



Projet SR04

Rapport final

Agrégation de données dans les réseaux de capteurs

Automne 2010

XUE Yong AGUILAR Andres
GONZALEZ Andres BARROUX Mickaël

Table des manières

1.	Introduction	3
2.	Réseaux de capteurs sans fils.....	4
2.1	Présentation des réseaux de capteurs	4
2.1.1	Histoire des réseaux de capteurs	4
2.1.2	Applications	4
2.2	Architecture d'un réseau de capteurs.....	6
2.2.1	Composants d'un nœud capteur.....	6
2.2.2	Architecture du réseau de capteurs.....	6
2.3	Communication dans les réseaux de capteurs	7
2.3.1	Architecture de communication basée sur le modèle OSI	7
2.3.2	Technologies de la communication dans les réseaux de capteurs	8
3.	Agrégation de données dans les réseaux de capteurs	9
3.1	Problématique d'agrégation	9
3.2	Définition	9
3.3	Types d'agrégation de données	10
3.4	Protocoles de l'agrégation.....	11
3.4.1	Low Energy Adaptive Clustering Hierarchy (LEACH).....	11
3.4.2	Threshold sensitive Energy Efficient sensor Network protocol (TEEN)	11
3.4.3	COUGAR	12
3.5	Sécurité de l'agrégation de données.....	12
3.5.1	Problématique de la sécurité dans l'agrégation de données.....	12
3.5.2	Les principales formes d'attaques.....	13
4.	Réalisation d'agrégation de données.....	15
4.1	Présentation des outils de réalisation	15
4.1.1	TinyOS	15
4.1.2	TOSSIM	16
4.2	Résultats de la simulation dans TOSSIM.....	16
5.	Conclusion.....	20
6.	Bibliographie.....	21

1. Introduction

En 2003, selon le magazine Technology Review du MIT, le réseau de capteurs sans fil est l'une des dix nouvelles technologies qui bouleverseront le monde et notre manière de vivre et de travailler. Il répond à l'émergence ces dernières décennies, de l'offre et d'un besoin accru d'observation et de contrôler des phénomènes physiques et biologiques dans différents domaines.

Un réseau de capteurs sans fil est composé d'un ensemble d'unités de traitement, appelées « motes », communiquant via des liens sans fil. Les différences entre les réseaux traditionnels (réseaux télécommunication, Internet, etc.) et les réseaux de capteurs sont les suivants:

- Les réseaux de capteurs ont plus de nœuds avec une plus haute densité
- Les nœuds dans les réseaux de capteurs sont assez fragiles et vulnérables à diverses formes de défaillances : cassure, faible énergie, etc.

En informatique et télécommunication, les réseaux de capteurs sans fil sont un domaine de recherche active.

Notre groupe s'occupe de la notion d'agrégation de données dans les réseaux de capteurs. Comme les ressources des motes sont limitées, la transmission des données individuelle de chaque mote vers la station de base n'est pas convenable. Donc il faut trouver des méthodes pour agréger les données pendant leur transmission et ainsi économiser l'énergie, ce qui est un des principaux défis de ce genre de technologie.

Nos objectifs :

1. Etudier les réseaux de capteurs sans fils
2. Etudier l'agrégation de données dans les réseaux de capteurs
3. Développer une méthode d'agrégation des données dans un modèle de réseau de capteurs

2. Réseaux de capteurs sans fils

2.1 Présentation des réseaux de capteurs

Un réseau de capteurs sans fil est un réseau ad hoc avec un grand nombre de nœuds qui sont des micro-capteurs capables de récolter et de transmettre des données environnementales d'une manière autonome. La position de ces nœuds n'est pas obligatoirement prédéterminée. Ils peuvent être aléatoirement dispersés dans une zone géographique, appelée « champ de captage » correspondant au terrain d'intérêt pour le phénomène capté (ex : laché de capteurs sur un volcan pour étudier les phénomènes volcanologiques et leurs évolutions, etc....)

2.1.1 Histoire des réseaux de capteurs

Les récents progrès des techniques ont provoqué un énorme intérêt dans les réseaux sans fil. La technologie de réseaux de capteurs sans fil est devenue une des merveilles technologiques dans le 21^{ème} siècle et les réseaux de capteurs ont montré leur impact sur notre vie quotidienne.

Dans « Sensor Networks: Evolution, Opportunities, and Challenges » [6], CHEE-YEE CHONG et SRIKANTA P. KUMAR ont parlé de trois générations de nœuds de capteurs :

Génération	Période	Taille	Poids	Batterie
1 ^{er}	Les années 80 et 90	Grande boîte à chaussures	Kilogrammes	Grosse
2 ^e	2000-2003	Boîte de cartes	Grammes	AA
3 ^e	2010	Particule de poussière	Négligeable	Solaire

Grâce aux progrès des techniques sans fil, les réseaux de capteurs seront aussi communs dans nos vies quotidiennes que les ordinateurs et l'Internet.

2.1.2 Applications

La diminution de taille et de coût des micro-capteurs, l'élargissement de la gamme des types de capteurs disponibles (thermique, optique, vibrations...) et l'évolution des supports de communication sans fil, ont élargi le champ d'application des réseaux de capteurs. Parmi elles, nous citons :

- **Applications militaires** : On peut penser à un réseau de capteurs

déployé sur un endroit stratégique ou difficile d'accès, afin de surveiller toutes les activités des forces ennemies, ou d'analyser le terrain avant d'y envoyer des troupes (détection d'agents chimiques, biologiques ou de radiations).

