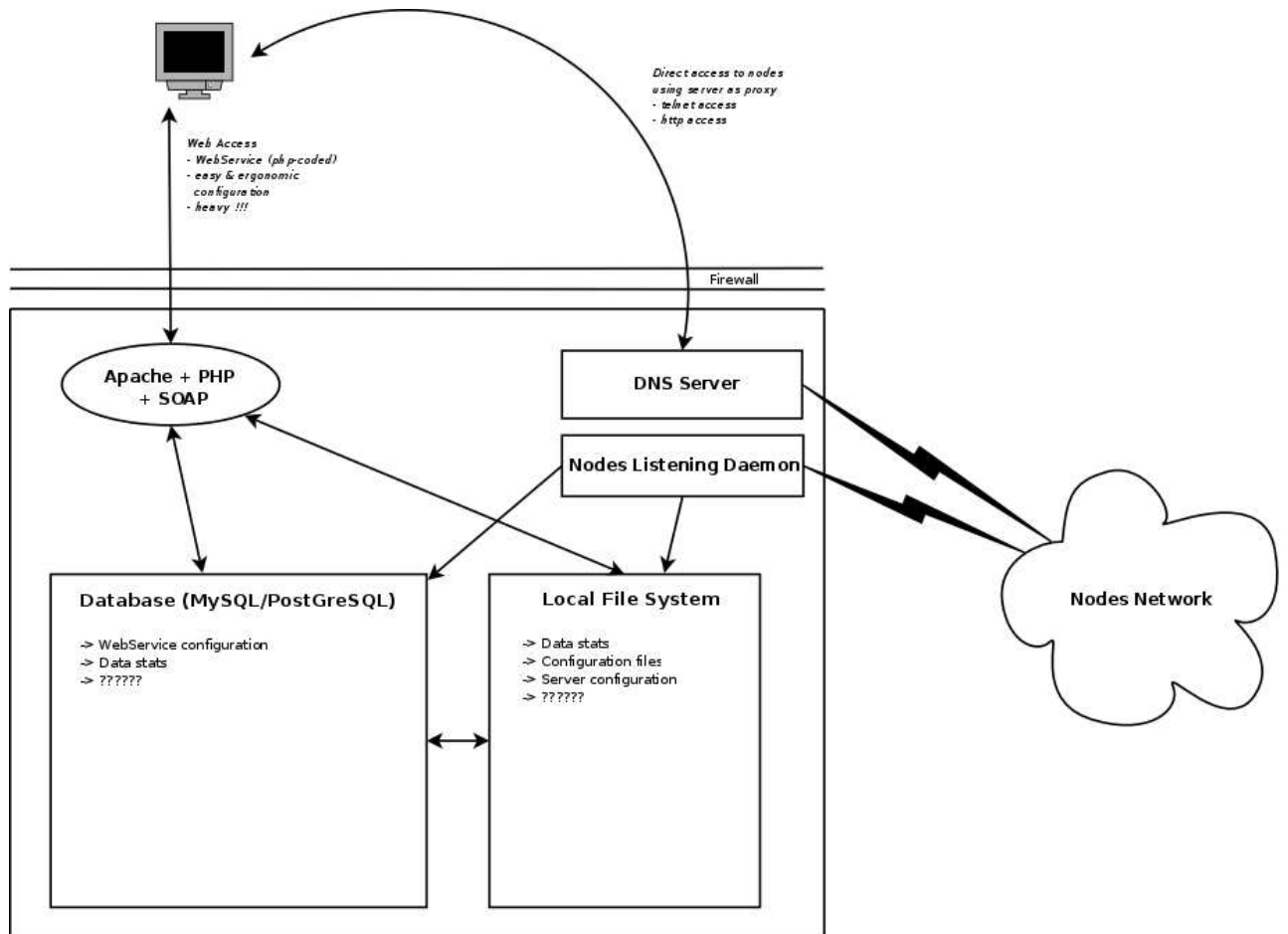


FIG. 7 – Schéma du gateway Archrock

1.6.2 L'approche software

L'interface Archrock permet d'administrer le réseaux de motes relativement aisément, en utilisant le protocole HTTP comme protocole principal, toutes les instructions sont donc prédéfinies. A cette fin, Archrock a choisi d'utiliser le protocole SOAP[27], expliqué dans la section 1.6.3. SOAP est greffé au daemons Apache et PostgreSQL, il permet en appelant une fonction dans une url de piloter le gateway, comme par exemple envoyer une

requête ping aux motes, ou encore afficher les valeurs de chaque capteur à un instant t , ou durant toute une période.