- **Applications domestiques** : En plaçant, sur le plafond ou dans le mur, des capteurs, on peut économiser l'énergie en gérant l'éclairage ou le chauffage en fonction de la localisation des personnes.
- **Applications environnementales** : Les réseaux de capteurs sont beaucoup appliqués dans ce domaine pour détecter des incendies, surveiller des catastrophes naturelles, détecter des pollutions et suivre des écosystèmes.
- **Applications agricoles** : Dans les champs agricoles, les capteurs peuvent être semés avec les graines. Ainsi, les zones sèches seront facilement identifiées et l'irrigation sera donc plus efficace.
- **Applications médicales** : Les réseaux de capteurs ont aussi des développements dans le domaine de diagnostic médical. Par exemple, des micro-caméras sont capables, sans avoir recours à la chirurgie, de transmettre des images de l'intérieur d'un corps humain avec une autonomie de 24 heures.
- **Applications transportés** : Il est possible d'intégrer des nœuds capteurs au processus de stockage et de livraison. Le réseau ainsi formé, pourra être utilisé pour connaître la position, l'état et la direction d'un paquet ou d'une cargaison.



Image 1 : Applications des réseaux de capteurs sans fil

2.2 Architecture d'un réseau de capteurs

2.2.1 Composants d'un nœud capteur

Un nœud capteur est composé principalement d'un processeur, une mémoire, un émetteur/récepteur radio, un ensemble de capteur et une pile.

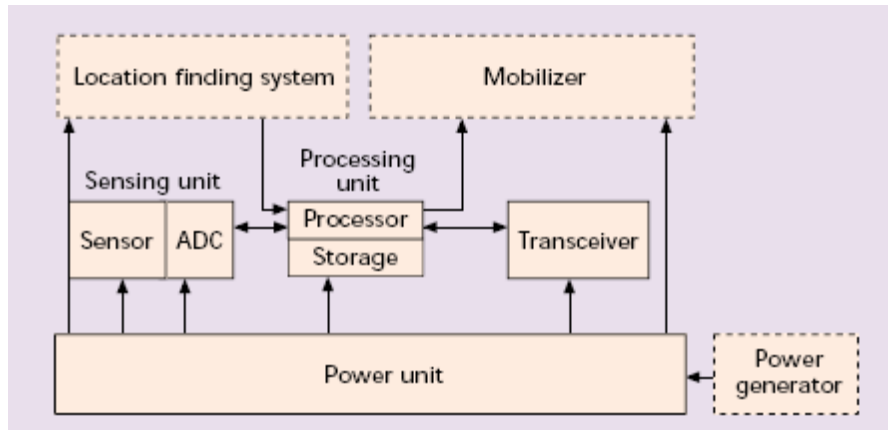


Image 2 : Composants d'un nœud capteur

2.2.2 Architecture du réseau de capteurs

Un réseau de capteurs sans fil est composé d'un ensemble de nœuds capteurs et des Gateway qui s'occupent de collecter les données des capteurs et de les transmettre à l'utilisateur via l'Internet ou le satellite.

Il existe plusieurs topologies pour les réseaux de capteurs :

- **Topologie en étoile**

La topologie en étoile est un système uni-saut. Tous les nœuds envoient et reçoivent seulement des données avec la station de base. Cette topologie est simple et elle demande une faible consommation d'énergie, mais la station de base est vulnérable et la distance entre les nœuds et la station est limitée.

- **Topologie en toile (Mesh Network)**

La topologie en toile est un système multi-saut. La communication entre les nœuds et la station de base est possible. Chaque nœud a plusieurs chemins pour envoyer des données. Cette topologie a plus de possibilités de passer à l'échelle du réseau, avec redondance et tolérance aux fautes, mais elle demande une consommation d'énergie plus importante.

- **Topologie hybride**

La topologie hybride est un mélange des deux topologies ci-dessus. Les stations de base forment une topologie en toile et les nœuds autour d'elles sont en topologie étoile. Elle assure la minimisation d'énergie dans les réseaux de capteurs.

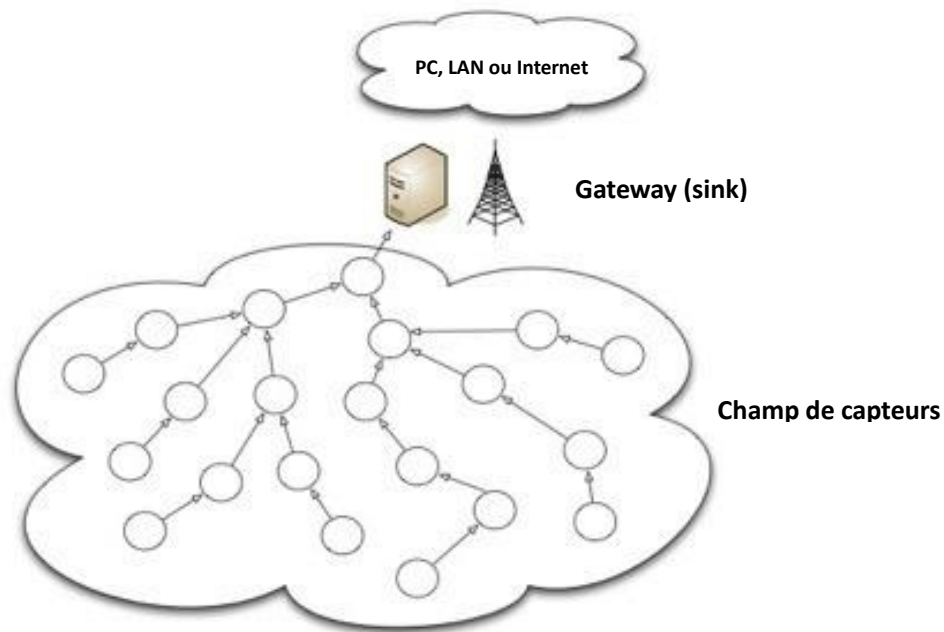


Image 3 : Topologie hybride d'un réseau de capteurs sans fil

2.3 Communication dans les réseaux de capteurs

2.3.1 Architecture de communication basée sur le modèle OSI

Le modèle de communication comprend cinq couches qui ont les mêmes fonctions que celles du modèle OSI ainsi que trois couches pour la gestion d'énergie, la gestion de la mobilité et la gestion des tâches.

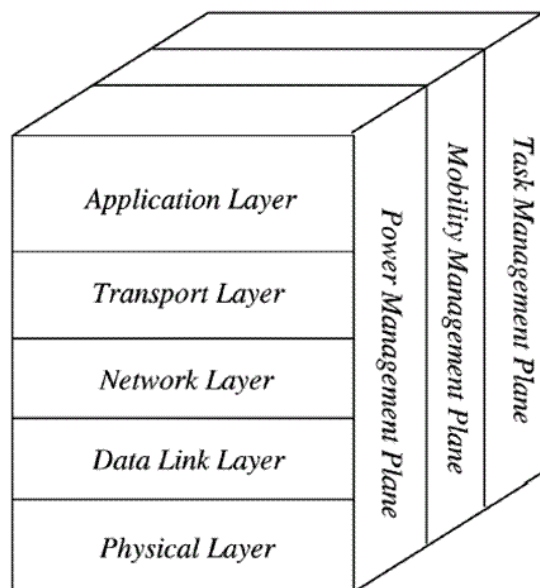


Image 4 : Modèle en couches du réseau de capteurs sans fil

Rôles des couches :

- ✓ **Couche physique** : Matériels pour envoyer et recevoir les données
- ✓ **Couche liaison de données** : Gestion des liaisons entre les nœuds et les stations de base, contrôle d'erreurs
- ✓ **Couche réseau** : Routage et transmission des données
- ✓ **Couche transport** : Transport des données, contrôle de flux
- ✓ **Couche application** : Interface pour les applications au haut niveau
- ✓ **Plan de gestion d'énergie** : Contrôle l'utilisation d'énergie
- ✓ **Plan de gestion de mobilité** : Gestion des mouvements des nœuds
- ✓ **Plan de gestion de tâche** : Balance les tâches entre les nœuds afin d'économiser de l'énergie

2.3.2 Technologies de la communication dans les réseaux de capteurs

Bluetooth / IEEE 802.15.4

Bluetooth est une spécification de l'industrie des télécommunications. Elle utilise une technique radio courte distance destinée à simplifier les connexions entre les appareils électroniques. Malheureusement, un grand défaut de cette technologie est sa trop grande consommation d'énergie.

ZigBee / IEEE 802.15.4

ZigBee est une norme de transmission de données sans fil permettant la communication de machine à machine. Zigbee offre des débits de données moindres, mais sa très faible consommation électrique et ses coûts de production très bas en font une candidate idéale pour la domotique ou les matériels de type capteur, télécommande ou équipement de contrôle dans le secteur industriel.

Dash 7 / ISO/IEC 18000-7

Dash7 est une nouvelle technologie de réseaux de capteurs sans fil en utilisant la norme ISO/IEC 18000-7. Sa consommation électrique est très faible, la durée de vie de batterie peut arriver à plusieurs ans. Sa distance de communication est 2km. Elle fournit une faible latence pour le suivi des objets en mouvement, un protocole petite pile, des supports de capteurs et de sécurité et un débit de transmission allant jusqu'à 200kbits/s.

3. Agrégation de données dans les réseaux de capteurs

3.1 Problématique d'agrégation

Dans un capteur, la problématique principale concerne la consommation d'énergie : en effet, ces derniers doivent restés opérationnels le plus longtemps possibles, dans des conditions parfois difficiles (ex : lâchés par avion sur les parois d'un volcan). Comme il n'est pas possible de recharger leur énergie ni changer les piles par exemple (on ne sait pas toujours où se trouvent les capteurs), il est nécessaire d'économiser au maximum l'énergie consommée par ces derniers.

On estime que la transmission des données d'un capteur représente environ 70% de sa consommation d'énergie.

De plus, les réseaux de capteurs étant assez denses en général, cela signifie que des nœuds assez proches en terme de distance (voisins) peuvent capter les mêmes données (température, pression, humidité équivalentes par exemple) et donc il apparaît nécessaire d'introduire le mécanisme d'agrégation de données afin d'éviter la duplication d'information au sein du réseau de capteurs et donc de préserver leur énergie et donc d'augmenter la durée de vie du réseau.