En arrière plan, tourne l'implémentation Archrock du réseau maillé et une pile IPv6 aux normes 6LoWPAN décrites plus haut.

Codée en Python pour le pilote du gateway et en NesC pour les motes, elle est complète et fonctionnelle. L'interface réseau 802.15.4 du gateway n'est ni plus ni moins qu'une mote qui bridge les informations entre son interface 802.15.4 et son port USB. Le tout est géré par GNU/Linux Fedora Core, la version gratuite de GNU/Linux Red Hat, et protégé par un IPTables. Malheureusement, l'accès réservé du système ne permet pas d'en dire plus à ce sujet.

OS	: GNU/Linux Fedora Core
Serveur Web	: Apache
Base de données	: PostGreSQL
Modules	: PHP, SOAP
Firewall	: IPTables

1.6.3 Interface SOAP

Simple Object Access Protocol (SOAP) est un protocole de RPC orienté objet bâti sur XML. Il permet la transmission de messages entre objets distants, ce qui veut dire qu'il autorise un objet à invoquer des méthodes d'objets physiquement situés sur un autre serveur. Le transfert se fait le plus souvent à l'aide du protocole HTTP, mais peut également se faire par un autre protocole, comme SMTP. Le protocole SOAP est composé de deux parties :

- une enveloppe, contenant des informations sur le message lui-même afin de permettre son acheminement et son traitement
- un modèle de données, définissant le format du message, c'est-à-dire les informations à transmettre.

SOAP a été initialement défini par Microsoft et IBM, mais est devenu une référence depuis une recommandation du W3C, utilisée notamment dans le cadre d'architectures de type SOA (Service Oriented Architecture) pour les Services Web Objets. Le protocole SOAP emploie des méta-données. Voir Using SOAP metadata[22] SOAP n'est plus un acronyme depuis la version 1.2. En effet, SOAP v1.2 a été réécrit en termes d'infosets XML, et non plus sous forme de sérialisations `<?xml... ?>` comme il l'était en v1.1. La notion d'objet (spécifiée dans Simple OBJECT Access Protocol) devient donc obsolète.

pensais rajouter des exemples d'appels soap, et leurs résultats qu'en penses-tu ?

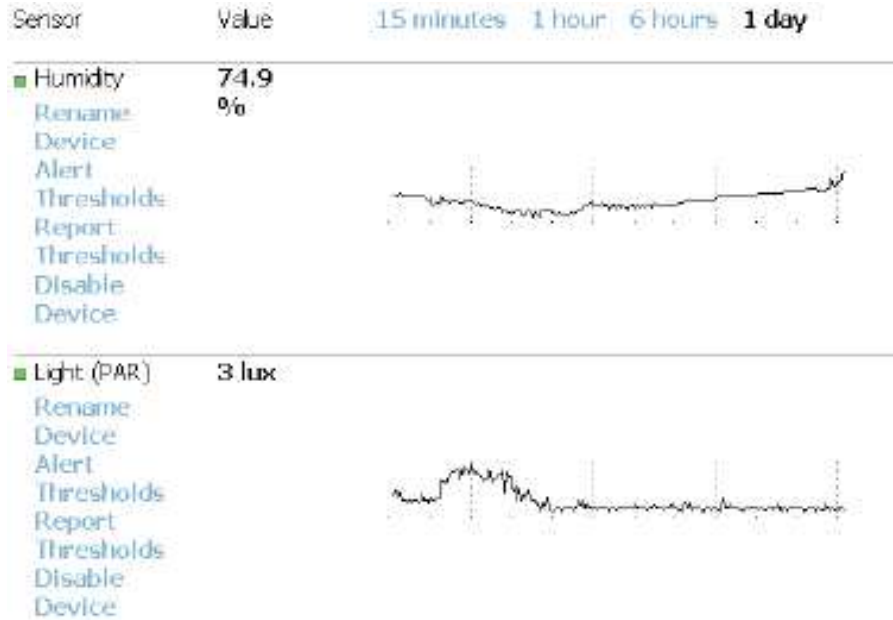


FIG. 8 – Screenshot de l’interface web Archrock

L’étude du système Archrock montre que celui-ci est créé pour une utilisation simple. Effectivement, aucun accès d’administration (tel que ssh, telnet, ...) n’est proposé. Le seul accès disponible est un accès via HTTP. Un daemon Apache sur une distribution GNU/Linux Fedora Core, sur lequel est greffé le module PHP5 très open-source le protocole SOAP, comme centre nerveux de cette solution. Ce protocole est utilisé pour accéder à la base de données, aux fichiers de configuration "principaux" du système, ou encore pour faire la liaison au niveau OSI applicatif entre le réseau principal, et le réseau de notes.

Malheureusement l’accès protégé du système interdit formellement n’importe quel ajout de modules personnels, telles que des pages web qui seraient greffées directement sur le driver de l’interface 802.15.4. L’utilisateur est obligé de créer des terminaux, ou des serveurs intermédiaires. En conclusion, une telle architecture avec un client, qui doit faire une requête HTTP sur un serveur qui va faire appel au gateway pour récupérer les informations de la base de données ou pour communiquer avec les notes, et finalement faire les calculs nécessaires et renvoyer via le protocole HTTP au client, n’est pas du tout adaptée pour une utilisation intensive.

En résumé d’une part, l’implémentation du réseau maillé en IPv6 aux normes de l’ieee 6LoWPAN est complète et fonctionnelle. Codée en python du côté

serveur, et naturellement NesC pour le côté mote, le mesh routing est bien géré, ainsi que l'IPv6 (et pas seulement les IPs courtes!). D'autre part, le choix du contrôle du Gateway via SOAP uniquement limite beaucoup les possibilités de la plateforme et introduit de la lourdeur.

2 Webaccess, une interface web pour exploiter efficacement le monitoring via Archrock SOAP

2.1 Architecture

L'interface de contrôle proposée par Archrock présente des limitations quant à l'utilisation directe des données recueillies par les motes. Nous avons décidé de coder une interface plus complète répondant à nos besoins, afin de pouvoir utiliser le système de motes à disposition, dans des études scientifiques.

Le Webaccess, est une interface Web, codée intégralement en PHP/HTML qui permet de faire des appels SOAP pour se greffer à l'interface Archrock.

Nous avons tout d'abord identifié les points faibles de l'architecture ArchRock pour notre utilisation et avons conçu et implémenté un middleware et son interface, basé sur SOAP, puisque greffé sur le système Archrock, utilisant un serveur applicatif supplémentaire.

Puis nous avons identifié ses points forts qui sont son Interface pratique pour une utilisation scientifique, des graphes clairs et exportables, une partie calculatoire optimisée, et le Multi-threading.

Webaccess permet de récupérer en temps réel les données stockées sur le gateway de Archrock et de créer des graphiques de valeurs, pour chaque mote et/ou capteur. Ces graphiques sont stockés dans un dossier au format png (Portable Network Graphics), ce qui facilite leur exportation. Cependant le gros inconvénient explicité précédemment de SOAP est sa lenteur d'exécution. Cet inconvénient ne peut être totalement supprimé. Mais nous avons codé en multi-threading les appels SOAP effectués par le Webaccess. Cela permet d'optimiser de manière considérable les temps d'attente. Ainsi, envoyer une requête ping sur le réseau maillé à chaque chargement de page n'est pas une gêne. D'autre part, la précision de l'affichage des graphiques dépend intrinsèquement de leur taille, car effectivement, on ne peut pas diviser un pixel. Il est donc nécessaire de faire des moyennes sur certaines plages de valeurs, dépendant naturellement de la durée demandée par l'utilisateur.