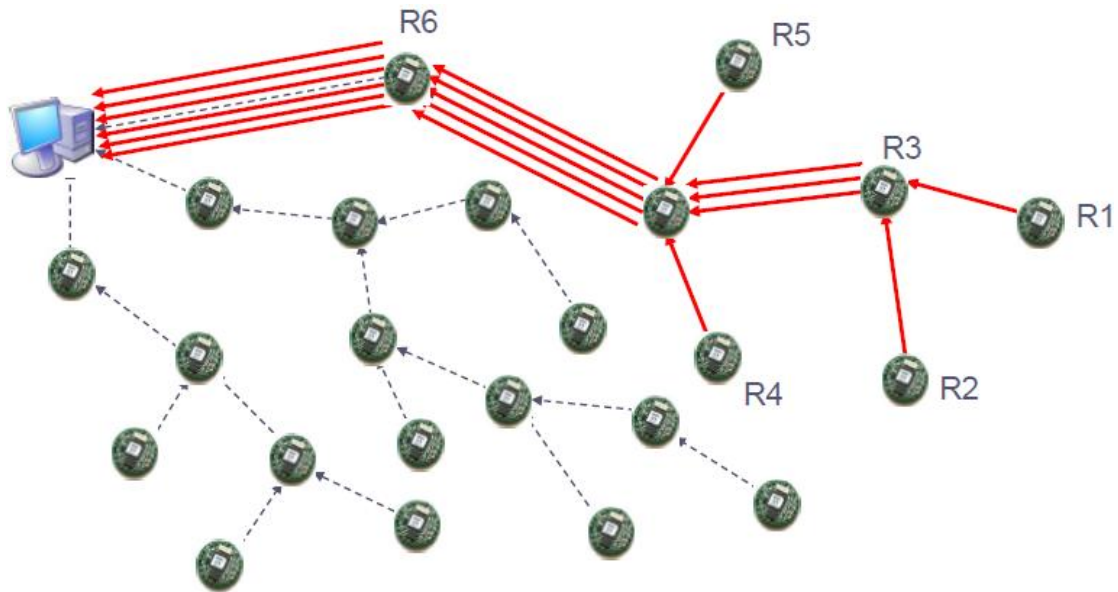
3.2 Définition

L'agrégation de données dans les réseaux de capteurs consiste à remplacer les lectures individuelles de chaque capteur par une vue globale, collaborative sur une zone donnée (clustering).

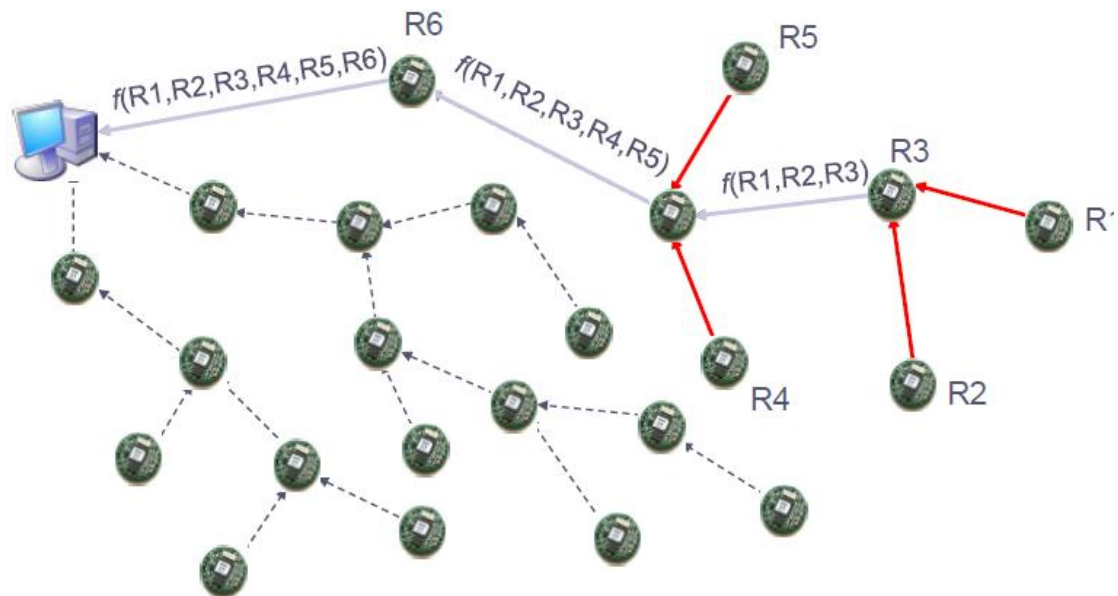
On peut utiliser par exemple de simples fonctions d'agrégat telles que MIN, MAX ou MOYENNE, qui permettent à partir d'une série de n messages reçus par un « chef de zone » (capteur chef d'une zone) de ne renvoyer vers le puits qu'un seul message résumant l'information contenue dans ces n messages.

Ceci réduit le nombre de messages envoyés et donc économise l'énergie.

Exemple sans agrégation :



Au total, 18 messages sont envoyés sur le réseau de capteurs.
 En utilisant le mécanisme d'agrégation de données, on obtient un total de 7 messages envoyés sur le réseau :



3.3 Types d'agrégation de données

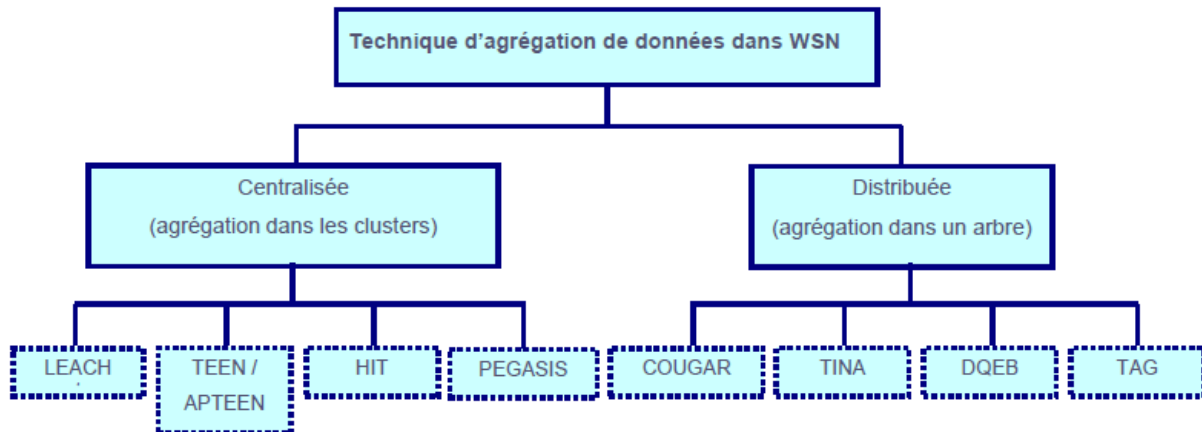
Les techniques d'agrégation de données peuvent être découpées en deux :