Dans l'optique d'une réduction des temps d'attente, l'implémentation de pré-calculs des valeurs récurrentes et de stockage de celles-ci dans une base de données, grâce à un démon, s'avère particulièrement intéressante.

2.2 Réalisation

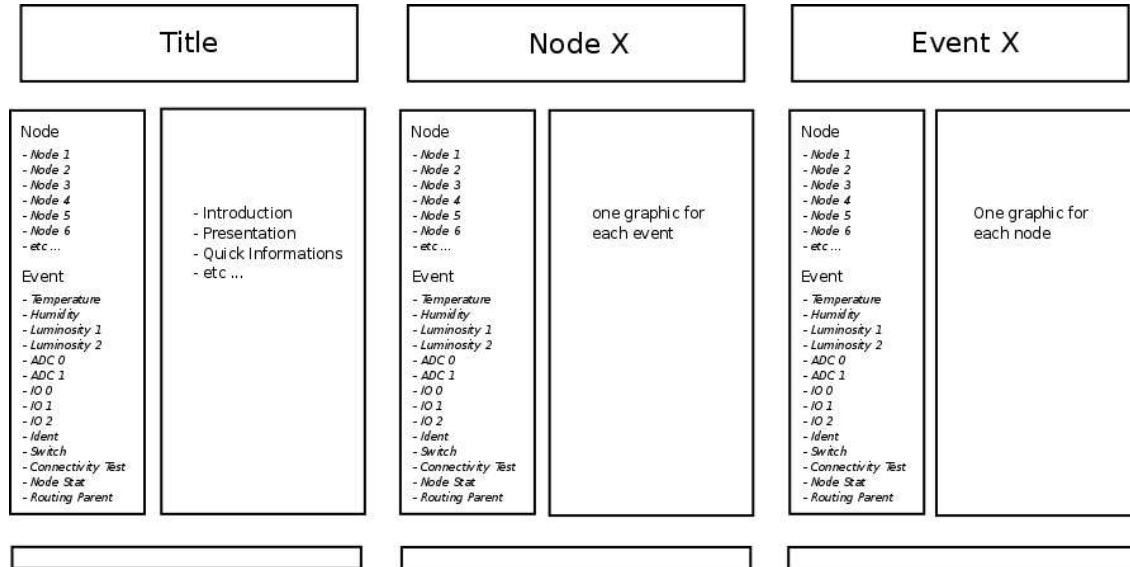


FIG. 9 – Apparence schématique du Webaccess

2.3 Tests et performances

La différence entre la version simple-threading du Webaccess et la version multi-threading est importante. Effectivement, la version simple-threading est difficilement utilisable du fait de ses temps de latence importants, explicités plus haut. Ainsi, il faudra 15 secondes pour charger une page contenant le graphe d'un capteur quelconque pour chaque mote, soit six graphes, tandis que la version multi-threading n'aura besoin que de 3,5 secondes. Malgré cette optimisation, l'utilisation de Webaccess au dessus du système Archrock n'est pas adapté à un réseau de grande envergure.

```

$req_event = mysql_query('SELECT * FROM Events');
while($row_event = mysql_fetch_assoc($req_event)) {
    $req_node = mysql_query('SELECT * FROM Nodes');
    while($row_node = mysql_fetch_assoc($req_node)) {
        if( isactivated($row_event["read"], $row_node["addr"], $proxy) == 1) {
            $offset = 0;
            $i=0;
            while(1) {
                $ret = $proxy->eventsRead($t2, $t1, $row_event["read"], $row_node["addr"], $offset);
                foreach($ret->results as $result) {
                    $value = $result->value;
                    echo '<p>Node : ' . $row_node["addr"] . ' :: Event : ' . $row_event["name"] . ' :: Value : ' . $value->{$value->name} . '</p>';
                    $i++;
                    $array_in[$i] = $value->{$value->name};
                }
                if( ($ret->offset + count($ret->results)) == $ret->total) {
                    break;
                }
                else {
                    $offset+= count($ret->results);
                }
            }
            $fileName = "./graphs/" . $row_node["addr"] . "_" . $row_event["name"] . ".png";
            grid($array_in, $fileName, $i);
            echo '';
        }
    }
}

```

FIG. 10 – Implémentation de la page event du Webaccess

La page "node" recense tous les senseurs actifs pour la node sélectionnée. Dans le même raisonnement, la page "event", recense les valeurs du senseur sélectionné pour chaque mote active.

```

echo '<p>';
$req_node = mysql_query('SELECT * FROM Nodes');
while($row_node = mysql_fetch_assoc($req_node)) {
    $req_event = mysql_query('SELECT * FROM Events');
    while($row_event = mysql_fetch_assoc($req_event)) {
        if( isactivated($row_event["read"], $row_node["addr"], $proxy) == 1) {
            $offset = 0;
            $i = 0;
            while(1) {
                $ret = $proxy->eventsRead($t2, $t1, $row_event["read"], $row_node["addr"], $offset);
                foreach($ret->results as $result) {
                    $value=$result->value;
                    echo '<p>Node : ' . $row_node["addr"] . ' :: Event : ' . $row_event["name"] . ' :: Value : ' . $value->{$value->name} . '</p>';
                    $array_in[$i] = assign($row_event["read"], $value->{$value->name});
                    $i++;
                }
                echo '<p>array_in : ' . $array_in[1] . ' </p>';
                echo '';
            }
            if( ($ret->offset + count($ret->results)) == $ret->total) {
                break;
            }
            else {
                $offset+= count($ret->results);
            }
        }
    }
}
$fileName = "./graphs/" . $row_node["addr"] . "_" . $row_event["name"] . ".png";
grid($array_in, $fileName, $i);
echo '';
}
}

```

FIG. 11 – Implémentation de la page node du Webaccess

La page "node" affiche le graphe de tous les senseurs actifs sur la mote en question, la fonction "isactivated" permet de vérifier l'état de chaque capteur avant de lui envoyer une requête.

Le résultat en images :