- **Agrégation centralisée**

Agrégation dans des clusters, formés via un protocole de clustering (classification). On définit d'abord des zones (clusters) via ce protocole puis ensuite on agrège les données dans ces zones grâce à un chef de zone. Ce chef peut éventuellement changer au cours du temps afin de répartir au mieux la consommation d'énergie entre tous les nœuds du réseau.

- **Agrégation distribuée**

Agrégation dans un arbre c'est-à-dire que le réseau est vu de façon globale.



3.4 Protocoles de l'agrégation

3.4.1 Low Energy Adaptive Clustering Hierarchy (LEACH)

LEACH est un protocole de routage hiérarchique qui utilise la classification (clustering) afin de diviser le réseau en deux parties : les cluster-heads (chefs de zone) et les nœuds membres. Ce protocole se décompose en deux phases qui se succèdent plusieurs fois : la construction et la communication.

Phase de construction :

Dans cette phase, on construit les différents clusters (zones) et on choisit leur chef. Chaque nœud du réseau choisit aléatoirement un nombre et si ce nombre est inférieur à une valeur, le nœud devient chef de zone (cluster-head). Ces chefs de zone sont élus en fonction de la force du signal reçu (proximité).

Phase de communication :

Les communications à l'intérieur d'une zone (cluster) sont effectuées avec la méthode TDMA (mode permettant de transmettre plusieurs signaux sur un seul canal) : chaque chef établit un *schedule TDMA* pour les membres de sa zone en indiquant pour chaque nœud son slot d'émission. Les membres émettent donc les données captées pendant ce slot d'émission. Ce mécanisme permet aux différents capteurs d'éteindre leur interface de communication en dehors de leur slot d'émission, ce qui a pour effet d'économiser leur énergie.

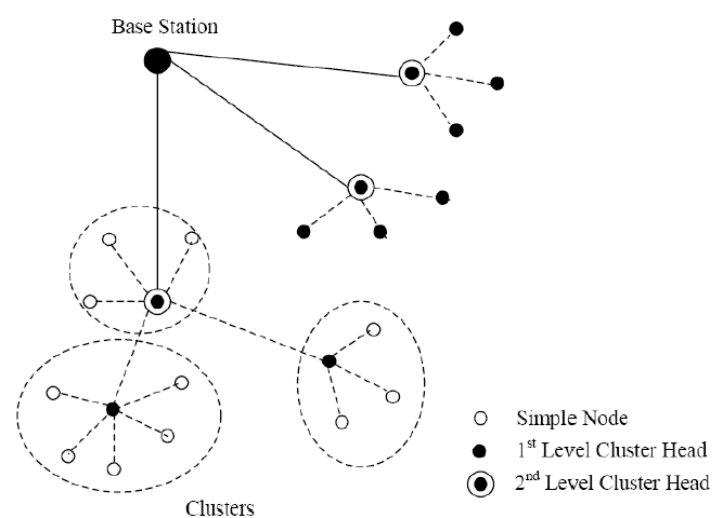
Ensuite les données agrégées (dans les chefs de zone) sont envoyées directement au puits.

3.4.2 Threshold sensitive Energy Efficient sensor Network protocol (TEEN)

Le protocole TEEN a été développé à partir de LEACH mais au contraire de LEACH qui est orienté temps, TEEN est quant à lui orienté événement.

La différence majeure entre ces deux protocoles résulte dans le fait que TEEN n'utilise pas TDMA qui est inadapté pour les applications orientées événements. Les chefs de zones élus dans TEEN ne transmettent donc pas de *schedule TDMA* à leurs membres, mais un message qui contient la tâche demandée au capteur, la valeur critique après laquelle les membres doivent envoyer leurs rapports de données (Hard Threshold) et le changement minimal obligeant le noeud à envoyer un nouveau rapport (Soft Threshold).

Donc lorsqu'un noeud détecte que sa valeur captée a dépassé le HT, il doit émettre un rapport au chef de zone, mais uniquement si sa valeur a change suffisamment (par rapport au ST). Le comportement est ainsi réactif (orienté événement) mais il limite également la consommation d'énergie.



3.4.3 COUGAR

Le protocole COUGAR est un exemple d'agrégation de données distribuée. Les données produites par le réseau sont modélisées comme une table relationnelle, le réseau étant vu comme une grande base de données distribuée. Les attributs de cette table sont soit des informations sur le capteur soit des données produites par ce capteur car COUGAR fournit une agrégation partielle au niveau des nœuds.

En effet, chaque nœud contient une liste d'attente de tous les nœuds fils qui doivent lui envoyer des données. Le nœud n'émet le paquet agrégé que lorsque tous les fils de sa file d'attente lui ont envoyé des données (+ timer au cas où un de ses fils deviendrait inaccessible).

Ce protocole nécessite donc une grande synchronisation afin de traiter les données dans le réseau (il faut attendre toutes les données avant d'effectuer le calcul de l'agrégation).