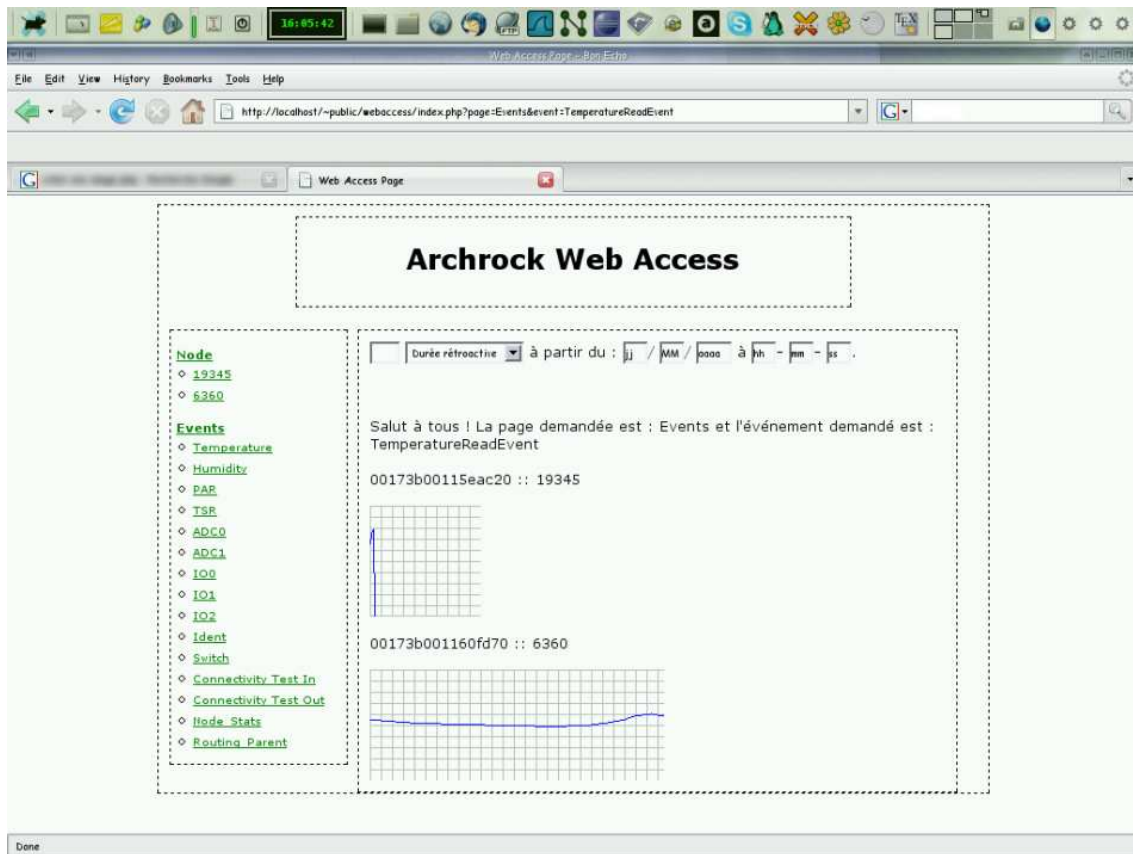


FIG. 14 – Apparence du Webaccess

3 Perspectives d'une approche plus ouverte pour le monitoring

3.1 Architecture

L'étude de la solution Archrock a montré ses limites dans le cadre d'une utilisation scientifique. C'est pourquoi il est préférable d'envisager d'élaborer une nouvelle solution plus ouverte et qui permettrait de faciliter les recherches utilisant un tel réseau de nœuds. Nous proposons la perspective d'un système proche avec quelques modifications :