3.5 Sécurité de l'agrégation de données

3.5.1 Problématique de la sécurité dans l'agrégation de données

L'agrégation de données pose évidemment des problèmes liés à la sécurité dans les réseaux de capteurs. Ces problématiques de sécurité sont les mêmes que dans tous les systèmes informatiques à savoir :

- Confidentialité des données : se doit d'être respectée pour certaines applications (pas forcément nécessaire pour l'agriculture mais dans les domaines du transport ou le domaine médical c'est une autre histoire...).
- Intégrité des données : les données ne doivent pas avoir été modifiées (par accident = erreur de transmission ou intentionnellement = attaque)
- Disponibilité des données : technique de répartition de la consommation d'énergie (par exemple le fait que les chefs de zone changent régulièrement), diagnostic régulier du réseau afin de réagir si un nœud est compromis, etc...
- Authentification : permet de vérifier l'identité d'un envoyeur de données sur le réseau afin d'éviter l'injection de données par un tiers non autorisé.
- Non-répudiation : permet de s'assurer que le message a été émis et reçu par la personne qui a déclaré l'avoir fait. Un nœud qui envoie le résultat de ses calculs d'agrégation ne peut pas nier l'avoir envoyé (permet à la station de base de déterminer d'où vient l'erreur si erreur il y a)

3.5.2 Les principales formes d'attaques

Différentes sortes d'attaques sont potentiellement en mesure de mettre en péril l'utilisation d'un réseau de capteurs :

- **Attaque par déni de service (DoS)** : si on envoie des signaux radio pour brouiller (jamming) la transmission entre les différents nœuds du réseau (sur la même fréquence de communication), on rend indisponible le RCSF car les nœuds sont incapables de communiquer.
- **Compromission d'un nœud** : on peut avoir physiquement accès aux capteurs du fait de leur dispersion en pleine nature bien souvent. Un attaquant peut donc éventuellement extraire des informations de ces capteurs puisqu'ils sont en sa possession physique.
- **Attaque sibylline** : l'attaquant se fait passer pour plusieurs capteurs différents afin de fausser les données et les résultats issus de ces mêmes données.
- **Attaque par routage sélectif** : l'attaquant s'occupe manuellement du routage sur un des nœuds du réseau et peut donc choisir s'il route les données ou non, quelle sera la destination des nœuds routés, filtrer certaines données, etc... Il modifie donc l'agrégat qu'aurait du envoyer le nœud compromis et donc fausse les résultats.
- **Attaque par répétition** : l'attaquant écoute et enregistre le trafic réseau (sniffing) dans le RCSF puis réinjecte les anciennes données plus tard dans le réseau, ce qui peut fausser le résultat de l'agrégation de données.
- **Attaque du lavabo** : l'attaquant compromet un nœud du réseau et le rend le plus attractif possible afin que les paquets lui soient envoyés avant d'être

envoyées vers le puits. L'attaquant peut ensuite rendre plus efficaces d'autres attaques car il faussera d'autant plus les résultats du fait que toutes les données transitent par lui.

- **Attaque furtive** : l'attaquant injecte de fausses données dans le réseau sans révéler son existence (pas de compromission d'un nœud), ce qui revient à injecter un faux agrégat et fausser les résultats.

Il semble évident que cette liste n'est pas exhaustive, mais elle présente les principales formes d'attaques menées contre les réseaux de capteurs sans fil en terme d'agrégation de données.

4. Réalisation d'agrégation de données

4.1 Présentation des outils de réalisation

Pour pouvoir travailler avec un réseau de capteurs sans fil, il est nécessaire d'avoir un environnement dédié à celui-ci, l'utilisation des systèmes d'exploitation traditionnels comme Windows ou Linux est impossible car ils ont beaucoup trop de fonctionnalités inutiles pour un réseau de capteurs.

C'est pourquoi il faut se tourner vers TinyOS.

4.1.1 TinyOS

La pile des capteurs est la plus grande contrainte au niveau du système car comme les capteurs se trouvent dans la majorité des cas dans des conditions atmosphériques extrêmes, cela rend le changement des piles très compliqué, voir même impossible.

Il est donc nécessaire de trouver une solution qui n'utilise pas autant d'énergie.

TinyOS est un système d'exploitation Open-Source spécialement dédié au réseau de capteurs sans fil. Il travaille sur une base d'association de composants, ce qui réduit significativement la taille du code.

Il travaille avec NesC, un langage dérivé du C qui a été conçu pour minimiser l'utilisation de mémoire des capteurs.

Installation de TinyOS :

L'installation de tinyOS s'est avérée ne pas être aussi simple que d'installer une application normale.

Nous avons essayé plusieurs méthodes avant de réussir à l'installer correctement.

Tout d'abord nous avons essayé de l'installer sur Windows avec Cygwin car M.Beckit dans sa présentation des réseaux de capteurs avait cette configuration, mais malheureusement pour nous, nous avons rencontré beaucoup de problèmes avec les versions de Python et Java et même après avoir mis les bonnes versions, le programme ne fonctionnait pas. Alors comme nos ordinateurs portables était aussi sous Fedora 13, nous avons suivi un tutoriel pas à pas mais là non plus nous ne nous expliquons pas pourquoi l'installation n'a pas abouti.

Alors nous avons finalement utilisé un tutoriel pour une version de Ubuntu qui nous a permis d'installer correctement TinyOS.

Utilisation de TinyOS :

L'environnement TinyOS nous offre quelques applications déjà prêtes à l'emploi dans le répertoire suivant: « *opt/tinyos-2.1.0/apps* »

Exemples (Blink, RadioCountToLeds)

4.1.2 TOSSIM

TOSSIM est un simulateur à événements discrets pour TinyOS. Au lieu de compiler une application TinyOS pour l'exécuter sur les capteurs, les utilisateurs peuvent compiler dans le cadre TOSSIM, qui fonctionne sur un PC et qui lève la contrainte de posséder déjà les capteurs pour faire tourner l'application (simulation d'un réseau de capteurs).

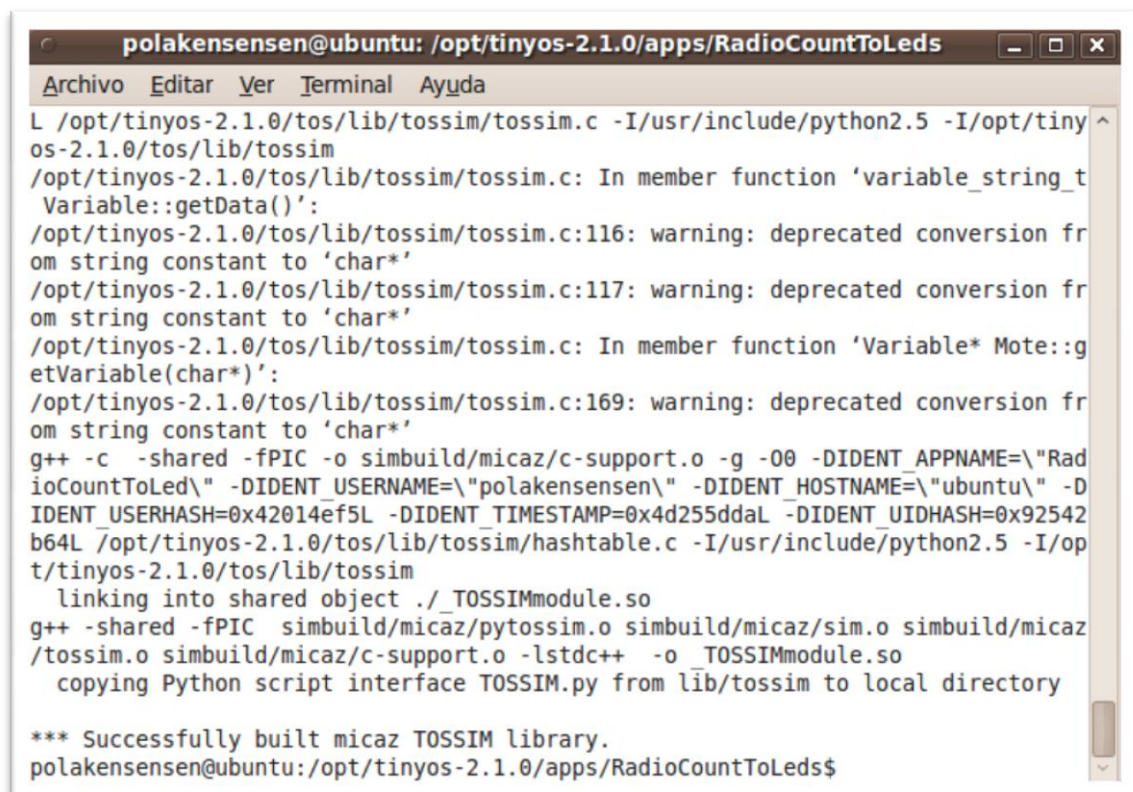
Cela permet aux utilisateurs de déboguer, tester et analyser des algorithmes dans un environnement contrôlé et reproductible. Comme TOSSIM fonctionne sur un PC, les utilisateurs peuvent examiner leur code TinyOS à l'aide de débogueurs et autres outils de développement.

4.2 Résultats de la simulation dans TOSSIM

Pour la simulation nous avons travaillé avec un exemple qui s'appelle RadioCountToLeds (tutoriel suivi sur le site officiel de TinyOS). Cet exemple se trouve dans l'arbre d'exemples déjà disponible avec l'installation TinyOS.

En premier lieu on doit se situer à l'intérieur du dossier et faire la construction de la simulation avec l'instruction suivante :

"user@ubuntu:/opt/tinyos-2.1.0/apps/RadioCountToLeds\$ make micaz sim"



```
polakensensen@ubuntu: /opt/tinyos-2.1.0/apps/RadioCountToLeds
Archivo Editar Ver Terminal Ayuda
L /opt/tinyos-2.1.0/tos/lib/toSSIM/toSSIM.c -I/usr/include/python2.5 -I/opt/tiny
os-2.1.0/tos/lib/toSSIM
/opt/tinyos-2.1.0/tos/lib/toSSIM/toSSIM.c: In member function 'variable_string_t
Variable::getData()':
/opt/tinyos-2.1.0/tos/lib/toSSIM/toSSIM.c:116: warning: deprecated conversion fr
om string constant to 'char*'
/opt/tinyos-2.1.0/tos/lib/toSSIM/toSSIM.c:117: warning: deprecated conversion fr
om string constant to 'char*'
/opt/tinyos-2.1.0/tos/lib/toSSIM/toSSIM.c: In member function 'Variable* Mote::g
etVariable(char*)':
/opt/tinyos-2.1.0/tos/lib/toSSIM/toSSIM.c:169: warning: deprecated conversion fr
om string constant to 'char*'
g++ -c -shared -fPIC -o simbuild/micaz/c-support.o -g -O0 -DIDENT_APPNAME="\Rad
ioCountToLed\" -DIDENT_USERNAME="\polakensensen\" -DIDENT_HOSTNAME="\ubuntu\" -D
IDENT_USERHASH=0x42014ef5L -DIDENT_TIMESTAMP=0x4d255ddaL -DIDENT_UIDHASH=0x92542
b64L /opt/tinyos-2.1.0/tos/lib/toSSIM/hashtable.c -I/usr/include/python2.5 -I/op
t/tinyos-2.1.0/tos/lib/toSSIM
linking into shared object ./_TOSSIMmodule.so
g++ -shared -fPIC simbuild/micaz/pytoSSIM.o simbuild/micaz/sim.o simbuild/micaz
/toSSIM.o simbuild/micaz/c-support.o -lstdc++ -o _TOSSIMmodule.so
copying Python script interface TOSSIM.py from lib/toSSIM to local directory

*** Successfully built micaz TOSSIM library.
polakensensen@ubuntu:/opt/tinyos-2.1.0/apps/RadioCountToLeds$
```