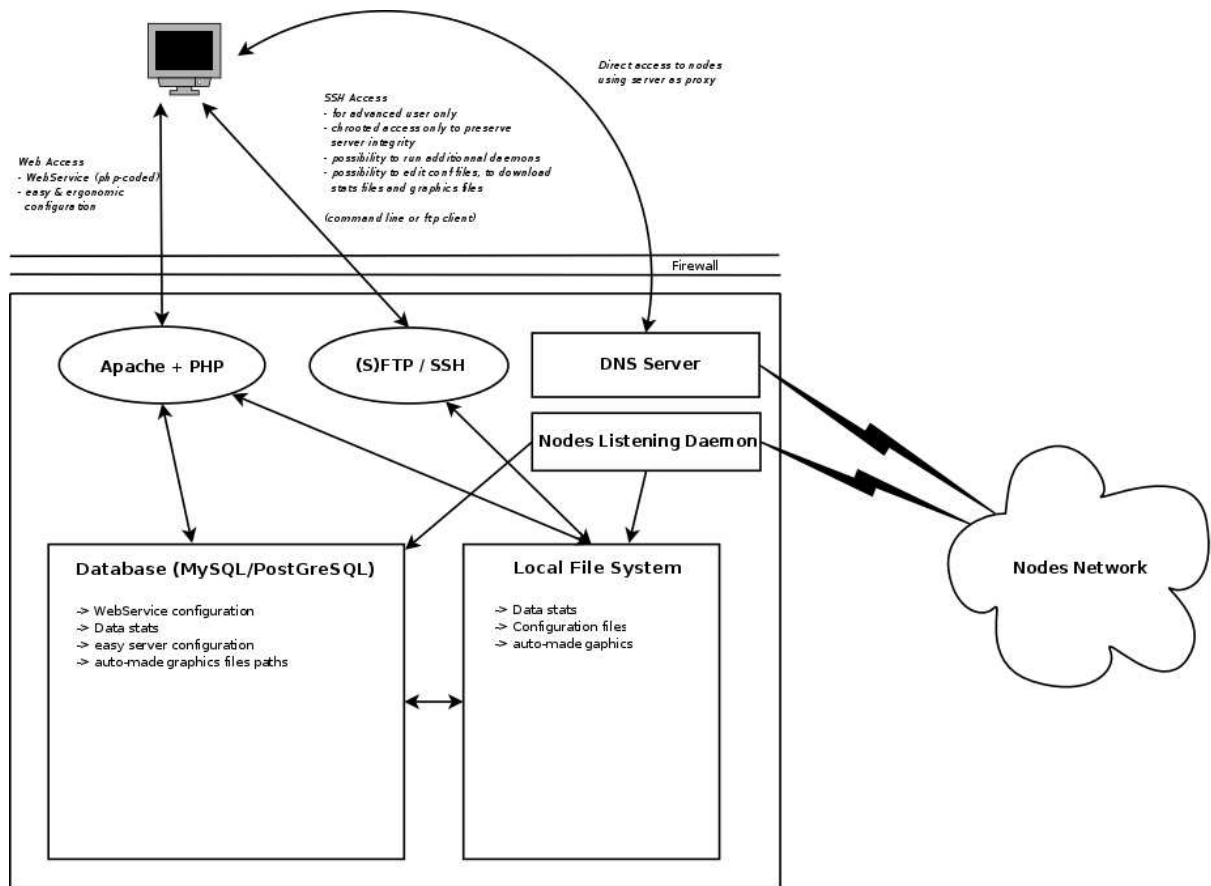


FIG. 15 – Schéma d'une approche plus ouverte

Réduire les temps d'accès est plutôt séduisant, le choix d'un portail en PHP paraît donc judicieux. Toutefois, n'ayant plus l'éventail d'opérations proposées par SOAP, il faut trouver un substitut ; SSH/SFTP sous licence

BSD semble adapté, il est effectivement sécurisé et illimité. Les options de l'utilisateur sont d'ores et déjà plus étendues, en effet, il est possible d'ajouter de nouveaux modules au système sur une seule et même machine. La configuration du système est certes moins aisée, car réservée à des connaisseurs du système Unix, mais pour une utilisation de base, une page d'administration peut remédier à cela. On retrouve une Base de Données MySQL ou PostgreSQL – les différences entre celles-ci sont si faibles, que le choix tient plus d'une préférence que d'autre chose – connectée à la fois à l'interface web et à TinyOS/TinyDB. Le portail Web contient un panel d'outils, qui, à l'instar du Webaccess, sont orientés analyse des données recueillies sur les senseurs.

3.2 Architecture d'une implémentation de 6LoWPAN open-source

L'Étudiant Allemand, Matus Harvan[9], a codé une implémentation de 6LoWPAN réduite. Le travail important qui reste à faire est donc la partie routage et la partie opérationnelle.

L'implémentation de Matus Harvan comprend à l'heure actuelle :

- Pile UDP
- réponse au ping
- fragmentation/défragmentation partielle
- short addresses
- long addresses
- mesh routing
- veille, sensing et partage des données sur le réseau

Une telle implémentation système présente de réelles difficultés car le debugging est difficile. Il n'y a aucune interface physique d'état avec les nœuds si ce ne sont les DELs. Le soucis permanent concerne la réduction de la taille du code sans pour autant omettre certains aspects de la norme 6lowpan ou du caractère opérationnel. La coordination de l'activation des extensions (modules I/O, et ADC [Analog to Digital Converter] externes) avec le portail internet d'administration et la base de données.

Difficultés de l'implémentation :

- debugging difficile, car aucune interface physique d'état avec les nœuds si ce ne sont les Leds

- soucis de réduction de la taille du code sans pour autant omettre certains aspect de la norme 6LoWPAN
- coordonner l'activation des extensions (modules I/O ADC externes) avec le portail et la base de données
- TCP

Conclusions

L'ensemble des tests effectués sur la plateforme existante a permis de faire ressortir les limites de celle-ci. Nous avons développé un système parallèle qui permet d'améliorer une grande partie de ces limitations. L'interface Webaccess permet ainsi d'exploiter au maximum les capacités du système hardware et software de motes proposé par Archrock tout en gardant le même esprit de portabilité puisqu'il n'est pas nécessaire d'installer quoi que ce soit sur la machine cliente, un navigateur web suffit.

La perspective d'un système plus ouvert et donc plus configurable, intégrant Webaccess, s'avère intéressante, car elle permettrait de supprimer les dernières limites du système actuel, et donc d'avoir un système ouvert, parfaitement fonctionnel et opérationnel.

Références

- [1] 6LoWPAN. Ipv6-based low-power wireless personal area networks. <http://6lowpan.net/>.
- [2] Zigbee Alliance. <http://www.zigbee.org/>.
- [3] Archrock. <http://www.archrock.com/>.
- [4] IEEE Standard Association. norme 802.11. <http://standards.ieee.org/getieee802/802.11.html>.
- [5] Atmel. <http://www.atmel.com/>.
- [6] UC Berkeley. Nesc webpage. <http://nesc.sourceforge.net/>.
- [7] Guillaume Chelius. Architecture et communication dans les réseaux spontanés sans-fil. <http://docinsa.insa-lyon.fr/these/2004/chelius/these.pdf>.
- [8] TinyOS Community. <http://www.tinyos.net/>.
- [9] Matus Harvan. <http://www.inf.ethz.ch/personal/mharvan/intro.html>.
- [10] IETF. Optimized link state routing protocol (olsr). <http://tools.ietf.org/html/rfc3626>.
- [11] Texas Instrument. <http://www.ti.com/>.
- [12] Intel. <http://www.intel.com/>.
- [13] Sun Microsystems. <http://www.sun.com/>.
- [14] Sun Microsystems. Sunspot project page. <http://www.sunspotworld.com/>.
- [15] Gabriel Montenegro, Benjamin Gaidioz, Pascale Primet, and Bernard Tourancheau. Equivalent differentiated services for aodvng. *Mobile Computing and Communications Review* 6, pages 110–111, 2002.
- [16] Belgique Réseau Citoyen. Aodv. <http://reseaucitoyen.be/wiki/index.php/AODV>.
- [17] Sentilla. <http://www.sentilla.com/>.
- [18] Andrew S. Tanenbaum. *Computer Networks*. PH PTR, 4th edition, 2003.
- [19] Andrew S. Tanenbaum. *Operating Systems*. Pearson, 2nd edition, 2003.
- [20] IEEE 802.15 WPAN task group 4. Norme 802.15.4. <http://www.ieee802.org/15/pub/TG4.html>.
- [21] IEEE 802.15 WPAN task group 5. Norme 802.15.5. <http://www.ieee802.org/15/pub/TG5.html>.

- [22] Sergey BERYOZKIN. Using soap metadata. <http://webservices.xml.com/pub/a/ws/2003/07/22/sessions.html?page=2>, 2003.
- [23] Wikipedia. Accumulateur lithium. http://fr.wikipedia.org/wiki/Accumulateur_lithium.
- [24] Wikipedia. Global system for mobile communications. http://fr.wikipedia.org/wiki/Global_System_for_Mobile_Communications.
- [25] Wikipedia. Mesh networking. http://en.wikipedia.org/wiki/Mesh_network.
- [26] Wikipedia. Topologie mesh. http://fr.wikipedia.org/wiki/Topologie_Mesh.
- [27] Wikipedia. Soap. <http://fr.wikipedia.org/wiki/SOAP>, 2007.
- [28] Wikipedia. Tynyos. <http://fr.wikipedia.org/wiki/TinyOS/>, 2007.
- [29] Wikipedia. Zigbee. <http://fr.wikipedia.org/wiki/Zigbee/>, 2007.
- [30] Wikipedia. Ad-hoc on-demand distance vector. <http://en.wikipedia.org/wiki/AODV>, 2008.
- [31] Wikipedia. Ipv4. <http://fr.wikipedia.org/wiki/Ipv4>, 2008.
- [32] Wikipedia. Ipv6. <http://fr.wikipedia.org/wiki/Ipv6>, 2008.
- [33] Wikipedia. Wi-fi. <http://fr.wikipedia.org/wiki/Wi-Fi>, 2008.



Centre de recherche INRIA Grenoble – Rhône-Alpes
655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Lille – Nord Europe : Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq
Centre de recherche INRIA Nancy – Grand Est : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Rennes – Bretagne Atlantique : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex
Centre de recherche INRIA Saclay – Île-de-France : Parc Orsay Université - ZAC des Vignes : 4, rue Jacques Monod - 91893 Orsay Cedex
Centre de recherche INRIA Sophia Antipolis – Méditerranée : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
<http://www.inria.fr>
ISSN 0249-0803