La simulation a été effectuée avec succès malgré qu'il existe beaucoup d'avertissements : ces avertissements sont en majorité des avertissements pour les versions de python, java et autres.

Nous devons créer ensuite un fichier qui s'appelle « topologies.txt », lequel contient les nœuds, à qui envoyer le message et finalement la fréquence à laquelle le message est envoyé. Un exemple de fichier :

```
1 2 -54.0
2 1 -55.0
1 3 -60.0
3 1 -60.0
2 3 -64.0
3 2 -64.0
```

Après avoir créé le fichier topologies il faut créer un autre fichier qui s'appelle « meyer-heavy.txt » et qui contient la fréquence de bruit (Noise) qui affecte la topologie. Nous appelons ce fichier pour que la simulation soit la plus réelle possible et s'approche le plus possible d'un environnement réel. Ce fichier nous a été donné par W. Bechkit qui simule des données réelles.

Un exemple de ce fichier :
(comme le fichier est extrêmement long, nous n'en mettons que quelques lignes)

```
-39
-98
-98
-98
-99
-98
-94
-98
-98
-98
```

Il est à savoir que le nom des fichiers topologies.txt et meyer-heavy.txt ne sont pas des noms définis obligatoirement, ils peuvent prendre n'importe quel nom, l'important étant qu'il soit correctement appelé dans le fichier python.

Finalement nous ajoutons le script réalisé en python pour effectuer la simulation :

```
#!/usr/bin/python
from TOSSIM import *
import sys

t = Tossim([]) // pour créer un objet tossim
r = t.radio()
f = open("topologies.txt", "r") // ouvre le fichier topologies en mode
lecture

lines = f.readlines()
for line in lines:
    s = line.split()
    if (len(s) > 0):
        print " ", s[0], " ", s[1], " ", s[2];
        r.add(int(s[0]), int(s[1]), float(s[2]))
// cette partie fait la lecture de chaque ligne du fichier topologies
t.addChannel("RadioCountToLedsC", sys.stdout)
t.addChannel("Boot", sys.stdout)

noise = open("meyer-heavy.txt", "r")
lines = noise.readlines()
for line in lines:
    str1 = line.strip()
    if (str1 != ""):
```

```
    val = int(str1)
    for i in range(1, 4):
        t.getNode(i).addNoiseTraceReading(val)
for i in range(1, 4):
    print "Creating noise model for ",i;
    t.getNode(i).createNoiseModel()
// cette partie fait la lecture de chaque ligne du fichier meyer-heavy

t.getNode(1).bootAtTime(100001);
t.getNode(2).bootAtTime(800008);
t.getNode(3).bootAtTime(1800009);
// initialisation de chaque noeud
for i in range(0, 100):
    t.runNextEvent() // pour lancer l'application dans chaque nœud
```

5. Conclusion

Malheureusement nous avons rencontré des difficultés avec le travail sur TinyOS et Tossim, ce qui ne nous a pas laissé le temps de développer une application en rapport avec l'agrégation de données.

Malgré n'avoir pas réussi à finaliser le travail, le domaine des réseaux de capteurs sans fil représente bel et bien une des technologies du futur car elle peut être utilisée dans de nombreux domaines tels que l'agriculture, l'armée, les surveillances pour les catastrophes naturelles, les carrefours sur les routes, etc...

Nous tenons à remercier M. Bouabdallah pour ce sujet d'étude innovant ainsi que l'aide et le temps précieux de Monsieur W. Bechkit.

6. Bibliographie

- [1] « Réseaux de capteurs UTC», Yacine CHALLAL
- [2] Réseaux de capteurs sans fils
http://fr.wikipedia.org/wiki/R%C3%A9seau_de_capteurs_sans_fil
- [3] Zigbee
<http://fr.wikipedia.org/wiki/Zigbee>
- [4] Bluetooth
<http://fr.wikipedia.org/wiki/Bluetooth>
- [5] Dash7
<http://en.wikipedia.org/wiki/DASH7>
- [6] « Sensor Networks: Evolution, Opportunities, and Challenges», CHEE-YEE CHONG et SRIKANTA P. KUMAR
- [7] «Wireless sensor networks: a survey», I.F.Akyildiz, Y.Sankarasubramaniam, W.Su, E.Cayirci
- [8] “Sécurité de l’agrégation de données dans les réseaux de capteurs”, A. Brianceau et J. Christin
- [9] “Clustering and Wireless Sensor Networks : a paradigm shift ?”, T. Watteyne et I. Augé-Blum
- [10] “Node Clustering in Wireless Sensor Networks by considering Structural Characteristics of the Network Graph”, N. Dimokas, D. Katsaros et Y. Manolopoulos
- [11] “Réseaux ad-hoc”, I. Guérin Lassous